Shanks function transformations in a vector space

Shanks function transformations in a vector space

Claude Brezinski* Michela Redivo–Zaglia†

Friendly dedicated to Prof. Francesco Costabile on the occasion of his 70th birthday

Abstract

In this paper, we show how to construct various extensions of Shanks transformation for functions in a vector space. They are aimed at transforming a function tending slowly to its limit when the argument tends to infinity into another function with better convergence properties. Their expressions as ratio of determinants and recursive algorithms for their implementation are given. A simplified form of one of them is derived. It allows us to obtain a convergence result for an important class of functions. An application to integrable systems is discussed.

Keywords: function tranformation, Shanks transformation, confluent ε–algorithm, Lotka–Volterra equation.

1 Introduction

In numerical analysis and in applied mathematics, many methods make use of sequences. If a sequence is slowly converging, it can be useful to accelerate it. This can be done, in some cases, by modifying the process which constructs the sequence. However, if this process is a black box and if one has no access to it, another possibility is to transform the sequence, by a so-called *sequence transformation*, into another sequence converging faster to the same limit under some assumptions. For sequences of numbers, one of the most well–known such transformations is due to Shanks [12]. It can be recursively implemented by the scalar ε–algorithm of Wynn [13]. This algorithm was extended by Wynn to sequences of vectors [15]. But, since the algebraic theory underlying this *vector ε–algorithm* could not be easily derived from its rule, two other generalizations were proposed in [3]. The first one can be recursively implemented by two different *topological ε–algorithms* (TEA1 and TEA2), while the second one requires the use of the $S\beta$–algorithm [10] (see also [11]). Recently, the

*Université de Lille, CNRS, UMR 8524 - Laboratoire Paul Painlevé, F-59000 Lille, France, E–mail: Claude.Brezinski@univ-lille1.fr.

†Università degli Studi di Padova, Dipartimento di Matematica, Via Trieste 63, 35121–Padova, Italy. E–mail: Michela.RedivoZaglia@unipd.it.

rules of the TEA1 and TEA2 were greatly simplified, thus leading to the *simplified topological ε–algorithms* (STEA1 and STEA2) [7]. The corresponding software and applications can be found in [8].

Similarly, when a real (or complex) function of a real (or complex) variable is tending slowly to a limit with its argument, it can be transformed into a new function by a *function transformation*. Function transformations are usually obtained by replacing the divided differences appearing in a scalar sequence transformation by derivatives after letting a parameter tend to zero. This is why the adjective *confluent* was added to the name of the algorithms for implementing them. A Shanks (real or complex) function transformation was presented by Wynn together with the corresponding *confluent ε–algorithm* for its implementation [14]. Let us mention that generalizations of this transformation and of this algorithm to the confluent vector case were never proposed. Later, a topological Shanks function transformation for functions in a vector space, together with its confluent topological ε–algorithm, were proposed in [4].

The aim of this paper is to consider several topological Shanks function transformations in a vector space and to discuss the implementation of some of them by confluent recursive algorithms. A new and cheaper confluent topological ε–algorithm is presented. This new algorithm allows us to obtain a convergence theorem for an important class of functions. An application to integrable systems is discussed.

2 Topological Shanks function transformations

Let $t \longmapsto \mathbf{f}(t) \in E$ be a function depending on a parameter $t \in D \subseteq \mathbb{K}$ ($\mathbb{R}$ or $\mathbb{C}$), where E is a vector space of sufficiently differentiable functions on $\mathbb{K}$. In many practical situations, E is $\mathbb{R}^p$ or $\mathbb{R}^{p\times q}$. We want to construct, for a fixed value of k, a *function transformation* $\mathbf{f} \in E \longmapsto \mathbf{e}_k \in E$ such that, for all t, $\mathbf{e}_k(t) = \mathbf{S} \in E$ if the function $\mathbf{f}$ satisfies the linear differential equation of order k, called the *kernel* of the transformation,

$$a_0(\mathbf{f}(t) - \mathbf{S}) + a_1\mathbf{f}'(t) + \cdots + a_k\mathbf{f}^{(k)}(t) = \mathbf{0}, \quad \forall t \tag{1}$$

where $\mathbf{S} \in E$ is independent of t, and $a_0, \ldots, a_k \in \mathbb{K}$ are independent of k and t. Obviously, a_0 has to be different from zero and it does not restrict the generality to impose the *normalization condition* $a_0 = 1$. In the case of a complex or real function, such a transformation was proposed by Wynn [14] together with the corresponding recursive algorithm, the confluent ε–algorithm, for its implementation. We will now generalize this transformation and this algorithm to a vector space E.

In Section 2.1, we consider a kernel of the form (1) leading to two transformations, while, in Section 2.3, we will discuss another form of the kernel giving rise to two other transformations.

2.1 First topological Shanks function transformations

Obviously if the function $\mathbf{f} \in E$ satisfies a relation of the form (1) where the coefficients a_i are known, such a function transformation is simply defined by

$$\mathbf{e}_k(\mathbf{f}(t)) = \mathbf{f}(t) + a_1\mathbf{f}'(t) + \cdots + a_k\mathbf{f}^{(k)}(t), \quad \forall t, \tag{2}$$

since $a_0 = 1$, and it holds, $\forall t, \mathbf{e}_k(\mathbf{f}(t)) = \mathbf{S}$.

Let us now see how to compute the coefficients a_i when they are unknown. Differentiating (1), we have, for all t

$$a_0\mathbf{f}'(t) + \cdots + a_k\mathbf{f}^{(k+1)}(t) = \mathbf{0}.$$

As in the case of sequences, we have to transform this relation in E into a relation in $\mathbb{K}$. Taking the duality product with $\mathbf{y} \in E^*$ (the algebraic dual space of E, that is the space of linear functionals on E), we obtain

$$a_0\langle \mathbf{y}, \mathbf{f}'(t)\rangle + \cdots + a_k\langle \mathbf{y}, \mathbf{f}^{(k+1)}(t)\rangle = 0.$$

Then, several possibilities occur. Considering this relation, differentiating $k-1$ times the functions appearing in it, and taking the normalization condition into account leads to a system of $k+1$ equations in the unknowns $a_0, \ldots, a_k$ (case 1). Instead of differentiating several times the preceding relation, another way is to use k different linear functionals $\mathbf{y}_1, \ldots, \mathbf{y}_k \in E^*$ (case 2). Obviously, a mixture of these two strategies could also be considered but it leads to a more complicated transformation and it will not be considered further.

In both cases, adding the normalization condition, a system of $k+1$ equations in $k+1$ unknowns is obtained, and, by construction, the transformation (2) gives $\forall t, \mathbf{e}_k(\mathbf{f}(t)) = \mathbf{S}$.

Let us now apply the same procedure to a function which does not belong to the kernel of the transformation. Any of the preceding strategies for constructing the algebraic system giving the coefficients a_i can still be used. However, since its solution now depends on k it will be designated by $a_i^{(k)}$ for $i = 0, \ldots, k$. Then, in both cases, we define the *first topological Shanks function transformation* by

$$\mathbf{e}_k(\mathbf{f}(t)) = a_0^{(k)}\mathbf{f}(t) + a_1^{(k)}\mathbf{f}'(t) + \cdots + a_k^{(k)}\mathbf{f}^{(k)}(t), \tag{3}$$

with $a_0^{(k)} = 1$, and, by construction, it holds

Theorem 1
If, for all t,

$$\mathbf{f}(t) = \mathbf{S} - a_1\mathbf{f}'(t) - \cdots - a_k\mathbf{f}^k(t),$$

then, for all t,

$$\mathbf{e}_k(\mathbf{f}(t)) = \mathbf{S}.$$

In the first case, the coefficients $a_i^{(k)}$ are solution of the system

$$\left.\begin{array}{ccccccc} a_0^{(k)} & & & & & = & 1 \\ a_0^{(k)}\langle \mathbf{y}, \mathbf{f}'(t)\rangle & + & \cdots & + & a_k^{(k)}\langle \mathbf{y}, \mathbf{f}^{(k+1)}(t)\rangle & = & 0 \\ \vdots & & & & \vdots & & \\ a_0^{(k)}\langle \mathbf{y}, \mathbf{f}^{(k)}(t)\rangle & + & \cdots & + & a_k^{(k)}\langle \mathbf{y}, \mathbf{f}^{(2k)}(t)\rangle & = & 0, \end{array}\right\}, \tag{4}$$

and it holds from (3)

$$\mathbf{e}_k(\mathbf{f}(t)) = \frac{\begin{vmatrix} \mathbf{f}(t) & \cdots & \mathbf{f}^{(k)}(t) \\ \langle \mathbf{y}, \mathbf{f}'(t)\rangle & \cdots & \langle \mathbf{y}, \mathbf{f}^{(k+1)}(t)\rangle \\ \vdots & & \vdots \\ \langle \mathbf{y}, \mathbf{f}^{(k)}(t)\rangle & \cdots & \langle \mathbf{y}, \mathbf{f}^{(2k)}(t)\rangle \end{vmatrix}}{\begin{vmatrix} \langle \mathbf{y}, \mathbf{f}''(t)\rangle & \cdots & \langle \mathbf{y}, \mathbf{f}^{(k+1)}(t)\rangle \\ \vdots & & \vdots \\ \langle \mathbf{y}, \mathbf{f}^{(k+1)}(t)\rangle & \cdots & \langle \mathbf{y}, \mathbf{f}^{(2k)}(t)\rangle \end{vmatrix}}, \quad k = 0, 1, \ldots, \tag{5}$$

where the determinant in the numerator denotes the element of E obtained by developing it with respect to its first row by the classical rule for expanding a determinant.

In case 2, we have the new system (notice that the $a_i^{(k)}$ are different from the previous ones but we kept the same notation)

$$\left.\begin{array}{ccccccc} a_0^{(k)} & & & & & = & 1 \\ a_0^{(k)}\langle \mathbf{y}_1, \mathbf{f}'(t)\rangle & + & \cdots & + & a_k^{(k)}\langle \mathbf{y}_1, \mathbf{f}^{(k+1)}(t)\rangle & = & 0 \\ \vdots & & & & \vdots & & \\ a_0^{(k)}\langle \mathbf{y}_k, \mathbf{f}'(t)\rangle & + & \cdots & + & a_k^{(k)}\langle \mathbf{y}_k, \mathbf{f}^{(k+1)}(t)\rangle & = & 0, \end{array}\right\} \tag{6}$$

where $\mathbf{y}_1, \ldots, \mathbf{y}_k \in E^*$, and (3) leads to

$$\mathbf{e}_k(\mathbf{f}(t)) = \frac{\begin{vmatrix} \mathbf{f}(t) & \cdots & \mathbf{f}^{(k)}(t) \\ \langle \mathbf{y}_1, \mathbf{f}'(t)\rangle & \cdots & \langle \mathbf{y}_1, \mathbf{f}^{(k+1)}(t)\rangle \\ \vdots & & \vdots \\ \langle \mathbf{y}_k, \mathbf{f}'(t)\rangle & \cdots & \langle \mathbf{y}_k, \mathbf{f}^{(k+1)}(t)\rangle \end{vmatrix}}{\begin{vmatrix} \langle \mathbf{y}_1, \mathbf{f}''(t)\rangle & \cdots & \langle \mathbf{y}_1, \mathbf{f}^{(k+1)}(t)\rangle \\ \vdots & & \vdots \\ \langle \mathbf{y}_k, \mathbf{f}''(t)\rangle & \cdots & \langle \mathbf{y}_k, \mathbf{f}^{(k+1)}(t)\rangle \end{vmatrix}}, \quad k = 0, 1, \ldots \tag{7}$$

These transformations can be implemented by solving the linear system (4) or the system (6), and then using (3), but, in case 1, there also exist various recursive algorithms for this purpose as we will see now.

2.2 Implementation by recursive algorithms

Let f be a real function depending on a parameter t. The *scalar Shanks function transformation* is given by (3) and it can be recursively implemented by the *confluent ε–algorithm* of Wynn [14] whose rule is

$$\varepsilon_{k+1}(t) = \varepsilon_{k-1}(t) + 1/\varepsilon'_k(t), \tag{8}$$

with $\varepsilon_{-1}(t) = 0$ and $\varepsilon_0(t) = f(t)$. It holds $\varepsilon_{2k}(t) = e_k(f(t))$ and $\varepsilon_{2k+1}(t) = 1/e_k(f'(t))$.

Remark 1
Let us mention that an extension of this algorithm to the vector case was never proposed. However, in the case where E is an Hilbert space, one can imagine the following extension

$$\varepsilon_{k+1}(t) = \varepsilon_{k-1}(t) + \varepsilon'_k(t)/\langle \varepsilon'_k(t), \varepsilon'_k(t)\rangle.$$

Such an algorithm remains to be studied. In particular, can $\varepsilon_k(t) \in E$ be expressed as a ratio of determinants? This is an interesting challenge.

The first topological Shanks function transformation (5), that is corresponding to case 1, can be implemented by a recursive algorithm called the *confluent topological ε–algorithm* [4]. After presenting its rules, we will explain how to modify them by eliminating the intermediate linear functionals and replacing them by the scalar functions given by the confluent ε–algorithm of Wynn.

2.2.1 The confluent topological $\boldsymbol{\varepsilon}$–algorithm

The topological Shanks function transformation (5) can be implemented by the *confluent topological ε–algorithm* [4] whose rules are, for $k = 0, 1, \ldots$,

$$\left.\begin{aligned} \boldsymbol{\varepsilon}_{2k+1}(t) &= \boldsymbol{\varepsilon}_{2k-1}(t) + \frac{\mathbf{y}}{\langle \mathbf{y}, \boldsymbol{\varepsilon}'_{2k}(t)\rangle} \in E^* \\ \boldsymbol{\varepsilon}_{2k+2}(t) &= \boldsymbol{\varepsilon}_{2k}(t) + \frac{\boldsymbol{\varepsilon}'_{2k}(t)}{\langle \boldsymbol{\varepsilon}'_{2k+1}(t), \boldsymbol{\varepsilon}'_{2k}(t)\rangle} \in E, \end{aligned}\right\} \tag{9}$$

with $\boldsymbol{\varepsilon}_{-1}(t) = \mathbf{0} \in E^*$ and $\boldsymbol{\varepsilon}_0(t) = \mathbf{f}(t) \in E$, and it holds

$$\boldsymbol{\varepsilon}_{2k}(t) = \mathbf{e}_k(\mathbf{f}(t)) \quad \text{and} \quad \boldsymbol{\varepsilon}_{2k+1}(t) = \mathbf{y}/\langle \mathbf{y}, \mathbf{e}_k(\mathbf{f}'(t))\rangle. \tag{10}$$

As can be seen, only the functions $\boldsymbol{\varepsilon}_{2k}$ are of interest.

Important remark: in order to understand how the confluent ε–algorithms (scalar, topological and, below, simplified) presented in this paper, work, an important

remark has to be made about the differential operator. Remind that $\mathbf{f}$ depends on a parameter t. Then the duality products $\langle \mathbf{y}, \mathbf{f}^{(i)} \rangle$ also depend on this parameter t. Thus, when differentiating the elements of $\mathbb{K}$, E and E^* appearing in the rules of the confluent ε–algorithm of Wynn, or in those of the confluent topological ε–algorithm, or in those of the simplified confluent topological ε–algorithm which will be defined below, the expressions containing duality products have also to be differentiated with respect to t. Obviously, the implementation of these algorithms requires the use of a symbolic language.

The main difficulty with the recursive rules (9) is that they need to differentiate the $\boldsymbol{\varepsilon}_k$'s and, predominately, to use the $\boldsymbol{\varepsilon}_{2k+1}$'s which are elements of E^*. Thus, we will now derive a simpler algorithm for the implementation of the topological Shanks function transformation in case 1.

2.2.2 The simplified confluent topological ε–algorithm

Consider the scalar function $\widehat{f}(t) = \langle \mathbf{y}, \mathbf{f}(t) \rangle \in \mathbb{K}$, and apply Shanks function transformation and Wynn's confluent ε–algorithm (8) to it. We denote by $\widehat{e}_k(\widehat{f}(t))$ and $\widehat{\varepsilon}_k(t)$ their results. As seen above, it holds

$$\widehat{\varepsilon}_{2k}(t) = \widehat{e}_k(\widehat{f}(t)), \qquad \widehat{\varepsilon}_{2k+1}(t) = 1/\widehat{e}_k(\widehat{f}\,'(t)).$$

We will now show how to modify the confluent topological ε–algorithm (9) by combining it with the confluent ε–algorithm of Wynn (8), thus leading to a new algorithm, cheaper and easier to implement.

We have

$$\langle \mathbf{y}, \boldsymbol{\varepsilon}_{2k}(t) \rangle = \widehat{e}_k(\langle \mathbf{y}, \mathbf{f}(t) \rangle), \tag{11}$$

From what precedes, we also have

$$\boldsymbol{\varepsilon}_{2k+1}(t) = \mathbf{y}/\widehat{e}_k(\langle \mathbf{y}, \mathbf{f}\,'(t) \rangle) = \mathbf{y}\,\widehat{\varepsilon}_{2k+1}(t), \tag{12}$$

and it follows

$$\boldsymbol{\varepsilon}'_{2k+1}(t) = \mathbf{y}\,\widehat{\varepsilon}'_{2k+1}(t), \tag{13}$$

We finally obtain the *simplified confluent topological ε–algorithm* which, thanks to (8), can be written under one of the following mathematically equivalent forms

$$\begin{aligned}
\boldsymbol{\varepsilon}_{2k+2}(t) &= \boldsymbol{\varepsilon}_{2k}(t) + \frac{1}{\widehat{\varepsilon}'_{2k}(t)\widehat{\varepsilon}'_{2k+1}(t)}\boldsymbol{\varepsilon}'_{2k}(t), && (14)\\
&= \boldsymbol{\varepsilon}_{2k}(t) + \frac{\widehat{\varepsilon}_{2k+1}(t) - \widehat{\varepsilon}_{2k-1}(t)}{\widehat{\varepsilon}'_{2k+1}(t)}\boldsymbol{\varepsilon}'_{2k}(t), && (15)\\
&= \boldsymbol{\varepsilon}_{2k}(t) + \frac{\widehat{\varepsilon}_{2k+2}(t) - \widehat{\varepsilon}_{2k}(t)}{\widehat{\varepsilon}'_{2k}(t)}\boldsymbol{\varepsilon}'_{2k}(t), && (16)\\
&= \boldsymbol{\varepsilon}_{2k}(t) + (\widehat{\varepsilon}_{2k+1}(t) - \widehat{\varepsilon}_{2k-1}(t))(\widehat{\varepsilon}_{2k+2}(t) - \widehat{\varepsilon}_{2k}(t))\boldsymbol{\varepsilon}'_{2k}(t), && (17)
\end{aligned}$$

with $\widehat{\varepsilon}_0(t) = \widehat{f}(t) = \langle \mathbf{y}, \mathbf{f}(t) \rangle$ as the initialization of Wynn's confluent ε–algorithm.

As explained above, this algorithm requires to differentiate the $\widehat{\varepsilon}_k$'s computed from $\widehat{f}$ by the confluent ε–algorithm but it no longer involves elements of E^*.

2.2.3 Convergence results

Some convergence results for this topological Shanks function transformation were already given in [4]. However, due to the nonlinearity and the complexity of the rules of the confluent topological ε–algorithm, no other results were proved. Now, the simplified confluent topological ε–algorithm allows us to obtain a convergence result for a wide class of functions. Let us begin by the

Definition 1
A function $\widehat{f}$ is said to be totally monotonic if, $\forall k$ and $\forall t \in [0, +\infty)$, $(-1)^k \widehat{f}^{(k)}(t) \geq 0$.

Let us remind that, as proved by S.N. Bernstein [1], a function in $[0, +\infty)$ is totally monotonic if and only if there exists a finite positive Borel measure μ on $[0, +\infty)$ such that

$$\widehat{f}(t) = \int_0^\infty e^{-xt}\, d\mu(x), \quad t \in [0, \infty).$$

We have the following result [2]

Theorem 2
If the confluent ε–algorithm of Wynn is applied to a totally monotonic function $\widehat{f}$ converging to S when t goes to infinity then, $\forall k$, $\widehat{\varepsilon}_{2k}(t)$ also tends to S.

Let us now apply the simplified confluent topological ε–algorithm to a totally monotonic function $\mathbf{f} \in E$ that is such that, $\forall k$ and $t \in [0, +\infty)$, $(-1)^k \mathbf{f}^{(k)}(t) \geq \mathbf{0} \in E$. Assume that $\widehat{f}(t) = \langle \mathbf{y}, \mathbf{f}(t) \rangle$ is also totally monotonic. Since the elements $\boldsymbol{\varepsilon}_{2k} \in E$ and the scalar functions $\widehat{\varepsilon}_{2k}$ are computed by a linear combination with the same coefficients, it implies that they share the same convergence and acceleration properties. Thus we have the

Theorem 3
If the topological Shanks function transformation is applied to a totally monotonic function $\mathbf{f} \in E$ converging to $\mathbf{S} \in E$ when t tends to infinity and such that, $\forall k$ and $\forall t \in [0, +\infty)$, $(-1)^k \langle \mathbf{y}, \mathbf{f}^{(k)}(t) \rangle \geq 0$ then, $\forall k$,

$$\lim_{t \to \infty} \boldsymbol{\varepsilon}_{2k}(t) = \mathbf{S}.$$

2.3 Other topological Shanks function transformations

In this Section, keeping the same notations as above for the coefficients although they are differently defined, we will derive two other topological Shanks function transformations starting from the kernel

$$a_0(\mathbf{f}(t)-\mathbf{S})+a_1(\mathbf{f}'(t)-\mathbf{S})+\cdots+a_k(\mathbf{f}^{(k)}(t)-\mathbf{S})=\mathbf{0},\quad \forall t. \tag{18}$$

They differ from the topological Shanks function transformation given in [4]. They are again defined by (3) but, now, the coefficients are obtained as the solution of the system (case 3)

$$\left.\begin{array}{ccccccc} a_0^{(k)} & + & \cdots & + & a_k^{(k)} & = & 1 \\ a_0^{(k)}\langle \mathbf{y}, \mathbf{f}'(t)\rangle & + & \cdots & + & a_k^{(k)}\langle \mathbf{y}, \mathbf{f}^{(k+1)}(t)\rangle & = & 0 \\ \vdots & & & & \vdots & & \\ a_0^{(k)}\langle \mathbf{y}, \mathbf{f}^{(k)}(t)\rangle & + & \cdots & + & a_k^{(k)}\langle \mathbf{y}, \mathbf{f}^{(2k)}(t)\rangle & = & 0, \end{array}\right\}, \tag{19}$$

where $\mathbf{y}\in E^*$.

It holds from (3) and (19)

$$\mathbf{e}_k(\mathbf{f}(t))=\frac{\begin{vmatrix} \mathbf{f}(t) & \cdots & \mathbf{f}^{(k)}(t) \\ \langle \mathbf{y}, \mathbf{f}'(t)\rangle & \cdots & \langle \mathbf{y}, \mathbf{f}^{(k+1)}(t)\rangle \\ \vdots & & \vdots \\ \langle \mathbf{y}, \mathbf{f}^{(k)}(t)\rangle & \cdots & \langle \mathbf{y}, \mathbf{f}^{(2k)}(t)\rangle \end{vmatrix}}{\begin{vmatrix} 1 & \cdots & 1 \\ \langle \mathbf{y}, \mathbf{f}'(t)\rangle & \cdots & \langle \mathbf{y}, \mathbf{f}^{(k+1)}(t)\rangle \\ \vdots & & \vdots \\ \langle \mathbf{y}, \mathbf{f}^{(k)}(t)\rangle & \cdots & \langle \mathbf{y}, \mathbf{f}^{(2k)}(t)\rangle \end{vmatrix}},\quad k=0,1,\ldots \tag{20}$$

The coefficients $a_i^{(k)}$ can also be obtained from the system (case 4)

$$\left.\begin{array}{ccccccc} a_0^{(k)} & + & \cdots & + & a_k^{(k)} & = & 1 \\ a_0^{(n)}\langle \mathbf{y}_1, \mathbf{f}'(t)\rangle & + & \cdots & + & a_k^{(k)}\langle \mathbf{y}_1, \mathbf{f}^{(k+1)}(t)\rangle & = & 0 \\ \vdots & & & & \vdots & & \\ a_0^{(k)}\langle \mathbf{y}_k, \mathbf{f}'(t)\rangle & + & \cdots & + & a_k^{(k)}\langle \mathbf{y}_k, \mathbf{f}^{(k+1)}(t)\rangle & = & 0, \end{array}\right\} \tag{21}$$

and, in this case, (3) leads to a different transformation defined by

$$\mathbf{e}_k(\mathbf{f}(t)) = \frac{\begin{vmatrix} \mathbf{f}(t) & \cdots & \mathbf{f}^{(k)}(t) \\ \langle \mathbf{y}_1, \mathbf{f}'(t)\rangle & \cdots & \langle \mathbf{y}_1, \mathbf{f}^{(k+1)}(t)\rangle \\ \vdots & & \vdots \\ \langle \mathbf{y}_k, \mathbf{f}'(t)\rangle & \cdots & \langle \mathbf{y}_k, \mathbf{f}^{(k+1)}(t)\rangle \end{vmatrix}}{\begin{vmatrix} 1 & \cdots & 1 \\ \langle \mathbf{y}_1, \mathbf{f}'(t)\rangle & \cdots & \langle \mathbf{y}_1, \mathbf{f}^{(k+1)}(t)\rangle \\ \vdots & & \vdots \\ \langle \mathbf{y}_k, \mathbf{f}'(t)\rangle & \cdots & \langle \mathbf{y}_k, \mathbf{f}^{(k+1)}(t)\rangle \end{vmatrix}}, \quad k = 0, 1, \ldots \tag{22}$$

For these two topological Shanks function transformations, if there exist $\mathbf{S} \in E$ independent of t, and $a_0, \ldots, a_k \in \mathbb{K}$, independent of k and t such that (18) holds then $\forall t, \mathbf{e}_k(\mathbf{f}(t)) = \mathbf{S}$.

These new topological Shanks function transformations can be implemented by solving the linear system (19) or the system (21), and then using (3), but the H–algorithm [5,9] can also be used for this purpose. Indeed, if we set, for a fixed value of t, $\mathbf{S}_n = \mathbf{f}^{(n)}(t)$ and $g_i(n) = \langle \mathbf{y}, \mathbf{f}^{(n+i)}(t)\rangle$ for $n = 0, 1, \ldots$, and $i = 1, 2, \ldots$, then $\mathbf{H}_k^{(0)} = \mathbf{e}_k(\mathbf{f}(t))$ as given by (20). With $g_i(n) = \langle \mathbf{y}_i, \mathbf{f}^{(n)}(t)\rangle$, we obtain $\mathbf{H}_k^{(0)} = \mathbf{e}_k(\mathbf{f}(t))$ as given by (22).

Remark 2
Let us remind that, if $\mathbf{f}(t) = (f_1(t), \ldots, f_p(t))^T$ *is a vector function, then a confluent vector* ε*–algorithm can be defined from the vector* ε*–algorithm following the same lines as those followed by Wynn for obtaining his confluent* ε*–algorithm [14]. Obviously,* $\boldsymbol{\varepsilon}_{k+1}(t)$ *is a combination of the form* (3) *but involving* $\mathbf{f}(t), \mathbf{f}'(t), \ldots, \mathbf{f}^{(2k)}(t)$*. It is not known if* $\boldsymbol{\varepsilon}_k(t)$ *can be expressed as a ratio of determinants or designants. Moreover, convergence results have to be proved for it. This is another interesting challenge. See [6] for details.*

3 An application

We will now give an application to integrable systems and complete the results given in [6]. The variable t has been suppressed everywhere for simplicity.

We assume that E is a real inner product space. Thus, the duality product $\langle \cdot, \cdot \rangle$ becomes the usual inner product, and the results of [6] hold without any modification which does not seem to be possible for a general algebraic vector space.

Differentiating the first rule of the confluent topological ε–algorithm (9) gives

$$\langle \mathbf{y}, \boldsymbol{\varepsilon}'_{2k}\rangle^2 [\boldsymbol{\varepsilon}'_{2k-1} - \boldsymbol{\varepsilon}'_{2k+1}] = \langle \mathbf{y}, \boldsymbol{\varepsilon}''_{2k}\rangle \mathbf{y}.$$

Taking the inner product of this relation with $\boldsymbol{\varepsilon}'_{2k}$ and setting $M_k = \langle \mathbf{y}, \boldsymbol{\varepsilon}'_k \rangle$ and $P_k = \langle \boldsymbol{\varepsilon}'_k, \boldsymbol{\varepsilon}'_{k+1} \rangle$ yields

$$M'_{2k} = M_{2k}[P_{2k-1} - P_{2k}]. \tag{23}$$

Similarly, differentiating the second rule of the algorithm and taking the duality product with $\mathbf{y}$ gives

$$\langle \boldsymbol{\varepsilon}'_{2k+1}, \boldsymbol{\varepsilon}'_{2k} \rangle^2 [\boldsymbol{\varepsilon}'_{2k+2} - \boldsymbol{\varepsilon}'_{2k}] = \boldsymbol{\varepsilon}''_{2k} \langle \boldsymbol{\varepsilon}'_{2k+1}, \boldsymbol{\varepsilon}'_{2k} \rangle - \boldsymbol{\varepsilon}'_{2k} \langle \boldsymbol{\varepsilon}'_{2k+1}, \boldsymbol{\varepsilon}'_{2k} \rangle',$$

that is

$$P_{2k}^2 [M_{2k+2} - M_{2k}] = M'_{2k} P_{2k} - M_{2k} P'_{2k}.$$

Replacing M'_{2k} by its expression (23), produces

$$M_{2k} P'_{2k} = P_{2k}[M_{2k} P_{2k-1} - M_{2k+2} P_{2k}]. \tag{24}$$

Thus, (23)–(24) form a couple of Lotka-Volterra type equations. They are, in fact, incomplete since there is no relation for determining P_{2k-1}. We will now obtain it from the simplified confluent topological ε–algorithm. From (13), we have $\langle \mathbf{y}, \boldsymbol{\varepsilon}'_{2k+1} \rangle = \langle \mathbf{y}, \mathbf{y} \rangle \widehat{\varepsilon}'_{2k+1}$ and it follows

$$\boldsymbol{\varepsilon}'_{2k+1} = \mathbf{y} M_{2k+1} / \langle \mathbf{y}, \mathbf{y} \rangle.$$

Taking the inner product with $\boldsymbol{\varepsilon}'_{2k}$ and then with $\boldsymbol{\varepsilon}'_{2k+2}$, we obtain

$$\begin{array}{rrrrr} \langle \boldsymbol{\varepsilon}'_{2k}, \boldsymbol{\varepsilon}'_{2k+1} \rangle = P_{2k} & = \langle \mathbf{y}, \boldsymbol{\varepsilon}'_{2k} \rangle M_{2k+1} / \langle \mathbf{y}, \mathbf{y} \rangle & = M_{2k} M_{2k+1} / \langle \mathbf{y}, \mathbf{y} \rangle \\ \langle \boldsymbol{\varepsilon}'_{2k+2}, \boldsymbol{\varepsilon}'_{2k+1} \rangle = P_{2k+1} & = \langle \mathbf{y}, \boldsymbol{\varepsilon}'_{2k+2} \rangle M_{2k+1} / \langle \mathbf{y}, \mathbf{y} \rangle & = M_{2k+2} M_{2k+1} / \langle \mathbf{y}, \mathbf{y} \rangle. \end{array}$$

Thus it follows

$$P_{2k+1} = P_{2k} M_{2k+2} / M_{2k}$$

which is the missing relation in (23)–(24), and we finally obtain

$$\begin{array}{rcl} M_{2k-2} M'_{2k} & = & M_{2k}[M_{2k} P_{2k-2} - M_{2k-2} P_{2k}] \\ M_{2k-2} M_{2k} P'_{2k} & = & P_{2k}[M_{2k}^2 P_{2k-2} - M_{2k-2} M_{2k+2} P_{2k}]. \end{array}$$

See [6] for more results on this topic. Other applications have still to be found.

Acknowledgements: The work of C.B. was supported in part by the Labex CEMPI (ANR-11-LABX-0007-01), and, in part, by the University of Padua. The work of M.R.-Z. was partially supported by the University of Padua, Project 2014 No. CPDA143275.

References

[1] S. N. Bernstein, Sur les fonctions absolument monotones, Acta Math., 52 (1929) 1–66.

[2] C. Brezinski, Convergence d'une forme confluente de l'ε algorithme, C. R. Acad. Sci. Paris, 273 A (1971) 582–585.

[3] C. Brezinski, Généralisation de la transformation de Shanks, de la table de Padé et de l'ε–algorithme, Calcolo, 12 (1975) 317–360.

[4] C. Brezinski, Forme confluente de l'ε-algorithme topologique, Numer. Math., 23 (1975) 363–370.

[5] C. Brezinski, About Henrici's method for nonlinear equations, Symposium on Numerical Analysis and Computational Complex Analysis, Zürich, unpublished, August 1983.

[6] C. Brezinski, Cross rules and non-Abelian lattice equations for the discrete and confluent non–scalar ε–algorithms, J. Phys. A: Math. Theor., 43 (2010) 205201.

[7] C. Brezinski, M. Redivo–Zaglia, The simplified topological ε–algorithms for accelerating sequences in a vector space, SIAM J. Sci. Comput., 36 (2014) A2227–A2247.

[8] C. Brezinski, M. Redivo–Zaglia, The simplified topological ε–algorithms: software and applications, in preparation.

[9] C. Brezinski, H. Sadok, Vector sequence transformations and fixed point methods, in *Numerical Methods in Laminar and Turbulent Flows*, C. Taylor et al. eds., Pineridge Press, Swansea, 1987, pp. 3–11.

[10] K. Jbilou, Méthodes d'Extrapolation et de Projection. Applications aux Suites de Vecteurs, Thèse de 3e Cycle, Université de Lille I, 1988.

[11] K. Jbilou, H. Sadok, Some results about vector extrapolation methods and related fixed point iteration, J. Comp. Appl. Math., 36 (1991) 385–398.

[12] D. Shanks, Non linear transformations of divergent and slowly convergent sequences, J. Math. Phys., 34 (1955) 1–42.

[13] P. Wynn, On a device for computing the $e_m(S_n)$ transformation, MTAC, 10 (1956) 91–96.

[14] P. Wynn, Confluent forms of certain nonlinear algorithms, Arch. Math., 11 (1960) 223–234.

[15] P. Wynn, Acceleration techniques for iterated vector and matrix problems, Math. Comput., 16 (1962) 301–322.