From implicit to recursive equations

Joris van der Hoeven

From implicit to recursive equations*

Joris van der Hoeven

LIX, CNRS
École polytechnique
91128 Palaiseau Cedex
France

Email: vdhoeven@lix.polytechnique.fr
Web: http://lix.polytechnique.fr/~vdhoeven

August 30, 2018

The technique of relaxed power series expansion provides an efficient way to solve so called recursive equations of the form $F = \Phi(F)$, where the unknown F is a vector of power series, and where the solution can be obtained as the limit of the sequence 0, $\Phi(0)$, $\Phi(\Phi(0))$, ... With respect to other techniques, such as Newton's method, two major advantages are its generality and the fact that it takes advantage of possible sparseness of Φ. In this paper, we consider more general implicit equations of the form $\Phi(F) = 0$. Under mild assumptions on such an equation, we will show that it can be rewritten as a recursive equation.

Keywords: Implicit equation, relaxed power series, algorithm
A.M.S. subject classification: 68W25, 42-04, 68W30, 65G20, 30B10

1. Introduction

Let $\mathbb{K}$ be an effective field of constants of characteristic zero. This means that elements in $\mathbb{K}$ can be encoded by data structures on a computer and that we have algorithms for performing the field operations of $\mathbb{K}$.

Let $F = (F^{[1]}, ..., F^{[r]})$ be a column vector of r indeterminate series in $\mathbb{K}[[z]]$. We may also consider F as a power series $F_0 + F_1 z + \cdots \in \mathbb{K}^r[[z]]$. Let $\Phi(F) = (\Phi(F)^{[1]}, ..., \Phi(F)^{[r]})$ be a column vector of expressions built up from F, z and constants in $\mathbb{K}$ using ring operations, differentiation and integration (with constant term zero). Finally, let $C_0, ..., C_{l-1} \in \mathbb{K}^r$ be a finite number of initial conditions. Assume that the system

$$\left\{ \begin{array}{rcl} \Phi(f) & = & 0 \\ f_0 & = & C_0 \\ & \vdots & \\ f_{l-1} & = & C_{l-1} \end{array} \right. \tag{1}$$

admits a unique solution $f \in \mathbb{K}[[z]]^r$. In this paper, we are interested in the efficient computation of this solution up to a given order n.

In the most favourable case, the equation $\Phi(f) = 0$ is of the form

$$f - \Psi(f) \;=\; 0, \tag{2}$$

where the coefficient $\Psi(f)_n$ of z^n in $\Psi(f)$ only depends on earlier coefficients $f_0, ..., f_{n-1}$ of f, for each $n \in \mathbb{N}$. In that case,

$$f_n \;=\; \Psi(f)_n$$

*. This work has been supported by the ANR-09-JCJC-0098-01 MaGiX and ANR-10-BLAN-0109 projects, as well as a Digiteo 2009-36HD grant and Région Ile-de-France.

actually provides us with a recurrence relation for the computation of the solution. Using the technique of relaxed power series expansions [Hoe02, Hoe07], which will briefly be recalled in section 2.4, it is then possible to compute the expansion $f_{;n} = f_0 + \cdots + f_n z^n$ at order n in time

$$\mathsf{T}(n) \;=\; s\,\mathsf{R}(n) + O(t\,n), \tag{3}$$

where s is the number of multiplications occurring in Ψ, t is the total size of Ψ as an expression, and $\mathsf{R}(n)$ denotes the complexity of relaxed multiplication of two power series at order n. Here we assume that Ψ is represented by a directed acyclic graph, with possible common subexpressions. For large n, it was shown in [Hoe02, FS74, Hoe03, LS16] that $\mathsf{R}(n) = O(\mathsf{M}(n)\log n)$, where $\mathsf{M}(n) = O(n \log n \log\log n)$ denotes the complexity [CT65, SS71, CK91] of multiplying two polynomials of degrees $< n$. More recently, it has been shown [Hoe14, Hoe07] that we even have $\mathsf{R}(n) = n \log n\, \mathrm{e}^{\sqrt{2\log 2}\sqrt{\log\log n}}\, (\log\log n)^{O(1)}$. For moderate n, when polynomial multiplication is done naively or using Karatsuba's method, relaxed multiplication is as efficient as the truncated multiplication of polynomials at order n [Hoe97, Hoe02].

One particularly important example of an equation of the above type is the integration of a dynamical system

$$f \;=\; f_0 + \int \Psi(f), \tag{4}$$

where Ψ is algebraic (i.e. does not involve differentiation or integration). In that case, given the solution f at order n, we may consider the linearized system

$$E' \;=\; \Psi(f) + J_\Psi(f)\, E + O(z^{2n})$$

at order $2\,n$, where $J_\Psi(f)$ stands for the Jacobian matrix associated to Ψ at f. If we have a fundamental system of solutions of $E' = J_\Psi(f)\, E$ at order n, then one step of Newton's method allows us to find the solution of (4) and a new fundamental system of solutions of the linearized equation at order $2\,n$ [BK78, Sed01, BCO+07]. A careful analysis shows that this leads to an algorithm of time complexity

$$\mathsf{T}(n) \;=\; \mathsf{M}(n)\,(2\,s\,r + 2\,s + 13/6\,r^2 + 4/3\,r + o(1)) + O(t\,r\,n + r^3\,n). \tag{5}$$

In [Hoe10], this bound has been further improved to

$$\mathsf{T}(n) \;=\; \mathsf{M}(n)\,(2\,s + 4/3\,r + o(1)) + O(t\,n), \tag{6}$$

under the assumptions that $\mathbb{K}$ admits sufficiently many 2^p-th roots of unity and that $r = O(\log n)$.

Although the complexity (5) is asymptotically better than (3) for very large n, the relaxed approach often turns out to be more efficient in practice. Indeed, Newton's method both suffers from a larger constant factor and the fact that we profit less from the potential sparsity of the system. In particular, if $r > \log n$, then the relaxed approach is generally faster. Moreover, as long as multiplications are done in the naive or Karatsuba model, the relaxed approach is optimal in the sense that the computation of the solution takes roughly the same time as its verification. Another advantage of the relaxed approach is that it generalizes to more general functional equations and partial differential equations.

Let us now return to our original implicit system (1). If Φ is a system of differentially algebraic equations, then we may also seek to apply Newton's method. For non degenerate systems and assuming that we have computed the solution f and a fundamental system of solutions for the linearized equation at order n, one step of Newton's method yields an extension of the solutions at order $2\,n - i$, for a fixed constant $i \in \mathbb{N}$. From an asymptotic point of view, this means that the complexities (5) and (6) remain valid, modulo multiplication of r by the differential order of the system in these bounds.

Another approach for the resolution of (1) is to keep differentiating the system with respect to f until it becomes equivalent to a system of the form (2). For instance, if Φ is algebraic, then differentiation of (1) yields

$$J_\Phi(f)\, f' + \frac{\partial \Phi}{\partial z}(f) \;=\; 0.$$

Consequently, if $J_\Phi(f)_0$ is invertible, then

$$f \;=\; f_0 - \int J_\Phi(f)^{-1} \frac{\partial \Phi}{\partial z}(f)$$

provides us with an equivalent system that can be solved by one of the previous methods. Unfortunately, this method requires the computation of the Jacobian, so we do not longer exploit the potential sparsity of the original system.

Yet another recent approach [Hoe09] is to consider not yet computed coefficients of f as formal unknowns, and solve the system of equations $\Phi(f)_0 = \cdots = \Phi(f)_n = 0$ for increasing values of n. For large n, the system $\Phi(f)_0 = \cdots = \Phi(f)_n = 0$ usually reduces to a linear system of equations. In particular, the coefficients of series with unknown coefficients are not polynomials but merely linear combinations. Using the so called "substitution product", the multiplication of series with unknown coefficients can be done while taking advantage of this linearity.

In this paper, we will present a variant of the approach of [Hoe09]. Roughly speaking, we reconsider the series with unknown coefficients as vectors of partially unknown series. Technically speaking, this is done *via* the concept of anticipators, which will be introduced in section 3. Using this technique, and under mild assumptions, we show in section 4.1 how to rewrite the original system of equations into a new system that is both recursive and not much larger in size. We may then apply a standard relaxed algorithm for its resolution. In section 4.2 we show that this leads to slightly sharper complexity bounds than those from [Hoe09]. Roughly speaking, in the case of a system of r differential equations of order $O(r)$ that must be evaluated at order $n+i$ in order to determine the solution at order n, we prove the following generalization of (3):

$$\mathsf{T}(n) = s\,\mathsf{R}(n) + O(\mathsf{M}(i)\, s\, n + i\, t\, n) + O(i^2 r^3 n). \tag{7}$$

Especially for larger values of r, this still compares favourably with respect to (5). Another major benefit of the new technique is the fact that it allows for the direct resolution of implicit equations using existing software for recursive equations.

For algebraic equations over the p-adic numbers, relaxed algorithms with similar complexities have been developed in [BL12, Leb15]. The index i in (7) corresponds to the opposite of the notion of shift in [Leb15]. We notice that a preprint version of the present paper [Hoe11] appeared shortly before [BL12], so the main ideas in the present paper were not influenced by [BL12, Leb15]. Predecessors of the algorithms in this paper were implemented in MATHEMAGIX (see [Hoe09]), as well as the algorithms from [BL12, Leb15]. Timings tend to reflect the theoretical complexity bounds. The new algorithms from this paper have not been implemented yet.

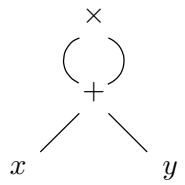
Our algorithm can for instance been used for the computation of power series solutions to differential-algebraic systems of equations. The well known example of a pendulum is treated in detail in section 5. The reader may wish to fast forward to this example when struggling through the more technical parts of this paper.

Acknowledgments. We wish to thank the referees for their careful reading and their comments and suggestions.

2. Preliminaries

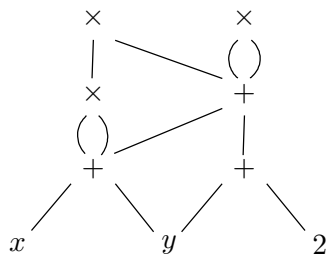
2.1. Dags

Assume that we have fixed a set Ω of function symbols, together with an **arity** n_ω for each $\omega \in \Omega$. Then a **dag** over Ω is a rooted finite directed acyclic Ω-labeled graph, such that each ω-labeled node has n_ω successors, together with an ordering on the successors. For instance,



is a typical dag for the expression $(x + y)^2$, with $\Omega = \{x, y, +, \times\}$, $n_x = n_y = 0$ and $n_+ = n_\times = 2$. We will denote by t_Φ the number of nodes of a dag Φ (also called its size) and by s_Φ its number of multiplications (also called its multiplicative size). For our example dag Φ, we thus have $t_\Phi = 4$ and $s_\Phi = 1$. We will denote by $\mathbb{D}_\Omega$ the set of dags over Ω.

More generally, we may consider **multivariate dags** with an arbitrary number of roots, which again come with ordering. For instance,



is a bivariate dag that represents a vector of two expressions $(x + y)^2$ $((x + y) + (y + 2))$ and $((x + y) + (y + 2))^2$. We will denote by d_Φ the number of roots of a multivariate dag Φ, which we will also call its **dimension**. We will write $\Phi = (\Phi^{[1]}, ..., \Phi^{[d_\Phi]})$, where $\Phi^{[i]}$ stands for the subdag whose root is the i-th root of Φ. We will denote by $\mathbb{D}^d_\Omega$ the set of multivariate dags over Ω of dimension d and $\mathbb{D}^*_\Omega = \mathbb{D}^1_\Omega \cup \mathbb{D}^2_\Omega \cup \cdots$.

2.2. Dags as operators on power series

Consider a linear operator $\omega : \mathbb{K}[[z]] \to \mathbb{K}[[z]]$. We say that ω is a **coefficientwise** operator, if there exist fixed constants $\omega_0, \omega_1, \omega_2, ... \in \mathbb{K}$ such that

$$\omega(f_0 + f_1 z + f_2 z^2 + \cdots) \;=\; \omega_0 f_0 + \omega_1 f_1 z + \omega_2 f_2 z^2 + \cdots,$$

for all $f = \sum_i f_i z^i \in \mathbb{K}[[z]]$. For every $c \in \mathbb{K}$, the operator $\times_c$ defined by

$$\times_c(f) \;=\; c f$$

is an example of a coefficientwise operator. The truncation operators $\top_i : f \mapsto f_{i;}$, $\top^j : f \mapsto f_{;j}$ and $\top^j_i : f \mapsto f_{i;j}$ defined by

$$\begin{aligned} f_{i;} &= f_i z^i + f_{i+1} z^{i+1} + \cdots \\ f_{;j} &= f_0 + \cdots + f_j z^j \\ f_{i;j} &= f_i z^i + \cdots + f_j z^j, \end{aligned}$$

constitute another family of examples. We will denote by $\times_{\mathbb{K}}$ and $\top_{\mathbb{N}}$ the sets of all operators of the form $\times_c$ with $c \in \mathbb{K}$ resp. $\top_i$ with $i \in \mathbb{N}$. Finally, we define the coefficientwise operators $\theta = z\,\partial/\partial z$ and θ^{-1} by

$$\begin{aligned}\theta(f_0 + f_1 z + f_2 z^2 + \cdots) &= f_1 z + 2 f_2 z^2 + \cdots\\ \theta^{-1}(f_0 + f_1 z + f_2 z^2 + \cdots) &= f_1 z + \tfrac{1}{2} f_2 z^2 + \cdots.\end{aligned}$$

The operator θ is called the **Euler derivation** with respect to z and we notice that θ^{-1} is the inverse of θ on $z\,\mathbb{K}[[z]]$. The Euler derivation admits the advantage with respect to the ordinary derivation that $\operatorname{val}\theta(f) \geqslant \operatorname{val} f$ for all $f \in \mathbb{K}[[z]]$, where "val" stands for the valuation in z. Nevertheless, any differential equation for the usual derivation can be rewritten as a differential equation for the Euler derivation: it suffices to multiply the equation by a sufficiently large power of z.

Let $F = (F^{[1]}, \ldots, F^{[r]})$ be r "indeterminate series" in $\mathbb{K}[[z]]$. We will sometimes consider F as a series with formal coefficients

$$\begin{aligned}F &= F_0 + F_1 z^1 + \cdots,\\ F_n &= (F_n^{[1]}, \ldots, F_n^{[r]}).\end{aligned}$$

Let Λ be a set of coefficientwise linear operators. In what follows, we will take

$$\Omega_{F,\Lambda} = \mathbb{K}[z] \cup \{F^{[1]}, \ldots, F^{[r]}, +, -, \times\} \cup \times_{\mathbb{K}} \cup \top_{\mathbb{N}} \cup \Lambda$$

and denote by $\mathbb{D}_{F,\Lambda}$ the set of dags over $\Omega_{F,\Lambda}$. Similarly, we set $\mathbb{D}^d_{F,\Lambda} = \mathbb{D}^d_{\Omega_{F,\Lambda}}$ and $\mathbb{D}^*_{F,\Lambda} = \mathbb{D}^*_{\Omega_{F,\Lambda}}$. Dags in $\mathbb{D}^*_{F,\varnothing}$, $\mathbb{D}^*_{F,\{\theta\}}$ and $\mathbb{D}^*_{F,\{\theta^{-1}\}}$ will respectively be called **algebraic**, **differential** and **integral**. Notice that polynomials in $\mathbb{K}[z]$ are regarded as dags of size 1, independently of their degree; this is motivated by the fact that coefficient extraction is trivial for explicit polynomials.

Clearly, any dag $\Phi \in \mathbb{D}^d_{F,\Lambda}$ can be considered as a function $\mathbb{K}[[z]]^r \to \mathbb{K}[[z]]^d$; $f \mapsto \Phi(f)$. Given a small symbolic perturbation $E = (E^{[1]}, \ldots, E^{[r]}) \in z^n\,\mathbb{K}[[z]]^r$, we may expand $\Phi(F+E)$ as a Taylor series in E

$$\Phi(F+E) = \Phi(F) + (D\Phi)(F)(E) + \tfrac{1}{2}(D^2\Phi)(F)(E) + \cdots$$

and truncation at order $2n$ yields

$$\Phi(F+E) = \Phi(F) + (D\Phi)(F)(E) + O(z^{2n}).$$

We claim that $(D\Phi)(F)(E)$ can be regarded as a dag in $\mathbb{D}_{(F,E),\Lambda}$. For instance, if

$$\Phi(F) = F\theta F + z\theta^{-1}(F\theta^{-1}F),$$

then

$$(D\Phi)(F)(E) = E\theta F + F\theta E + z\theta^{-1}(E\theta^{-1}F + F\theta^{-1}E).$$

In general, the claim is easily shown by induction over t_Φ.

2.3. Dags as series

We claim that any dag $\Phi \in \mathbb{D}^d_{F,\Lambda}$ can be regarded as a series in z

$$\Phi = \Phi_0 + \Phi_1 z + \cdots,$$

such that each coefficient Φ_n is a dag over

$$\Omega_{F,\Lambda;n} = \mathbb{K} \cup \{F_0^{[1]}, \ldots, F_0^{[r]}, \ldots\ldots, F_n^{[1]}, \ldots, F_n^{[r]}, +, -, \times\} \cup \times_{\mathbb{K}} \cup \Lambda.$$

Indeed, by induction over the size of Φ, we first define the valuation v_Φ of Φ by

$$\begin{aligned}
v_P &= \operatorname{val} P \qquad (P \in \mathbb{K}[z]) \\
v_{F^{[i]}} &= 0 \\
v_{\Psi \pm \Xi} &= \min(v_\Psi, v_\Xi) \\
v_{\Psi \Xi} &= v_\Psi + v_\Xi \\
v_{\omega(\Psi)} &= \max(v_\Psi, \min\{n : \omega_n \neq 0\}) \qquad (\omega \in \times_{\mathbb{K}} \cup \top_{\mathbb{N}} \cup \Lambda) \\
v_{(\Phi^{[1]}, \ldots, \Phi^{[d]})} &= \min(v_{\Phi^{[1]}}, \ldots, v_{\Phi^{[d]}}) \qquad (d \geqslant 2)
\end{aligned}$$

We next define the coefficients Φ_n by another induction over the size of Φ. If $n < v_\Phi$, then we take $\Phi_n = 0$. Otherwise, we take

$$\begin{aligned}
P_n &= P_n \qquad (P \in \mathbb{K}[z]) \\
\top_i(\Psi)_n &= \Psi_n \qquad (i \in \mathbb{N}, n \geqslant i) \\
(\Psi \pm \Xi)_n &= \Psi_n \pm \Xi_n \\
(\Psi\, \Xi)_n &= \Psi_{v_\Psi}\, \Xi_{n - v_\Psi} + \Psi_{v_\Psi + 1}\, \Xi_{n - v_\Psi - 1} + \cdots + \Psi_{n - v_\Xi}\, \Xi_{v_\Xi} \\
\omega(\Psi)_n &= \omega_n\, \Psi_n \qquad (\omega \in \times_{\mathbb{K}} \cup \Lambda) \\
(\Phi^{[1]}, \ldots, \Phi^{[d]})_n &= (\Phi_n^{[1]}, \ldots, \Phi_n^{[d]}) \qquad (d \geqslant 2)
\end{aligned}$$

As a result of the claim, we emphasize that Φ_n only depends on the coefficients $F_0, \ldots, F_n$ in $z^0, \ldots, z^n$ of F.

Remark 1. The formula for $(\Psi\ \Xi)_n$ can be replaced by any algebraically equivalent formula, as long as $(\Psi\, \Xi)_n$ only depends on $\Psi_0, \ldots, \Psi_{n - v_\Xi}$ and $\Xi_0, \ldots, \Xi_{n - v_\Psi}$. Assuming the concept of relaxed power series, to be introduced below, this allows us to compute $\Psi\ \Xi$ using the formula

$$\Psi\, \Xi = [(\Psi \operatorname{div} z^{v_\Psi})\, (\Xi \operatorname{div} z^{v_\Xi})]\, z^{v_\Psi + v_\Xi},$$

where we may use a relaxed algorithm for the multiplication $(\Psi \operatorname{div} z^{v_\Psi})\, (\Xi \operatorname{div} z^{v_\Xi})$. From now on, we will assume that all products $\Psi\ \Xi$ are expanded in this way.

2.4. Relaxed power series

Let us briefly recall the technique of relaxed power series computations, which is explained in more detail in [Hoe02]. In this computational model, a power series $f \in \mathbb{K}[[z]]$ is regarded as a stream of coefficients f_0, f_1, ... When performing an operation $g = \Phi(f_1, \ldots, f_k)$ on power series it is required that the coefficient g_n of the result is output as soon as sufficiently many coefficients of the inputs are known, so that the computation of g_n does not depend on the further coefficients. For instance, in the case of a multiplication $h = f\, g$, we require that h_n is output as soon as $f_0, \ldots, f_n$ and $g_0, \ldots, g_n$ are known. In particular, we may use the naive formula $h_n = \sum_{i=0}^{n} f_i\, g_{n-i}$ for the computation of h_n.

The additional constraint on the time when coefficients should be output admits the important advantage that the inputs may depend on the output, provided that we add a small delay. For instance, the exponential $g = \exp f$ of a power series $f \in z\, \mathbb{K}[[z]]$ may be computed in a relaxed way using the formula

$$g = \int f'\, g.$$

Indeed, when using the naive formula for products, the coefficient g_n is given by

$$g_n = \tfrac{1}{n}\, (f_1\, g_{n-1} + 2\, f_2\, g_{n-2} + \cdots + n\, f_n\, g_0),$$

and the right-hand side only depends on the previously computed coefficients $g_0, \ldots, g_{n-1}$.

The main drawback of the relaxed approach is that we cannot directly use fast algorithms on polynomials for computations with power series. For instance, assuming that $\mathbb{K}$ has sufficiently many 2^p-th roots of unity and that field operations in $\mathbb{K}$ can be done in time $O(1)$, two polynomials of degrees $\leqslant n$ can be multiplied in time $\mathsf{M}(n) = O(n \log n)$, using FFT multiplication [CT65]. Given the truncations $f_{;n} = f_0 + \cdots + f_n z^n$ and $g_{;n} = g_0 + \cdots + g_n z^n$ at order n of power series $f, g \in \mathbb{K}[[z]]$, we may thus compute the truncated product $(fg)_{;n}$ in time $\mathsf{M}(n)$ as well. This is much faster than the naive $O(n^2)$ relaxed multiplication algorithm for the computation of $(fg)_{;n}$. However, the formula for $(fg)_0$ when using FFT multiplication depends on all input coefficients $f_0, \ldots, f_{n-1}$ and $g_0, \ldots, g_{n-1}$, so the fast algorithm is not relaxed. Fortunately, efficient relaxed multiplication algorithms do exist:

Theorem 2. [Hoe97, Hoe02, FS74] *Let $\mathsf{M}(n)$ be the time complexity for the multiplication of polynomials of degrees $< n$ in $\mathbb{K}[z]$. Then there exists a relaxed multiplication algorithm for series in $\mathbb{K}[[z]]$ of time complexity $\mathsf{R}(n) = O(\mathsf{M}(n) \log n)$.*

Theorem 3. [Hoe07] *There exists a relaxed multiplication algorithm of time complexity $\mathsf{R}(n) = n \log n \, \mathrm{e}^{\sqrt{2\log 2}\sqrt{\log\log n}} (\log\log n)^{O(1)}$.*

In what follows, we will denote by $\mathsf{R}(n)$ the complexity of relaxed multiplication at order n. Let us now consider a general equation of the form

$$f = \Phi(f), \tag{8}$$

where $\Phi \in \mathbb{D}^r_{F,\Lambda}$ an r-dimensional dag. We say that (8) is a **recursive equation**, if each coefficient $\Phi(F)_n$ only depends on earlier coefficients $F_0, \ldots, F_{n-1}$ of F. That is, $\Phi(F)_n \in \mathbb{D}^r_{(F_0,\ldots,F_{n-1}),\Lambda}$ for all n. In order to solve (8) at order n, we then need to perform s_Φ relaxed multiplications at order n and t_Φ coefficientwise operations $+, -$ or $\omega \in \times_{\mathbb{K}} \cup \mathbb{C}$ at order n. This yields the following complexity bound:

Proposition 4. *Any recursive equation (8) can be solved at order n in time*

$$T(n) = s_\Phi \mathsf{R}(n) + O(t_\Phi n).$$

3. Anticipators

When solving an implicit equation in f using a relaxed algorithm, the coefficients f_n are computed only gradually. During the resolution process, it might happen that we wish to evaluate dags at higher orders than the number of known coefficients of f. That is, given $\Phi \in \mathbb{D}^*_{F,\Lambda}$ and $i \geqslant 1$, we might need $\Phi(f)_{n+i}$, even though only $f_0, \ldots, f_n$ are known. In that case, we have a problem, but we may still do the best we can, and compute $\Phi(f_0 + \cdots + f_n z^n)_{n+i}$ instead of $\Phi(f)_{n+i}$.

This motivates the introduction of the i-th order **anticipator** $\Phi^{\langle i \rangle}$ of Φ by

$$\begin{aligned} \Phi^{\langle i \rangle}(F)_{;i-1} &= 0 \\ \Phi^{\langle i \rangle}(F)_{n+i} &= \Phi(F_{;n})_{n+i}, \end{aligned}$$

where we recall that $F_{i;j} = F_i z^i + \cdots + F_j z^j$ and $F_{;j} = F_{0;j}$. On the one hand, we will show in this section that $\Phi^{\langle 0 \rangle}, \ldots, \Phi^{\langle i \rangle}$ can be computed simultaneously by a dag Ψ of multiplicative size $s_\Psi = s_\Phi$ and total size $t_\Psi = O(i\, t_\Phi + \mathsf{M}(i)\, s_\Phi)$. On the other hand, we will show that $\Phi^{\langle i \rangle}$ is essentially a linear perturbation of Φ, that can be computed explicitly.

3.1. Computation of $\Phi^{\langle i\rangle}$ as a dag

Let us show how to compute a dag for $\Phi^{\langle i\rangle}$. The following rules are straightforward:

$$\begin{aligned} P^{\langle i\rangle} &= P_{i;} \qquad (P \in \mathbb{K}[z]) \\ (F^{[k]})^{\langle i\rangle} &= 0 \\ (\Phi \pm \Psi)^{\langle i\rangle} &= \Phi^{\langle i\rangle} \pm \Psi^{\langle i\rangle} \\ \omega(\Phi)^{\langle i\rangle} &= \omega(\Phi^{\langle i\rangle}) \qquad (\omega \in \times_{\mathbb{K}} \cup \top_{\mathbb{N}} \cup \Lambda). \end{aligned}$$

As to multiplication, for $n \geqslant i-1$, we have

$$\begin{aligned} (\Phi\,\Psi)^{\langle i\rangle}(F)_{n+i} &= \sum_{k=0}^{n+i} \Phi(F_{;n})_k\, \Psi(F_{;n})_{n+i-k} \\ &= \sum_{k=i}^{n} \Phi(F_{;n})_k\, \Psi(F_{;n})_{n+i-k} + \\ &\quad \sum_{k=0}^{i-1} \Phi(F_{;n})_k\, \Psi(F_{;n})_{n+i-k} + \Phi(F_{;n})_{n+i-k}\, \Psi(F_{;n})_k \\ &= \sum_{k=i}^{n} \Phi(F)_k\, \Psi(F)_{n+i-k} + \\ &\quad \sum_{k=0}^{i-1} \Phi(F)_k\, \Psi^{\langle i-k\rangle}(F)_{n+i-k} + \Phi^{\langle i-k\rangle}(F)_{n+i-k}\, \Psi(F)_k \\ &= \left[(\Phi_{i;}\,\Psi_{i;})(F) + \sum_{k=0}^{i-1} \left(\Phi(F)_k\, \Psi^{\langle i-k\rangle}(F) + \Psi(F)_k\, \Phi^{\langle i-k\rangle}(F)\right) z^k \right]_{n+i}, \end{aligned}$$

where $\Phi_{i;} = \Phi - \Phi_0 - \cdots - \Phi_{i-1}\, z^{i-1}$ stands for the operator with $\Phi_{i;}(F) = \Phi(F)_{i;} = \Phi(F)_i\, z^i + \Phi(F)_{i+1}\, z^{i+1} + \cdots$ and similarly for $\Psi_{i;}$. Consequently,

$$(\Phi\,\Psi)^{\langle i\rangle} = P^{\langle i\rangle}_{\Phi,\Psi} + \Phi_{i;}\,\Psi_{i;} + \sum_{k=0}^{i-1} \left(\Phi(F)_k\, \Psi^{\langle i-k\rangle} + \Psi(F)_k\, \Phi^{\langle i-k\rangle}\right) z^k, \tag{9}$$

for some polynomial $P^{\langle i\rangle}_{\Phi,\Psi} \in \mathbb{K}[z]$ with $\operatorname{val} P^{\langle i\rangle}_{\Phi,\Psi} \geqslant i$ and $\deg P^{\langle i\rangle}_{\Phi,\Psi} \leqslant 2\,i-2$ (in particular, $P^{\langle 1\rangle}_{\Phi,\Psi} = 0$). Notice also that

$$\begin{aligned} \Phi_{i-1;}\,\Psi_{i-1;} &= \Phi_{i;}\,\Psi_{i;} + \left(\Phi(F)_{i-1}\,\Psi_{i;} + \Psi(F)_{i-1}\,\Phi_{i;}\right) z^{i-1} + \\ &\quad \Phi(F)_{i-1}\,\Psi(F)_{i-1}\, z^{2i-2}. \end{aligned} \tag{10}$$

Now assume that $\Phi^{\langle 0\rangle}, \ldots, \Phi^{\langle i\rangle}$ and $\Psi^{\langle 0\rangle}, \ldots, \Psi^{\langle i\rangle}$ are known. Then we may simultaneously compute $(\Phi\,\Psi)^{\langle 0\rangle}, \ldots, (\Phi\,\Psi)^{\langle i\rangle}$ in the following way:

$$\begin{aligned} \Phi_{1;} &= \Phi - \Phi_0 \\ &\vdots \\ \Phi_{i;} &= \Phi_{i-1;} - \Phi_{i-1}\, z^{i-1} \\ \Psi_{1;}, \ldots, \Psi_{i;} &\quad \text{similarly} \\ \Phi_{i;}\,\Psi_{i;} & \\ \Phi_{i-1;}\,\Psi_{i-1;}, \ldots, \Phi_{0;}\,\Psi_{0;} &\quad \text{using formula (10)} \\ (\Phi\,\Psi)^{\langle 1\rangle}, \ldots, (\Phi\,\Psi)^{\langle i\rangle} &\quad \text{using formula (9).} \end{aligned}$$

This computation involves one series product and $O(i^2)$ additions and scalar multiplications. For large i, we may further reduce the cost to $O(\mathsf{M}(i))$ since the computation of $(\Phi\,\Psi)^{\langle 1\rangle}, ..., (\Phi\,\Psi)^{\langle i\rangle}$ really comes down to the computation of two truncated power series products $\Phi(F)\,(\Psi^{\langle 0\rangle} + \Psi^{\langle 1\rangle}\,z + \cdots)$ and $\Psi(F)\,(\Phi^{\langle 0\rangle} + \Phi^{\langle 1\rangle}\,z + \cdots)$ at order i. In summary, we obtain

LEMMA 5. *Given $\Phi \in \mathbb{D}^*_{F,\Lambda}$, there exists a simultaneous dag Ψ for $\Phi^{\langle 0\rangle}$, ..., $\Phi^{\langle i\rangle}$ of multiplicative size $s_\Psi = s_\Phi$ and total size $t_\Psi = O(i\,t_\Phi + \mathsf{M}(i)\,s_\Phi)$.*

3.2. Computation of $\Phi^{\langle i\rangle}$ as a perturbation of Φ

Since $\Phi(F)_n$ only depends on $F_0, ..., F_n$, we notice that

$$\Phi^{\langle 0\rangle} \;=\; \Phi.$$

In general, for $n \geqslant i$ and $E = F_{n+1;n+i}$, we may expand

$$\begin{aligned}\Phi(F_{;n} + E) &= \Phi(F_{;n}) + (D\Phi)(F_{;n})(E) + O(z^{2n+2})\\ &= \Phi(F_{;n}) + (D\Phi)(F_{;i-1})(E) + O(z^{n+i+1}).\end{aligned}$$

Let $\mathbb{e}_{[k]}$ denote the k-th basis element of $\mathbb{K}^r$, so that $F_j = F_j^{[1]}\,\mathbb{e}_{[1]} + \cdots + F_j^{[r]}\,\mathbb{e}_{[r]}$ for all j. When considering F as a column vector, it follows by linearity that

$$\Phi(F)_{n+i} \;=\; \Phi^{\langle i\rangle}(F)_{n+i} + \Phi^{\{i,1\}}(n)\,F_{n+1} + \cdots + \Phi^{\{i,i\}}(n)\,F_{n+i}, \tag{11}$$

where $\Phi^{\{i,j\}}$ is a row matrix whose k-th entry is given by

$$\Phi^{\{i,j\}}_{[k]}(n) \;=\; [D(\Phi)(F_{;i-1})(z^{n+j}\,\mathbb{e}_{[k]})]_{n+i}.$$

Notice that $\Phi^{\{i,j\}}(n)$ depends on n, but $D(\Phi)(F_{;i-1})$ does not. Let us investigate the functions $\Phi^{\{i,j\}}(n)$ more closely for some important examples.

Example 6. If Φ is algebraic, then we have

$$D(\Phi)(F_{;i-1})(z^{n+j}\,\mathbb{e}_{[k]}) \;=\; \frac{\partial\Phi}{\partial F^{[k]}}(F_{;i-1})\,z^{n+j},$$

whence

$$\Phi^{\{i,j\}}_{[k]}(n) \;=\; \frac{\partial\Phi}{\partial F^{[k]}}(F_{;i-1})_{i-j}. \tag{12}$$

In particular, $\Phi^{\{i,j\}}(n)$ is actually constant.

Example 7. If Φ is differential, of differential order d (this means that d is the maximal number of θ-nodes on a path from the root of Φ to a leaf), then, considering Φ as a differential polynomial in $F, \theta F, ..., \theta^d F$, we have

$$D(\Phi)(F_{;i-1})(z^{n+j}\,\mathbb{e}_{[k]}) \;=\; \frac{\partial\Phi}{\partial F^{[k]}}(F_{;i-1})\,z^{n+j} + \cdots + \frac{\partial\Phi}{\partial\,(\theta^d F^{[k]})}(F_{;i-1})\,(\theta^d\,z^{n+j}),$$

whence

$$\Phi^{\{i,j\}}_{[k]}(n) \;=\; \left[\frac{\partial\Phi}{\partial F^{[k]}}(F_{;i-1})\right]_{i-j} + \cdots + \left[\frac{\partial\Phi}{\partial\,(\theta^d F^{[k]})}(F_{;i-1})\right]_{i-j}(n+j)^d \tag{13}$$

is a polynomial of degree at most d.

Example 8. Similarly, if Φ is algebraic in $F, \int F, ..., \int^d F$, where $\int = \theta^{-1}\,z$, then

$$D(\Phi)(F_{;i-1})(z^{n+j}\,\mathbb{e}_{[k]}) \;=\; \frac{\partial\Phi}{\partial F^{[k]}}(F_{;i-1})\,z^{n+j} + \cdots + \frac{\partial\Phi}{\partial\,(\int^d F^{[k]})}(F_{;i-1})\,(\int^d z^{n+j}),$$

whence

$$\Phi_{[k]}^{\{i,j\}}(n) \;=\; \left[\frac{\partial\Phi}{\partial F^{[k]}}(F_{;i-1})\right]_{i-j} + \cdots + \left[\frac{\partial\Phi}{\partial\,(\int^d F^{[k]})}(F_{;i-1})\right]_{i-j-d} \frac{1}{(n+j)\cdots(n+j+d-1)}. \tag{14}$$

Consequently, there exists a polynomial $A_{i,j,k}$ of degree $\leqslant d$ with

$$\Phi_{[k]}^{\{i,j\}}(n) \;=\; \frac{A_{i,j,k}(n)}{(n+j)\,(n+j+1)\cdots(n+j+d-1)},$$

for all $n \geqslant i$.

Example 9. For more general integral dags Φ, it can be checked by induction over the size of Φ that $\Phi_{[k]}^{\{i,j\}}(n)$ is still a rational function in n, that remains bounded for $n \to \infty$, and whose denominator has integer coefficients. Similarly, for any dag $\Phi \in \mathbb{D}_{F,\Lambda}$ where $\Lambda \subseteq \{\theta, \theta^{-1}\}$, the expression $\Phi_{[k]}^{\{i,j\}}(n)$ is a rational function in n, whose denominator has integer coefficients.

4. Relaxed resolution of implicit equations

Assume now that we want to solve a system of power series equations

$$\left\{\begin{array}{rcl} \Phi(f) &=& 0 \\ f_0 &=& C_0 \\ &\vdots& \\ f_{l-1} &=& C_{l-1} \end{array}\right. \tag{15}$$

where $\Phi = (\Phi^{[1]}, ..., \Phi^{[r]}) \in \mathbb{D}_{F,\Lambda}^r$ is a vector of dags and $C_0, ..., C_{l-1} \in \mathbb{K}^r$ a finite number of initial conditions. For definiteness, it is also important that (15) admits a unique solution f. This will be guaranteed by an even stronger technical assumption to be detailed below. Roughly speaking, for a given **index** $i \leqslant \frac{1}{2}(l+1)$, we will assume that each coefficient f_n with $n \geqslant l$ can be determined as a function of the previous coefficients $f_0, ..., f_{n-1}$ using only the equations $\Phi(f)_0 = \cdots = \Phi(f)_{n+i-1} = 0$. In fact, we will only use the equations $\Phi(f)_n = \cdots = \Phi(f)_{n+i-1} = 0$, which requires us to assume that $n - i \geqslant i - 1, ..., n - 1 \geqslant i - 1$ in order to determine f_n *via* (11); this explains why we need $l \geqslant 2\,i - 1$ initial conditions.

4.1. Construction of a recursive equation

Let $i \leqslant \frac{1}{2}(l+1)$. For each n and $j \in \{1, ..., i\}$, we introduce the $r \times r$ matrix

$$M_n^{\{j\}} \;=\; \begin{pmatrix} (\Phi^{[1]})_{[1]}^{\{i,j\}}(n) & \cdots & (\Phi^{[1]})_{[r]}^{\{i,j\}}(n) \\ \vdots & & \vdots \\ (\Phi^{[r]})_{[1]}^{\{i,j\}}(n) & \cdots & (\Phi^{[r]})_{[r]}^{\{i,j\}}(n) \end{pmatrix},$$

the $i\,r \times (2\,i-1)\,r$ block matrix

$$\boldsymbol{M}_n \;=\; \begin{pmatrix} M_{n-i}^{\{1\}} & \cdots & M_{n-i}^{\{i\}} & & \\ & \ddots & & \ddots & \\ & & M_{n-1}^{\{1\}} & \cdots & M_{n-1}^{\{i\}} \end{pmatrix}, \tag{16}$$

the $(i-1)\,r \times 1$, $i\,r \times 1$ and $(2\,i-1)\,r \times 1$ block column vectors

$$\check{\boldsymbol{f}}_n \;=\; \begin{pmatrix} f_{n-(i-1)} \\ \vdots \\ f_{n-1} \end{pmatrix}, \qquad \boldsymbol{f}_n \;=\; \begin{pmatrix} f_{n-(i-1)} \\ \vdots \\ f_n \end{pmatrix}, \qquad \hat{\boldsymbol{f}}_n \;=\; \begin{pmatrix} f_{n-(i-1)} \\ \vdots \\ f_{n+i-1} \end{pmatrix},$$

and the $i\,r \times 1$ column vector

$$\boldsymbol{g}_n \;=\; \begin{pmatrix} -\Phi^{\langle i\rangle}(f)_n \\ \vdots \\ -\Phi^{\langle i\rangle}(f)_{n+i-1} \end{pmatrix}.$$

In view of (11), the equations $\Phi(f)_n = \cdots = \Phi(f)_{n+i-1} = 0$ then translate into

$$\boldsymbol{M}_n\, \hat{\boldsymbol{f}}_n \;=\; \boldsymbol{g}_n.$$

We will say that (15) is ***i*-predictive** or **predictive** of **index *i***, if, for all $n \geqslant l$, there exist $r \times i\,r$ and $r \times (i-1)\,r$ matrices $\boldsymbol{P}_n$ and $\boldsymbol{Q}_n$, such that

$$\boldsymbol{P}_n\, \boldsymbol{M}_n \;=\; (\; \boldsymbol{Q}_n \;\; \mathrm{Id}_r \;\; 0\;).$$

In that case, we have

$$\begin{aligned} \boldsymbol{P}_n\, \boldsymbol{M}_n\, \hat{\boldsymbol{f}}_n \;&=\; \boldsymbol{Q}_n\, \check{\boldsymbol{f}}_n + f_n \\ &=\; \boldsymbol{P}_n\, \boldsymbol{g}_n, \end{aligned}$$

whence

$$f_n \;=\; \boldsymbol{P}_n\, \boldsymbol{g}_n - \boldsymbol{Q}_n\, \check{\boldsymbol{f}}_n \tag{17}$$

provides us with an explicit formula for f_n. Now let $\boldsymbol{P}$ and $\boldsymbol{Q}$ be the operators on vectors of power series V with the property that $(\boldsymbol{P}V)_n = \boldsymbol{P}_n\, V_n$ and $(\boldsymbol{Q}\,V)_n = \boldsymbol{Q}_n\, V_n$. Then we may rewrite (17) into

$$f \;=\; \boldsymbol{P} \begin{pmatrix} -\Phi^{\langle i\rangle}(f) \\ \vdots \\ -z^{-(i-1)}\, \Phi^{\langle i\rangle}(f) \end{pmatrix} - \boldsymbol{Q} \begin{pmatrix} z^{i-1}\, f \\ \vdots \\ z\, f \end{pmatrix} + C' \tag{18}$$

for a suitable vector C' of polynomials in $\mathbb{K}[z]$ of degree $<l$. This is the desired recursive equation for f.

Example 10. If Φ is algebraic, then we recall from Example 6 that

$$\Phi^{\{1,1\}}_{[k]}(n) = \left(\frac{\partial \Phi}{\partial F^{[k]}}(C_0)\right)_0$$

for all k. If $i = 1$, then it follows that the matrix

$$\boldsymbol{M}_n = M^{\{1\}}_{n-1} = \left(\frac{\partial \Phi}{\partial F}(C_0)\right)_0$$

does not depend on n and that it simply equals the evaluation of the Jacobian matrix of Φ with respect to F at the "initial condition" C_0. In particular, the equation (15) is 1-predictive if and only if this matrix is invertible.

Example 11. Assume, as in Example 7, that Φ is of differential order d. If $i = 1$, then it follows in a similar way as above that

$$\boldsymbol{M}_n = \left(\frac{\partial \Phi}{\partial F}(C_0) + \cdots + \frac{\partial \Phi}{\partial (\theta^d\, F)}(C_0)\, n^d\right)_0,$$

so we may regard $\boldsymbol{M}_n$ as the evaluation at $N = n$ of a matrix $\boldsymbol{M}^*$ in $\mathbb{K}[N]^{r\times r}$ of degree $\leqslant d$ in N. In particular, the equation (15) is 1-predictive for a sufficiently large value of l if and only if $\boldsymbol{M}^*$ admits an inverse in $\mathbb{K}(N)^{r\times r}$. More precisely, if $\boldsymbol{M}^*$ is invertible, then we need l to be larger than each of the poles of $(\boldsymbol{M}^*)^{-1}$ (which are finite in number).

Example 12. If Φ is algebraic and $C \in \mathbb{K}[[z]]^r$ is such that the Jacobian matrix $\partial\Phi/\partial F$ evaluated at C admits an inverse $V \in \mathbb{K}((z))^{r\times r}$ with Laurent series coefficients, then it can be shown that the system $\Phi(f) = 0$, $f_{;l-1} = C_{;l-1}$ is i-predictive for $i = 1 - \operatorname{val} V$, whenever $l \geqslant 2\,i - 1$. Notice that Example 10 is a special case of this situation when $\operatorname{val} V = 0$ and $i = 1$. The observation actually generalizes to more general dags Φ, for a suitable interpretation of the inverse V as a matrix of operators acting on $\mathbb{K}((z))[\log z]^{r\times r}$ and assuming that l is taken sufficiently large. This also means that the method of this paper can be applied to the same class of equations as a suitable generalization of Newton's method. However, the details behind these claims are beyond the scope of this paper.

Remark 13. The main sources of unpredictivity are an insufficient number of initial conditions and the existence of multiple solutions. In the latter case, we may usually restore predictivity by differentiating the equation a finite number of times.

4.2. Complexity analysis

Let us first consider the case of an algebraic dag Φ. In that case, the matrix $\boldsymbol{M} = \boldsymbol{M}_n$ does not depend on n and its coefficients are explicitly given by (12). We may now determine $r \times i\,r$ and $r \times (i-1)\,r$ matrices $\boldsymbol{P}$ and $\boldsymbol{Q}$ with

$$\boldsymbol{P}\boldsymbol{M} \;=\; (\;\boldsymbol{Q}\;\; \mathrm{Id}_r\;\; 0\;), \tag{19}$$

using Gaussian elimination in order, and whenever such matrices exist. The equation (15) is i-predictive, if and only if this is indeed possible.

THEOREM 14. *Consider an i-predictive equation (15), such that Φ is algebraic. Then we may compute n terms of its unique solution f in time*

$$T_{\mathrm{alg}}(n) \;=\; s_\Phi\,\mathsf{R}(n) + O(\mathsf{M}(i)\,s_\Phi\,n + i\,t_\Phi\,n) + O(i\,r^2\,n).$$

Proof. By what precedes, the operators $\boldsymbol{P}$ and $\boldsymbol{Q}$ in (18) are really the constant matrices from (19). By lemma 5, the size of the righthand side of (18) as a dag is therefore bounded by $O(\mathsf{M}(i)\,s_\Phi + i\,t_\Phi + i\,r^2)$ and its multiplicative size is exactly s_Φ. The result thus follows from proposition 4. □

Assume now that $\Lambda \subseteq \{\theta, \theta^{-1}\}$. Then we claim that there exists an algorithms for checking i-predictivity and constructing a general formula for the corresponding matrices $\boldsymbol{P}_n$ and $\boldsymbol{Q}_n$. Indeed, we recall from section 3.2 that $\boldsymbol{M}_n$ is the evaluation at $N = n$ of a matrix $\boldsymbol{M}^*$ with entries in $\mathbb{K}(N)$ and denominators in $\mathbb{Z}[N]$. We may thus use Gaussian elimination in order to compute $r \times i\,r$ and $r \times (i-1)\,r$ matrices $\boldsymbol{P}^*$ and $\boldsymbol{Q}^*$ with entries in $\mathbb{K}(N)$ and

$$\boldsymbol{P}^*\,\boldsymbol{M}^* \;=\; (\;\boldsymbol{Q}^*\;\; \mathrm{Id}_r\;\; 0\;),$$

whenever such matrices exist. For those $n \geqslant l$ that are not positive integer roots of one of the denominators of the entries of $\boldsymbol{P}^*$, we now have $\boldsymbol{P}_n = \boldsymbol{P}^*(n)$ and $\boldsymbol{Q}_n = \boldsymbol{Q}^*(n)$. For each of the finite number of integer roots $n \geqslant l$, we may directly compute the matrices $\boldsymbol{P}_n$ and $\boldsymbol{Q}_n$ by Gaussian elimination over $\mathbb{K}$, whenever such matrices exist.

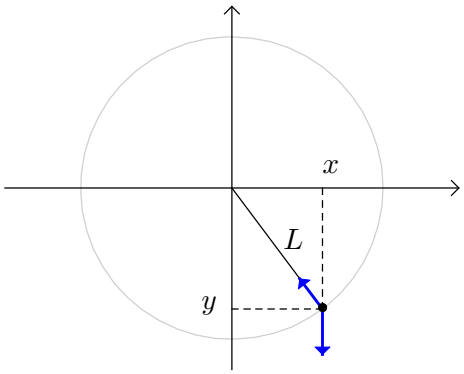


Figure 1. Illustration of the equations of the pendulum.

THEOREM 15. *Consider an i-predictive equation (15) and let d be the maximal degree of an entry of $\boldsymbol{M}^*$. Then we may compute n terms of the solution f to (15) in time*

$$T(n) = s_\Phi \mathsf{R}(n) + O(\mathsf{M}(i)\, s_\Phi\, n + i\, t_\Phi\, n) + O((i^2\, r^3 + i\, r^2\, d)\, n).$$

Proof. The computation of $\boldsymbol{M}^*$ (and the finite number of exceptional $\boldsymbol{M}_n$ for which n is a root of one of the denominators) is a precomputation. The determination of every next $\boldsymbol{M}_n$ can be done in time $O(i^2\, r^2 + i\, r^2\, d)$, via a diagonal translation of $\boldsymbol{M}_{n-1}$ and evaluation of the $O(i\, r^2)$ rational functions that are the entries of the bottom $r \times (2\, i - 1)\, r$ submatrix. Now assume that we maintain upper and lower triangular matrices $\boldsymbol{U}_n$, $\boldsymbol{L}_n$ and a permutation matrix $\boldsymbol{\Pi}_n$ at each stage such that $\boldsymbol{L}_n = \boldsymbol{U}_n\, \boldsymbol{\Pi}_n\, \boldsymbol{M}_n$. Then the determination of $\boldsymbol{U}_n$, $\boldsymbol{L}_n$ and $\boldsymbol{\Pi}_n$ as a function of $\boldsymbol{U}_{n-1}, \boldsymbol{L}_{n-1}$ and $\boldsymbol{\Pi}_{n-1}$ can be done in time $O(i^2\, r^3)$ using naive linear algebra. The determination of $\boldsymbol{P}_n$ and $\boldsymbol{Q}_n$ from $\boldsymbol{U}_n$, $\boldsymbol{L}_n$ and $\boldsymbol{\Pi}_n$ can again be done in time $O(i^2\, r^3)$. Consequently, the cost of applying the operators $\boldsymbol{P}$ and $\boldsymbol{Q}$ during the relaxed resolution of (18) at order n is bounded by $O((i^2\, r^3 + i\, r^2\, d)\, n)$. The cost of the evaluation of the remaining dag is bounded by $s_\Phi\, \mathsf{R}(n) + O(\mathsf{M}(i)\, s_\Phi\, n + i\, t_\Phi\, n)$, as in the algebraic case. □

5. A WORKED EXAMPLE

The prototype application of the techniques in this paper is the integration of differential-algebraic systems of equations. Let us consider the traditional example of a pendulum, whose equations are given by

$$\begin{aligned} \dot{x} &= u \\ \dot{y} &= v \\ \dot{u} &= \lambda\, x \\ \dot{v} &= \lambda\, y - g \\ x^2 + y^2 &= L^2. \end{aligned}$$

Here x, u, y, v, λ are the unknowns and g, L are constant parameters; see Figure 1. When presented in this way, the pendulum is a differential-algebraic system of index 3. Using differentiation and simplification, it can be rewritten as a system of index 1:

$$\begin{aligned} \dot{x} &= u \\ \dot{u} &= \lambda\, x \\ L^2 &= x^2 + y^2 \\ 0 &= u\, x + v\, y \\ 0 &= u^2 + v^2 - g\, y + L^2\, \lambda. \end{aligned}$$

Setting $F = (F^{[1]}, F^{[2]}, F^{[3]}, F^{[4]}, F^{[5]}) = (x, u, y, v, \lambda)$, the latter system corresponds the following system Φ using our notations:

$$\begin{aligned}
\Phi^{[1]} &= \theta\, x - z\, u \\
\Phi^{[2]} &= \theta u - z\, \lambda\, x \\
\Phi^{[3]} &= x^2 + y^2 - L^2 \\
\Phi^{[4]} &= u\, x + v\, y \\
\Phi^{[5]} &= u^2 + v^2 - g\, y + L^2\, \lambda.
\end{aligned}$$

The order one anticipators of these equations are given by

$$\begin{aligned}
(\Phi^{[1]})^{\langle 1\rangle} &= -z\, u \\
(\Phi^{[2]})^{\langle 1\rangle} &= -z\, \lambda\, x \\
(\Phi^{[3]})^{\langle 1\rangle} &= x_{1;}^2 + y_{1;}^2 \\
(\Phi^{[4]})^{\langle 1\rangle} &= u_{1;}\, x_{1;} + v_{1;}\, y_{1;} \\
(\Phi^{[5]})^{\langle 1\rangle} &= u_{1;}^2 + v_{1;}^2
\end{aligned}$$

and the Jacobian matrix of Φ is given by

$$\frac{\partial \Phi}{\partial F} = \begin{pmatrix}
\theta & -z & 0 & 0 & 0 \\
-z\,\lambda & \theta & 0 & 0 & -z\, x \\
2\, x & 0 & 2\, y & 0 & 0 \\
u & x & v & y & 0 \\
0 & 2\, u & -g & 2\, v & L^2
\end{pmatrix}$$

For the initial conditions, one typically takes $x_0 = L \sin\alpha$, $y_0 = L\cos\alpha$, $u_0 = v_0 = 0$, and $\lambda_0 = g\cos\alpha / L$. For these initial conditions, the formula for the Jacobian implies

$$\boldsymbol{M}_n = \begin{pmatrix}
n & 0 & 0 & 0 & 0 \\
0 & n & 0 & 0 & 0 \\
2\, L \sin\alpha & 0 & 2\, L\cos\alpha & 0 & 0 \\
0 & L\sin\alpha & 0 & L\cos\alpha & 0 \\
0 & 0 & -g & 0 & L^2
\end{pmatrix}.$$

For every $n \in \mathbb{N}$, this matrix admits the inverse

$$\boldsymbol{P}_n = \begin{pmatrix}
\frac{1}{n} & 0 & 0 & 0 & 0 \\
0 & \frac{1}{n} & 0 & 0 & 0 \\
-\frac{\tan\alpha}{n} & 0 & \frac{1}{2\, L\cos\alpha} & 0 & 0 \\
0 & -\frac{\tan\alpha}{n} & 0 & \frac{1}{L\cos\alpha} & 0 \\
-\frac{1}{L^2\, n} & 0 & \frac{g}{2\, L^3 \cos\alpha} & 0 & \frac{1}{L^2}
\end{pmatrix}.$$

Now consider the operator

$$\boldsymbol{P} = \begin{pmatrix}
\theta^{-1} & 0 & 0 & 0 & 0 \\
0 & \theta^{-1} & 0 & 0 & 0 \\
-\tan\alpha\, \theta^{-1} & 0 & \frac{1}{2\, L\cos\alpha} & 0 & 0 \\
0 & -\tan\alpha\, \theta^{-1} & 0 & \frac{1}{L\cos\alpha} & 0 \\
-\frac{1}{L^2}\, \theta^{-1} & 0 & \frac{g}{2\, L^3\cos\alpha} & 0 & \frac{1}{L^2}
\end{pmatrix}.$$

This operator is the coefficientwise operator with $(\boldsymbol{P}g)_n = \boldsymbol{P}_n\, g$ for all $g \in \mathbb{K}[[z]]^{5\times 1}$ and $n \in \mathbb{N}$. The desired recursive equation (18) for f is therefore given by

$$f \;=\; f_0 - \boldsymbol{P}\,\Phi^{\langle 1\rangle},$$

or, equivalently,

$$\begin{aligned}
x &= L\sin\alpha + \theta^{-1}(z\,u)\\
u &= \theta^{-1}(z\,\lambda\,x)\\
y &= L\cos\alpha - \tan\alpha\,\theta^{-1}(z\,u) - \tfrac{1}{2\,L\cos\alpha}\,(x_{1;}^2 + y_{1;}^2)\\
v &= -\tan\alpha\,\theta^{-1}(z\,\lambda\,x) - \tfrac{1}{L\cos\alpha}\,(u_{1;}\,x_{1;} + v_{1;}\,y_{1;})\\
\lambda &= \tfrac{g\cos\alpha}{L} - \tfrac{1}{L^2}\,\theta^{-1}(z\,u) - \tfrac{g}{2\,L^3\cos\alpha}\,(x_{1;}^2 + y_{1;}^2) - \tfrac{1}{L^2}\,(u_{1;}^2 + v_{1;}^2).
\end{aligned}$$

One may now use any traditional relaxed evaluation algorithm in order to compute the power series solution:

$$\begin{aligned}
x &= L\sin\alpha + \left(\frac{1}{2}\,g\sin\alpha\cos\alpha\right) z^2 + O(z^4)\\
u &= (g\sin\alpha\cos\alpha)\,z + \frac{g\sin\alpha}{6\,L}((1-3\sin^2\alpha)\,g - \sin\alpha\cos\alpha)\,z^3 + O(z^4)\\
y &= L\cos\alpha - \left(\frac{1}{2}\,g\sin^2\alpha\right) z^2 + O(z^4)\\
v &= -(g\sin^2\alpha)\,z + \frac{g\sin^2\alpha}{6\,L\cos\alpha}(\sin\alpha\cos\alpha - 3 + (3\sin^2\alpha - 1)\,g)\,z^3 + O(z^4)\\
\lambda &= \frac{g\cos\alpha}{L} - \frac{g\sin\alpha\cos\alpha}{2\,L^2} - \frac{g^2\sin^2\alpha}{L^2}\,z^2 + O(z^4).
\end{aligned}$$

This shows how our algorithm applies to the derived index 1 formulation of the equations of the pendulum. In fact, our algorithm can also be applied directly to the original system of index 3, which we indeed regard as a major advantage of the method. However, the resulting matrices $\boldsymbol{M}_n$ have size 25×15, which makes this variant less suitable for pedagogic purposes.

Bibliography

[BCO+07] A. Bostan, F. Chyzak, F. Ollivier, B. Salvy, É. Schost, and A. Sedoglavic. Fast computation of power series solutions of systems of differential equations. In *Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms*, pages 1012–1021. New Orleans, Louisiana, U.S.A., January 2007.

[BK78] R. P. Brent and H. T. Kung. Fast algorithms for manipulating formal power series. *Journal of the ACM*, 25:581–595, 1978.

[BL12] J. Berthomieu and R. Lebreton. Relaxed p-adic hensel lifting for algebraic systems. In J. van der Hoeven and M. van Hoeij, editors, *Proc. ISSAC '12*, pages 59–66. Grenoble, France, July 2012.

[CK91] D. G. Cantor and E. Kaltofen. On fast multiplication of polynomials over arbitrary algebras. *Acta Informatica*, 28:693–701, 1991.

[CT65] J. W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Math. Computat.*, 19:297–301, 1965.

[FS74] M. J. Fischer and L. J. Stockmeyer. Fast on-line integer multiplication. *Proc. 5th ACM Symposium on Theory of Computing*, 9:67–72, 1974.

[Hoe97] J. van der Hoeven. Lazy multiplication of formal power series. In W. W. Küchlin, editor, *Proc. ISSAC '97*, pages 17–20. Maui, Hawaii, July 1997.

[Hoe02] J. van der Hoeven. Relax, but don't be too lazy. *JSC*, 34:479–542, 2002.

[Hoe03] J. van der Hoeven. Relaxed multiplication using the middle product. In Manuel Bronstein, editor, *Proc. ISSAC '03*, pages 143–147. Philadelphia, USA, August 2003.

[Hoe07] J. van der Hoeven. New algorithms for relaxed multiplication. *JSC*, 42(8):792–802, 2007.

[Hoe09] J. van der Hoeven. Relaxed resolution of implicit equations. Technical Report, HAL, 2009. http://hal.archives-ouvertes.fr/hal-00441977.

[Hoe10] J. van der Hoeven. Newton's method and FFT trading. *JSC*, 45(8):857–878, 2010.

[Hoe11] J. van der Hoeven. From implicit to recursive equations. Technical Report, HAL, 2011. http://hal.archives-ouvertes.fr/hal-00583125.

[Hoe14] J. van der Hoeven. Faster relaxed multiplication. In *Proc. ISSAC '14*, pages 405–412. Kobe, Japan, July 2014.

[Leb15] R. Lebreton. Relaxed hensel lifting of triangular sets. *JSC*, 68(2):230–258, 2015.

[LS16] R. Lebreton and É. Schost. A simple and fast online power series multiplication and its analysis. *JSC*, 72:231–251, 2016.

[Sed01] A. Sedoglavic. *Méthodes seminumériques en algèbre différentielle ; applications à l'étude des propriétés structurelles de systèmes différentiels algébriques en automatique*. PhD thesis, École polytechnique, 2001.

[SS71] A. Schönhage and V. Strassen. Schnelle Multiplikation großer Zahlen. *Computing*, 7:281–292, 1971.