A derivation of extrapolation algorithms based on error estimates

Claude Brezinski, Ana C. Matos
Laboratoire d'Analyse Numérique et d'Optimisation
UFR IEEA – M3, Université des Sciences et Technologies de Lille
59655 – Villeneuve d'Ascq cedex, France

Abstract: In this paper, we shall emphasize the role played by error estimates and annihilation difference operators in the construction of extrapolations processes. It is showed that this approach leads to a unified derivation of many extrapolation algorithms and related devices, to general results about their kernels and that it opens the way to many new algorithms. Their convergence and acceleration properties could also be studied within this framework.

Keywords: extrapolation methods, convergence acceleration.

Subject Classification: AMS(MOS): 65B05, 65B10.

1 Introduction

The first methods used for accelerating the convergence of sequences were the linear summation processes which go back to Euler, Cesaro, Hausdorff and others; see [42], for example. Among them, is also the Richardson extrapolation process based on polynomial extrapolation at zero [43] and which gave rise to Romberg's method for accelerating the convergence of the trapezoidal rule [44]. The first nonlinear convergence acceleration method was Aitken's Δ^2 process [1]. It was generalized by Shanks in 1955 [47] and, one year later, Wynn produced his ε–algorithm [56] for its implementation. Since then, many other extrapolation algorithms have been proposed and studied; see [20, 54] for a review and [16] for some history.

A quite general framework has been constructed along the years for the theory of extrapolation methods which, nowadays, lies on a firm basis; see [28, 54] and the first chapter of [20].

The situation is quite different for the practical construction of extrapolation methods, that is for the algorithms and, up to now, there was no systematic way for deriving them. Thus, in survey papers or books on the subject (see the bibliography at the end of the paper), each algorithm was usually presented separately from the others with no logical link between them.

It is our purpose, in this paper, to propose a systematic approach to extrapolation algorithms and their construction. In fact, such an approach was already used in [49] and in [50] for Levin's transforms, but its formalism is due to Weniger [51]. It is based on remainder (or error) estimates and annihilation operators. We think that this approach is a very interesting and powerful one and that it has not been sufficiently exploited. In this paper, we shall develop this point of view and show its impact on the subject. We shall not enter here into all the details (they will be treated in subsequent publications) but we only intend to open the way. As we shall see below, this approach leads to a better understanding of the mechanism of extrapolation algorithms, it gives us a framework where all the processes actually known can be included and it also provides us new algorithms and new theoretical results.

2 The scenery

Let us begin by some definitions.

Sequences will be denoted by letters without any index and their terms by the same letter (unless indicated) with a subscript. If $u = (u_n)$ and $v = (v_n)$, we shall make use of the notation $u/v = (u_n/v_n)$ and a similar notation for the product. A sequence will always start with the index 0.

Definition 1 [55, p.196]:
Let $\mathcal{S}$ be the set of complex sequences. A difference operator L is a linear mapping of $\mathcal{S}$ into itself

$$L : u = (u_n) \in \mathcal{S} \longmapsto L(u) = ((L(u))_n) \in \mathcal{S}.$$

Such an operator L is represented by an infinite matrix or, in other terms, by the sequence (L_n) of linear forms mapping u into the n–th term of the sequence $L(u)$. This is why, for briefness, the n–th member $(L(u))_n$ of the sequence $L(u)$ will be denoted by $L_n(u)$ or, sometimes, by $L(u_n)$. The notation $L = l$ (for example, $L = \Delta$) means that $\forall n, L_n(u) = l(u_n)$ (for example, $\forall n, L_n(u) = \Delta u_n$).

It is well known [3] that the most general difference operator is defined by

$$L_n(u) = \sum_{i=-p_n}^{q_n} G_i(n) u_{n+i}$$

where p_n and q_n are nonnegative integers which can eventually depend on n, $u_i = 0$ for $i < 0$, and the G_i's are given functions of n which can also depend on auxiliary fixed sequences (in theory, with some supplementary assumptions on the $G_i(n)$'s, q_n can be infinite. However, in practical situations, we shall only consider the finite case). It must be clearly understood that, if the auxiliary sequences on which the G_i's could depend, also depend on some terms of the sequence (u_n) itself, then these terms are *fixed* in the G_i's and thus the operator L is still a linear one. In other terms, for defining L, we must first choose the fixed sequence u which is used in the G_i's and then keep the same G_i's for all the sequences to which L is applied. Thus we have $L(u + v) = L(u) + L(v)$ since, after choosing the sequence u which enters into the definition of the auxiliary sequences G_i, these G_i's are kept fixed. Such an assumption is usual for extrapolation algorithms; see [17].

This remark is very important since, in fact, it allows us to use the Toeplitz theorem for proving the convergence of the transformed sequence to the same limit as the initial sequence. However, when the $G_i(n)$ depend on the sequence to be transformed, the convergence is ensured only for the sequence under consideration and not for *all* converging sequences as is the case for linear summation processes when the $G_i(n)$'s do not depend on the initial sequence. This point will be discussed in more details in the sequel.

Definition 2 [51, p.212]:
L is called an annihilation difference operator for the sequence $a = (a_n)$ *if* $\exists N$ *such that* $\forall n \geq N, L_n(a) = 0$.

We can assume, without any loss of generality, that $N = 0$.

Definition 3 [51, p.212]:
The sequence (D_n) *is called a remainder (or error) estimate of the sequence* (S_n) *if* $\forall n, S_\infty - S_n = a_n D_n$ *where* (a_n) *is an unknown sequence and* S_∞ *a (usually unknown) number. If* (S_n) *converges to* S_∞*, then* S_∞ *is called its limit and, otherwise, its antilimit.*

As explained in [20], the first step in the construction of an extrapolation method is to assume that the sequence under consideration has a certain behavior. In other terms, one should construct an algorithm able to find the exact limit (or antilimit) S_∞ of certain sequences. This will be achieved by using an annihilation difference operator, as in [51], but with a slightly generalized approach.

We assume that the sequence $S = (S_n)$ satisfies $\forall n$,

$$S_\infty - S_n = a_n D_n$$

where $a = (a_n)$ is an unknown sequence and $D = (D_n)$ a known one. We want to construct a sequence transformation $T : (S_n) \longmapsto (T_n)$ such that $\exists N, \forall n \geq N, T_n = S_\infty$.
We assume also that $\exists b = (b_n)$ such that $L(b)$ is known and L is an annihilation difference operator for $a - b$. Thus, we have

$$\frac{S_\infty}{D_n} - \frac{S_n}{D_n} = a_n$$

and

$$\frac{S_\infty}{D_n} - \frac{S_n}{D_n} - b_n = a_n - b_n.$$

Applying L to both sides of this relation and using its linearity property, we obtain $\forall n \geq N$

$$S_\infty L_n(1/D) - L_n(S/D) - L_n(b) = L_n(a - b) = 0$$

and it follows

$$S_\infty = \frac{L_n(S/D) + L_n(b)}{L_n(1/D)}.$$

Thus, if we define the transformation $T : (S_n) \longmapsto (T_n)$ by

$$T_n = \frac{L_n(S/D) + L_n(b)}{L_n(1/D)}$$

then, by construction, $\forall n \geq N, T_n = S_\infty$ if and only if $\forall n \geq N, S_\infty - S_n = a_n D_n$. We recall the

Definition 4 [20, p.4]:
Let $T : (S_n) \longmapsto (T_n)$ be a sequence transformation. The kernel $\mathcal{K}_T$ of T is the set of sequences such that $\exists S_\infty, \exists N, \forall n \geq N, T_n = S_\infty$.

Remark: if a sequence $\alpha = (\alpha_n)$ and an operator L are known such that $L(\alpha a - b) = 0$, then we can write

$$S_\infty - S_n = (\alpha_n)a_n(D_n/\alpha_n)$$

and thus

$$S_\infty = \frac{L_n(\alpha S/D) + L_n(b)}{L_n(\alpha/D)}.$$

This remark will be useful later.

The simplest form of the transformation T is obtained when $b = (0)$. Unless specified (see sections 3.1, 3.9, 3.14 and 3.15), we shall only consider this case.

Obviously, if the sequence (D_n) is invariant under translation of the sequence (S_n) (that is, if it remains the same when a constant is added to all the terms of (S_n)), then the transformation T is translative (that is, the same constant is added to all the terms of (T_n)). As proved in [14], a necessary and sufficient condition for that property to hold is that T_n could be written as

$$T_n = F_n(S_0, \ldots, S_{k(n)})$$

with

$$F_n(x_0, \ldots, x_{k(n)}) = \frac{f_n(x_0, \ldots, x_{k(n)})}{Df_n(x_0, \ldots, x_{k(n)})}$$

and $D^2 f_n$ identically zero, where Df_n denotes the sum of the partial derivatives of f_n with respect to $x_0, \ldots, x_{k(n)}$. The remark that all the translative sequence transformations can be written under this form was originally made by Benchiboun [4].
The transformation T, as obtained above, is also homogeneous which means that if, when all the terms of (S_n) are multiplied by a non–zero constant c, the D_n's become cD_n, then the T_n's are also multiplied by the same constant c. A transformation which is translative and homogeneous is called quasi–linear.

The vector, matrix and confluent cases can also be included into this framework.
If S_∞, S_n and a_n are vectors and D_n numbers, then we have

$$S_\infty L_n(1/D) - L_n(S/D) = L_n(a).$$

where L_n is applied componentwise.

If, now, S_∞, S_n, a_n and D_n are $p \times p$ complex matrices and $\mathcal{S}$ is the set of sequences of complex $p \times p$ matrices, then two cases have to be considered, the right case and the left one as for Padé approximants for matrix series [2, vol.2, pp.50ff].
We have to assume that $L(\alpha u_n) = \alpha L(u_n)$ in the right case and that $L(u_n \alpha) = L(u_n)\alpha$ in the left case where α is an arbitrary matrix.
i) Right case
We write

$$S_\infty - S_n = a_n D_n \quad \text{that is} \quad S_\infty D_n^{-1} - S_n D_n^{-1} = a_n.$$

Thus

$$S_\infty L(D_n^{-1}) - L(S_n D_n^{-1}) = L(a_n)$$

and the transformation T is defined by

$$T_n = L(S_n D_n^{-1}) \left[L(D_n^{-1}) \right]^{-1}.$$

ii) Left case
We write

$$S_\infty - S_n = D_n a_n \quad \text{that is} \quad D_n^{-1} S_\infty - D_n^{-1} S_n = a_n.$$

Thus

$$L(D_n^{-1}) S_\infty - L(D_n^{-1} S_n) = L(a_n)$$

and the transformation T is defined by

$$T_n = \left[L(D_n^{-1}) \right]^{-1} L(D_n^{-1} S_n).$$

These questions are discussed in detail in [24].

Let us now consider the confluent case where S, a and D are functions of a variable t. We have

$$S_\infty - S(t) = a(t) D(t).$$

If L is a linear operator on the set of functions, then

$$S_\infty L(1/D(t)) - L(S(t)/D(t)) = L(a(t))$$

and the transformation T is given by

$$T(t) = L(S(t)/D(t))/L(1/D(t)).$$

Obviously, it is also possible to consider the vector and matrix confluent cases.

3 Some extrapolation algorithms

Let us now consider different choices of the linear operator L and some particular choices of the error estimates (D_n). We shall see that they lead to different well–known transformations and we shall also propose some natural generalizations.

The examples 1 to 8 concern a linear operator which is independent of the sequence (S_n). In the examples 9 to 11, L depends on (S_n). The case where L is a linear combination of several other operators is considered in the examples 12 and 13. Finally, generalizations with a correction factor are treated in the examples 14 and 15.

3.1 A simple transformation

The simplest difference operator is the identity. In this case, the most general form of the transformation T leads to

$$T_n = S_n + b_n D_n.$$

The kernel of this transformation is the set of sequences of the form $S_\infty - S_n = b_n D_n$. Such a transformation was considered in [21] where some theoretical results about it can be found.

3.2 The Θ–procedure

If $\forall n, a_n = a, b_n = 0$, and if $L = \Delta$, the forward difference operator, then we have

$$T_n = \frac{D_{n+1}S_n - D_n S_{n+1}}{D_{n+1} - D_n} = S_n - \frac{\Delta S_n}{\Delta D_n} D_n.$$

This transformation was considered in [10]. It is called the Θ–procedure.

When $D_n = x_n - x$, where (x_n) is a known sequence converging to a known limit x, then the second standard process of Germain–Bonne [30] is recovered. For some particular choices of the sequence (x_n), new acceleration processes based on convergence tests for sequences and error estimates can be constructed as introduced in [15] and developed in [34].

3.3 Summation processes

If $L_n(u) = \sum_{i=0}^{k_n} G_i(n) u_i$ where the $G_i(n)$'s are given numbers, if $L(b) = 0$, and if $\forall n, D_n = 1$, then T is a linear summation process as defined, for example, in [54]. The convergence of the sequence (T_n) to S_∞ for all sequences (S_n) converging to S_∞ is given by the Toeplitz theorem. Their acceleration properties were studied in [53].

3.4 Column and diagonal transformations

If $b_n = 0, D_n = \Delta S_n$ and $L = \Delta^k$, then for $k = 1$ we obtain Aitken's Δ^2 process, while $k = 2$ corresponds to the second column of the θ–algorithm [5]. The case of an arbitrary value of k was considered by Drummond [29].

Because of this example, let us discuss the case where the linear forms defining L also depend on a second integer k. We shall denote them by $L_{n,k}$ and consider, as before, the difference operator L defined by the sequence $(L_n = L_{n,k})_n$ for a fixed value of k or the operator obtained by reversing the roles of n and k, that is the operator L' defined by the sequence $(L'_k = L_{n,k})_k$ for a fixed value of n. We shall see, in the sequel, many other examples of such linear forms.

Thus, for the above operator, we obtain two alternatives

i) k fixed and n varying. In this case we shall consider the sequence $L = \Delta^k$ as above (that is $L_n(u) = \Delta^k u_n$). We obtain, for $n = 0, 1, \ldots$

$$T_n = \Delta^k(S_n/\Delta S_n)/\Delta^k(1/\Delta S_n)$$

and we shall speak about a column transformation,
ii) n fixed and k varying. In that case we shall consider the sequence $L'_k = \Delta^k$ (that is $L'_k(u) = \Delta^k u_n$) which gives, for $k = 0, 1, \ldots$

$$T'_k = \Delta^k(S_n/\Delta S_n)/\Delta^k(1/\Delta S_n)$$

and we shall speak about a diagonal transformation.
The reasons for these names are clearly understood if we set

$$T_k^{(n)} = \Delta^k(S_n/\Delta S_n)/\Delta^k(1/\Delta S_n)$$

and if we display these quantities into a two–dimensional table as follows

$$\begin{array}{lllll}
T_0^{(0)} & & & & \\
T_0^{(1)} & T_1^{(0)} & & & \\
T_0^{(2)} & T_1^{(1)} & T_2^{(0)} & & \\
T_0^{(3)} & T_1^{(2)} & T_2^{(1)} & T_3^{(0)} & \\
\vdots & \vdots & \vdots & \vdots & \ddots
\end{array}$$

So, n indicates the minimal index of the sequence S which is used in the computation of $T_k^{(n)}$ while the index k is related to the number of terms of S needed and, thus, it is, in some sense, a measure of the complexity of the sequence transformation.

Obviously, it is also possible to define other difference operators by letting n and k vary arbitrarily.

3.5 Levin's transforms

If $b_n = 0$ and if a_n is a polynomial of degree $k-1$ in $n+1$, then it is well known that $L = \Delta^k$ is an annihilation difference operator for (a_n). This remark is the basis used by Levin [33] for constructing his various sequence transformations based on different choices of the sequence (D_n) and by Weniger [51] for extending them. Levin's transforms can be generalized by taking, for a_n, a polynomial of degree $k-1$ in x_n. In that case, the annihilation difference operator is the divided difference operator δ_k that will be defined in the next subsection and Levin's transformations appear as a generalization of Richardson's. The generalization of Richardson process introduced by Sidi [48] can also be put into this framework and Drummond's method [29] as well. The connections between these processes are discussed in more detail in [51, pp.236–238] and [20, pp.116–119]. Levin's transforms will be discussed again in sections 4.1 and 4.2.

3.6 Richardson extrapolation

Let us now consider the well–known Richardson extrapolation procedure [43]. It consists in assuming that

$$S_n = S_\infty - c_1 x_n - \cdots - c_k x_n^k$$

where (x_n) is a given auxiliary sequence. Thus

$$S_\infty - S_n = (c_1 + \cdots + c_k x_n^{k-1}) x_n$$

which corresponds to $D_n = x_n$ and $a_n = c_1 + \cdots + c_k x_n^{k-1}$. The divided difference operator δ_k of order k at the points x_i is an annihilation operator for the sequence (a_n). Let us recall that this operator is recursively defined by

$$\delta_{k+1}(u_n) = \frac{\delta_k(u_{n+1}) - \delta_k(u_n)}{x_{n+k+1} - x_n}$$

with $\delta_0(u_n) = u_n$. Thus, we are in the case, described in the example 3.4, of an operator depending on a second index k. Using the same notations as above, we can construct a two–dimensional array by

$$T_k^{(n)} = \frac{\delta_k(S_n/x_n)}{\delta_k(1/x_n)}$$

with $T_0^{(n)} = S_n$. By construction, if $S_\infty - S_n$ has the form above, then $\forall n, T_k^{(n)} = S_\infty$.

We shall now prove that these numbers are identical to the numbers $U_k^{(n)}$ constructed by the Richardson extrapolation scheme

$$U_k^{(n)} = \frac{x_{n+k} U_{k-1}^{(n)} - x_n U_{k-1}^{(n+1)}}{x_{n+k} - x_n}$$

with $U_0^{(n)} = S_n$.
By using the recurrence relation between divided differences, it is easy to prove by induction that

$$\delta_k(1/x_n) = (-1)^k / (x_n \cdots x_{n+k}).$$

We can also prove by induction that

$$\delta_k(S_n/x_n) = \frac{(-1)^k}{x_n \cdots x_{n+k}} U_k^{(n)}$$

where $U_k^{(n)}$ is the quantity computed by the Richardson scheme. The property is true for $k = 0$. Assuming that it is true for k, we have

$$\delta_{k+1}(S_n/x_n) = (-1)^k \frac{x_n \cdots x_{n+k} U_k^{(n+1)} - x_{n+1} \cdots x_{n+k+1} U_k^{(n)}}{x_n x_{n+1}^2 \cdots x_{n+k}^2 x_{n+k+1} (x_{n+k+1} - x_n)}.$$

Thus, using the Richardson scheme, we obtain $\delta_{k+1}(S_n/x_n) = \delta_{k+1}(1/x_n) U_{k+1}^{(n)}$ which shows that $T_k^{(n)} = U_k^{(n)}$. A similar derivation can be found in [51, pp.246–247]. The expression for the divided differences of $1/x_n$ is also given [39, p.8].

Thus, the Richardson extrapolation method can also be implemented via the recurrence relations for divided differences as already mentioned in [48]. However, the Richardson scheme is simpler because it is a relation between the $T_k^{(n)}$'s themselves instead of two separate recurrence relations for their numerators and their denominators. If the Richardson extrapolation method has to be applied simultaneously to several sequences with the same auxiliary sequence (x_n) then, since the denominators are the same, it could be preferable to use the preceding scheme based on divided differences.

Using the usual determinantal formula for divided differences and the definition of $T_k^{(n)}$ given above, we recover the expression of these quantities as a ratio of two determinants, see [20, p.72]. It will probably be possible to derive new extrapolation schemes by using the recurrence relationship for generalized divided differences obtained by Mühlbach [40].

3.7 Overholt's process

The Overholt process [41] is defined by

$$V_{k+1}^{(n)} = \frac{\Delta(V_k^{(n)}/(\Delta S_{n+k})^{k+1})}{\Delta(1/(\Delta S_{n+k})^{k+1})}$$

with $V_0^{(n)} = S_n$.
For this algorithm, it is known that the quantities $V_k^{(n)}$ can be expressed as a ratio of determinants but these determinants have not yet been found [26].

From the above expression, we immediately see that $\forall n, V_{k+1}^{(n)} = S_\infty$ if and only if there exists a constant c_k such that

$$S_\infty - V_k^{(n)} = c_k(\Delta S_{n+k})^{k+1}.$$

Thus, assuming that the constant c_k is replaced by a polynomial of degree $m_k - 1$ in n, we obtain a generalization of Overholt's process

$$V_{k+1}^{(n)} = \frac{\Delta^{m_k}(V_k^{(n)}/(\Delta S_{n+k})^{k+1})}{\Delta^{m_k}(1/(\Delta S_{n+k})^{k+1})}.$$

If c_k is replaced by a polynomial in x_n, then divided differences have to be used instead of the operator Δ.

3.8 The E–algorithm

Let us now consider some more difficult examples where, in particular, the sequences G_i in the definition (given above) of the most general difference operator, can depend on some terms of the sequence (S_n).

Let us take

$$L_n(u) = L_k'(u) = \begin{vmatrix} u_n & u_{n+1} & \cdots & u_{n+k} \\ g_1(n) & g_1(n+1) & \cdots & g_1(n+k) \\ \vdots & \vdots & & \vdots \\ g_k(n) & g_k(n+1) & \cdots & g_k(n+k) \end{vmatrix}$$

where the $(g_i(n))$ are given auxiliary sequences which can depend on (S_n) itself and define T as above.

If we set

$$M_n(u) = \begin{vmatrix} u_n & u_{n+1} & \cdots & u_{n+k} \\ g_1(n)D_n & g_1(n+1)D_{n+1} & \cdots & g_1(n+k)D_{n+k} \\ \vdots & \vdots & \ddots & \vdots \\ g_k(n)D_n & g_k(n+1)D_{n+1} & \cdots & g_k(n+k)D_{n+k} \end{vmatrix},$$

then we also have

$$T_n = \frac{L_n(S/D)}{L_n(1/D)} = \frac{M_n(S)}{M_n(1)}$$

and a similar property for L'_k. For more properties of this type, see [17].

If $\forall n, D_n = 1$ and $S_n = x^n$, then the formal biorthogonal polynomials defined in [19] are recovered.

If $D_n = 1$, if $b_n = \sum_{i=1}^{k} c_i g_i(n)$ where the c_i's are arbitrary numbers and if the operator L is defined by the sequence (L_n) of linear forms given above, then the transformation T corresponds to the k–th column of the E–algorithm [9, 31] with the $(g_i(n))$'s as auxiliary sequences. If L' is defined by the sequence of linear forms (L'_k), then T corresponds to the n–th diagonal of the E–algorithm. Another approach to the E–algorithm and some related algorithms can be found in [23]. It is also related to annihilation difference operators, but does not make use of determinants in their definition.

The E–algorithm is the most general existing extrapolation algorithm. Almost all the sequence transformations actually known are particular cases of the E–algorithm and they can be recovered by different choices of the auxiliary sequences g_i. We shall now study one of them in more details.

Let us take

$$L_n(u) = L'_k(u) = \begin{vmatrix} u_n & u_{n+1} & \cdots & u_{n+k} \\ \Delta S_n & \Delta S_{n+1} & \cdots & \Delta S_{n+k} \\ \vdots & \vdots & & \vdots \\ \Delta S_{n+k-1} & \Delta S_{n+k} & \cdots & \Delta S_{n+2k-1} \end{vmatrix}.$$

If $D_n = 1$, if $b_n = \sum_{i=1}^{k} c_i \Delta S_{n+i-1}$ where the c_i's are arbitrary numbers and if the operator L is defined by the sequence (L_n), then T is the k–th column of the Shanks transformation [47], or, in other words, $2k$–th column of the ε–algorithm of Wynn [56]. If L' is defined by the sequence (L'_k), then T corresponds to the n–th diagonal of the Shanks transformation or the ε–algorithm. The first and second generalizations of the ε–algorithm [6] can also be put into this framework since they correspond to replacing the operator Δ by a more general one [7, 45].

3.9 Padé and Padé–type approximants

Let us consider, in the ε–algorithm, the particular case $S = (S_n)$ where $S_n = c_0 + c_1 z + \cdots + c_n z^n$ (that is the n–th partial sum of the series $S_\infty = f(z) = c_0 + c_1 z + c_2 z^2 + \cdots$). Since we have

$S_\infty - S_n = c_{n+1}z^{n+1} + \cdots$, then it corresponds to $D_n = 1$ and we shall take

$$L_p(u) = L'_q(u) = \begin{vmatrix} z^q u_{p-q} & z^{q-1}u_{p-q+1} & \cdots & u_p \\ c_{p-q+1} & c_{p-q+2} & \cdots & c_{p+1} \\ \vdots & \vdots & & \vdots \\ c_p & c_{p+1} & \cdots & c_{p+q} \end{vmatrix}$$

with the convention that $c_i = 0$ for $i < 0$.
If $b_p = \sum_{i=1}^{q} c_{p+i}z^{p+i}$, and if the operator L is defined by the sequence (L_p), then T_p is the $[p/q]$ Padé approximant of the series f at the point z [25]. We obtain

$$f(z)L(1) - L(S) = L((c_{p+1}z^{p+1} + \cdots)_p).$$

Using the definition of L, we easily see that

$$L((c_{p+1}z^{p+1} + \cdots)_p) = \begin{vmatrix} z^q(c_{p-q+1}z^{p-q+1} + \cdots) & \cdots & c_{p+1}z^{p+1} + \cdots \\ c_{p-q+1} & \cdots & c_{p+1} \\ \vdots & & \vdots \\ c_p & \cdots & c_{p+q} \end{vmatrix} = (z^q O(z^{p+1}))$$

since the terms in $z^{p+1}, \ldots, z^{p+q}$ in the first row disappear by linear combination with the other rows. This is the usual approximation–through–order property of the Padé approximants and we have, for a fixed value of q

$$L_p(S)/L_p(1) = [p/q]_f(z).$$

Of course, the same property holds if p is fixed and q changes.

If we set

$$L_n(u) = a_0^{(k)}z^k u_n + a_1^{(k)}z^{k-1}u_{n+1} + \cdots + a_k^{(k)}u_{n+k}$$

then, for $b_n = 0$ and $D_n = 1$, it is easy to see that

$$f(z)L_n(1) - L_n(S) = O(z^{n+k+1})$$

which shows that $L_n(S)/L_n(1)$ is the $(n+k/k)$ Padé–type approximant of f with $v_k(z) = a_0^{(k)} + \cdots + a_k^{(k)}z^k$ as its generating polynomial [8] since $L_n(1)$ is a polynomial of degree k and $L_n(S)$ is a polynomial of degree $n+k$.

3.10 θ–algorithm

The θ–algorithm [5] can be written as

$$\theta_{2k+2}^{(n)} = \frac{\Delta(\theta_{2k}^{(n+1)}/D_{2k+1}^{(n)})}{\Delta(1/D_{2k+1}^{(n)})}$$

with

$$\theta_{2k+1}^{(n)} = \theta_{2k-1}^{(n+1)} + D_{2k}^{(n)}$$

$$D_k^{(n)} = 1/(\theta_k^{(n+1)} - \theta_k^{(n)})$$

and $\theta_{-1}^{(n)} = 0, \theta_0^{(n)} = S_n$.

Thus, the θ–algorithm can also be put into our framework since it corresponds to an iterative use of our basic procedure with $S = \left(\theta_{2k}^{(n+1)}\right), D = \left(D_{2k+1}^{(n)}\right)$ and $b = (0)$. From theoretical reasons, it is known [26] that the quantities $\theta_{2k}^{(n)}$ can be expressed as a ratio of two determinants but these determinants have not yet been found.

From the above expression, we immediately see that $\forall n, \theta_{2k+2}^{(n)} = S_\infty$ if and only if there exists a constant c_k such that

$$S_\infty - \theta_{2k}^{(n+1)} = c_k D_{2k+1}^{(n)} = c_k / \Delta\theta_{2k+1}^{(n)}.$$

Thus, assuming that the constant c_k is replaced by a polynomial of degree $m_k - 1$ in n, we obtain a generalization of the θ–algorithm

$$\theta_{2k+2}^{(n)} = \frac{\Delta^{m_k}(\theta_{2k}^{(n+1)}/D_{2k+1}^{(n)})}{\Delta^{m_k}(1/D_{2k+1}^{(n)})}.$$

If c_k is replaced by a polynomial in x_n, then divided differences have to be used instead of the operator Δ.

3.11 ρ–algorithm

Extrapolation at infinity by a rational function consists in assuming that

$$S_n = \frac{S_\infty x_n^k + c_1 x_n^{k-1} + \cdots + c_k}{x_n^k + b_1 x_n^{k-1} + \cdots + b_k}.$$

It leads to a slight generalization the ρ–algorithm of Wynn [57] whose quantities are defined by a ratio of two determinants as follows (we only indicate their first rows)

$$\rho_{2k}^{(n)} = \frac{\begin{vmatrix} 1 & S_n & x_n & x_n S_n & \cdots & x_n^{k-1} & x_n^{k-1} S_n & x_n^k S_n \end{vmatrix}}{\begin{vmatrix} 1 & S_n & x_n & x_n S_n & \cdots & x_n^{k-1} & x_n^{k-1} S_n & x_n^k \end{vmatrix}}.$$

It corresponds to $D_n = 1$ and $L_n(u) = \begin{vmatrix} 1 & S_n & x_n & x_n S_n & \cdots & x_n^{k-1} & x_n^{k-1} S_n & x_n^k u_n \end{vmatrix}$.

These ratios of determinants can be recursively computed by the ρ–algorithm whose rules are

$$\rho_{k+1}^{(n)} = \rho_{k-1}^{(n+1)} + (x_{n+k+1} - x_n)/(\rho_k^{(n+1)} - \rho_k^{(n)})$$

with $\rho_{-1}^{(n)} = 0$ and $\rho_0^{(n)} = S_n$. These quantities are the usual reciprocal differences which play, in rational interpolation and in continued fractions, the role of divided differences in polynomial interpolation.

3.12 Composite transformations

Let $L^{(1)}, L^{(2)}, \ldots, L^{(k)}$ be difference operators and $(b_i^{(n)})$ given sequences of numbers. We shall define the difference operator L by

$$L_n = \sum_{i=1}^{k} b_i^{(n)} L_n^{(i)}$$

and the transformation T as usual by $T_n = L_n(S/D)/L_n(1/D)$.
Defining the transformation $T^{(i)}: \quad (S_n) \longmapsto (T_i^{(n)})$ by

$$T_i^{(n)} = L_n^{(i)}(S/D)/L_n^{(i)}(1/D),$$

we have

$$T_n = \sum_{i=1}^{k} c_i^{(n)} T_i^{(n)}$$

with

$$c_i^{(n)} = b_i^{(n)} \frac{L_n^{(i)}(1/D)}{L_n(1/D)}.$$

Thus

$$\sum_{i=1}^{k} c_i^{(n)} = 1$$

and we recover the composite sequence transformations introduced in [13].

We want to choose the $b_i^{(n)}$'s so that the kernel of the transformation T contains the kernels of the transformations $T^{(i)}$. So, let us assume that $\exists i, L^{(i)}(a) = 0$. In order that, $\forall n, T_n = S_\infty$, we must have $b_i^{(n)} \neq 0$ and $\forall j \neq i, b_j^{(n)} = 0$. Thus, $c_i^{(n)} = 1$ and $\forall j \neq i, c_j^{(n)} = 0$. Since, in that case, $T_i^{(n)} = S_\infty$ and $\Delta T_i^{(n)} = 0$, a possible choice for L, when $\forall n, D_n = 1$, is given by

$$L_n = \begin{vmatrix} L_n^{(1)} & \cdots & L_n^{(k)} \\ \Delta T_1^{(n)} & \cdots & \Delta T_k^{(n)} \\ \vdots & & \vdots \\ \Delta T_1^{(n+k-2)} & \cdots & \Delta T_k^{(n+k-2)} \end{vmatrix}.$$

This transformation, given in [13], generalizes Shanks' which is recovered for the choice $T_i^{(n)} = S_{n+i-1}$.
A possible generalization consists in taking

$$L_n(u) = \sum_{i=1}^{k} b_i^{(n)} L_n(u/D^{(i)})$$

where the $D^{(i)}$'s are known sequences.

3.13 Least squares extrapolation

T_n, as defined in section 2, is the value minimizing $f(t) = (tL_n(1/D) - L_n(S/D))^2$. Let us now define T_n as the value minimizing

$$f(t) = \sum_{i=0}^{k} \left[tL_n^{(i)}(1/D) - L_n^{(i)}(S/D) \right]^2$$

where $L^{(1)}, L^{(2)}, \cdots L^{(k)}$ are difference operators. We obtain

$$T_n = \frac{\sum_{i=0}^{k} L_n^{(i)}(1/D) L_n^{(i)}(S/D)}{\sum_{i=0}^{k} \left(L_n^{(i)}(1/D) \right)^2}.$$

If the difference operators are taken as $L_n^{(i)}(u) = L_{n+i}(u)$, where L is some difference operator, then we recover an extrapolation procedure, called extrapolation in the least squares sense, which was studied by Cordellier [27] (see also [11]).
If we define the operator L' by

$$L'_n(u) = \sum_{i=0}^{k} L_n^{(i)}(1/D) L_n^{(i)}(u)$$

then $T_n = L'_n(S/D)/L'_n(1/D)$ which shows that such an extrapolation process can also be considered as the composite sequence transformation corresponding to the choice $b_i^{(n)} = L_n^{(i)}(1/D)$.

3.14 Cauchy–type approximants

If we define β_n by

$$\beta_n = -L_n(b)/L_n(S/D)$$

then the most general form of the transformation T, as given in section 2 with an auxiliary sequence b, can be written

$$T_n = (1 - \beta_n) \frac{L_n(S/D)}{L_n(1/D)}$$

which shows that we have, in fact, introduced a correction factor in its simplest form (corresponding to $b = (0)$).
Let us see now how, using this expression, the Cauchy-type approximants can be put into our framework.
Let $f(z) = \sum_{i=0}^{\infty} c_i z^i$ be a formal power series and $f_n(z)$ its partial sums. Let $g(z) = \sum_{i=0}^{\infty} d_i z^i$ be a known auxiliary series with partial sums $g_n(z)$. We set $r_n(z) = g(z) - g_n(z)$. The Cauchy–type approximants [18] are defined by

$$C_n(z) = \frac{h_n(z)}{g(z)} = \frac{\sum_{i=0}^{n} a_i z^i}{g(z)}$$

where $h_n(z) = \sum_{i=0}^{n} a_i z^i$ is the n–th partial sum of the series $f(z)g(z)$. Thus

$$a_i = d_0 c_i + d_1 c_{i-1} + \cdots + d_i c_0$$

and $C_n(z)$ can also be written as

$$C_n(z) = B_0^{(n)} f_0(z) + B_1^{(n)} f_1(z) + \cdots + B_n^{(n)} f_n(z)$$

with $B_i^{(n)} = d_{n-i} z^{n-i}/g(z)$ for $i = 0, \ldots, n$.
Taking $D_n = 1$ and defining the operator L by

$$L_n(u) = \sum_{i=0}^{n} d_{n-i} z^{n-i} u_i$$

we obtain

$$C_n(z) = (1 - \beta_n) \frac{L_n(S/D)}{L_n(1/D)}$$

with $\beta_n = r_n(z)/g(z)$. It must be noticed that (β_n) tends to 0 when n goes to infinity.
Acceleration properties of the Cauchy–type approximants were studied in [35, 36, 37].

3.15 Error control

Procedures for controlling the error in convergence acceleration methods were introduced in [12]. They consist in setting

$$T_n - S = \frac{L_n(S/D)}{L_n(1/D)} - S = \frac{L_n(a)}{L_n(1/D)} = e_n D_n^{'}$$

with $e_n = L_n(a)$ and $D_n^{'} = 1/L_n(1/D)$. Then, a second sequence transformation T^* is defined by

$$\begin{aligned} T_n^* &= (1 - \beta_n) \frac{L_n(S/D)}{L_n(1/D)} \\ &= T_n - \gamma_n D_n^{'} = T_n(\gamma_n) \end{aligned}$$

with $\gamma_n = \beta_n L_n(S/D)$. It can be proved that, under certain assumptions, the limit S_∞ of the sequence (S_n) belongs to the interval with bounds $T_n(\gamma_n)$ and $T_n(-\gamma_n)$.
In our case, if

$$\gamma_n = \frac{\Delta T_n}{\Delta D_n^{'}} \frac{1}{1 + D_{n+1}^{'}/D_n^{'}}$$

and if

$$\lim_{n\to\infty} \frac{L_n(1/D)}{L_{n+1}(1/D)} = \lim_{n\to\infty} \frac{L_{n+1}(a)}{L_n(a)} \neq \pm 1$$

then

$$\lim_{n\to\infty} \frac{T_n^* - S}{T_n - S} = 0.$$

Moreover, if we set

$$T_n^*(\varepsilon) = T_n - (1+\varepsilon)\gamma_n D_n^{'}$$

then, under the preceding assumptions, $\forall \varepsilon \neq 0, \exists N, \forall n \geq N, S_\infty$ belongs to the interval whose bounds are $T_n^*(\varepsilon)$ and $T_n^*(-\varepsilon)$.
These results can be applied to the Cauchy–type approximants introduced in the preceding example. These approximants are recovered by the choice

$$\gamma_n = \frac{r_n(z)}{g(z)} h_n(z)$$

and we have

$$\lim_{n \to \infty} \frac{C_n(z) - f(z)}{f_n(z) - f(z)} = 0$$

if

$$\lim_{n \to \infty} \frac{r_n(z)}{g_n(z) - h_n(z)/f(z)} = 1.$$

4 Composition of operators

In this section it will be easier to denote by $L(u_n)$ the n–th term of the sequence $L(u)$. We shall use again several difference operators denoted by $L^{(i)}$.

4.1 A particular case

As in section 2, let us assume that

$$S_\infty - S_n = a_n D_n.$$

In practical situations, most of the time, an annihilation operator for the sequence (a_n) is not known. Thus we begin by choosing an operator $L^{(1)}$. Applying it to the previous relation leads to a first sequence transformation defined by

$$T_1^{(n)} = L^{(1)}(S_n/D_n)/L^{(1)}(1/D_n)$$

and we have

$$S_\infty - T_1^{(n)} = L^{(1)}(a_n)/L^{(1)}(1/D_n).$$

Setting $a_1^{(n)} = L^{(1)}(a_n)$ and $D_1^{(n)} = 1/L^{(1)}(1/D_n)$, we have

$$S_\infty - T_1^{(n)} = a_1^{(n)} D_1^{(n)}.$$

Let us now choose another difference operator $L^{(2)}$ (which can be the same as the first one) and apply it to the previous relation. It leads to a second sequence transformation defined by

$$T_2^{(n)} = L^{(2)}(T_1^{(n)}/D_1^{(n)})/L^{(2)}(1/D_1^{(n)})$$

and we have

$$S_\infty - T_2^{(n)} = L^{(2)}(a_1^{(n)})/L^{(2)}(1/D_1^{(n)}).$$

It is easy to see that

$$T_2^{(n)} = \frac{L^{(2)}L^{(1)}(S_n/D_n)}{L^{(2)}L^{(1)}(1/D_n)}$$

and so on.

Thus we have the following iterative use of the procedure.
We set

$$T_0^{(n)} = S_n \quad \text{and} \quad D_0^{(n)} = D_n.$$

Then, for $k = 0, 1, \ldots$, we set

$$T_{k+1}^{(n)} = \frac{L^{(k+1)}(T_k^{(n)}/D_k^{(n)})}{L^{(k+1)}(1/D_k^{(n)})},$$

and

$$D_{k+1}^{(n)} = 1/L^{(k+1)}(1/D_k^{(n)}).$$

We have

$$S_\infty - T_{k+1}^{(n)} = a_{k+1}^{(n)} D_{k+1}^{(n)}$$

with

$$a_{k+1}^{(n)} = L^{(k+1)}(a_k^{(n)}).$$

It can be easily checked that

$$T_k^{(n)} = \frac{L^{(k)} \cdots L^{(1)}(S_n/D_n)}{L^{(k)} \cdots L^{(1)}(1/D_n)},$$

$$a_k^{(n)} = L^{(k)} \cdots L^{(1)}(a_n)$$

and

$$D_k^{(n)} = 1/L^{(k)} \cdots L^{(1)}(1/D_n).$$

For example, if $\forall k, L^{(k)}$ is the operator Δ, we have

$$T_{k+1}^{(n)} = T_k^{(n)} - \frac{\Delta T_k^{(n)}}{\Delta D_k^{(n)}} D_k^{(n)}$$

where Δ operates on the superscripts n, and

$$D_{k+1}^{(n)} = -D_k^{(n)} D_k^{(n+1)} / \Delta D_k^{(n)}.$$

This is a particular case of the algorithm (30) given by Homeier [32] when his quantities $\Delta r_k^{(n)}$ are all taken equal to 1.
As another application of this iterative procedure, let us consider the family of sequence transformations defined by

$$T_k^{(n)} = \frac{L^{(k)}(S_n/D_n)}{L^{(k)}(1/D_n)}$$

where the difference operators $L^{(k)}$ satisfy the following recurrence relation

$$L^{(k+1)}(u_n) = \alpha_n^{(k)} L^{(k)}(u_{n+1}) + \beta_n^{(k)} L^{(k)}(u_n)$$

with $L^{(0)}(u_n) = u_n, (\alpha_n^{(k)})$ and $(\beta_n^{(k)})$ being given auxiliary sequences of numbers. Then, we can write

$$\begin{aligned} T_{k+1}^{(n)} &= \frac{L^{(k+1)}(S_n/D_n)}{L^{k+1)}(1/D_n)} = \frac{\alpha_n^{(k)} L^{(k)}(S_{n+1}/D_{n+1}) + \beta_n^{(k)} L^{(k)}(S_n/D_n)}{\alpha_n^{(k)} L^{(k)}(1/D_{n+1}) + \beta_n^{(k)} L^{(k)}(1/D_n)} \\ &= \frac{\alpha_n^{(k)} L^{(k)}(1/D_{n+1}) T_k^{(n+1)} + \beta_n^{(k)} L^{(k)}(1/D_n) T_k^{(n)}}{\alpha_n^{(k)} L^{(k)}(1/D_{n+1}) + \beta_n^{(k)} L^{(k)}(1/D_n)}. \end{aligned}$$

So we get

$$T_{k+1}^{(n)} = \frac{\mathcal{L}^{(k+1)}(T_k^{(n)}/D_k^{(n)})}{\mathcal{L}^{(k+1)}(1/D_k^{(n)})},$$

with

$$\begin{aligned} \mathcal{L}^{(k+1)}(u_n) &= \alpha_n^{(k)} u_{n+1} + \beta_n^{(k)} u_n \\ D_k^{(n)} &= 1/L^{(k)}(1/D_n) = 1/\mathcal{L}^{(k)}\mathcal{L}^{(k-1)} \cdots \mathcal{L}^{(1)}(1/D_n). \end{aligned}$$

We can easily see that

$$D_{k+1}^{(n)} = 1/\mathcal{L}^{(k+1)}(1/D_k^{(n)})$$

which shows that $T_{k+1}^{(n)}$ can be obtained by composing difference operators.
As an example, let us consider Levin's transforms introduced in section 3.5. The algorithm for their implementation given in [50] enables us to write

$$T_k^{(n)} = \frac{\Delta^k((n+1)^{k-1} S_n/D_n)}{\Delta^k((n+1)^{k-1}/D_n)} = \frac{L^{(k)}(S_n/D_n)}{L^{(k)}(1/D_n)}$$

with

$$L^{(k+1)}(u_n) = L^{(k)}(u_{n+1}) - \frac{(n+1)(n+k+1)^{k-1}}{(n+k+2)^k} L^{(k)}(u_n), \qquad L^{(0)}(u_n) = u_n$$

and D_n chosen according to the Levin's transform under consideration. We immediately see that this algorithm fits into the previous iterative procedure by choosing

$$\alpha_n^{(k)} = 1 \quad \text{and} \quad \beta_n^{(k)} = -\frac{(n+1)(n+k+1)^{k-1}}{(n+k+2)^k}.$$

4.2 The general case

It is also possible to iterate as before, but with arbitrary $D_k^{(n)}$'s no more related to the $D_{k-1}^{(n)}$'s by the above recurrence relation. This is the most general case and it includes, for example, the E–algorithm since we have

$$E_k^{(n)} = \frac{\Delta(E_{k-1}^{(n)}/g_{k-1,k}^{(n)})}{\Delta(1/g_{k-1,k}^{(n)})}$$

with $E_0^{(n)} = S_n$ and the quantities $g_{k,i}^{(n)}$ computed by

$$g_{k,i}^{(n)} = \frac{\Delta(g_{k-1,i}^{(n)}/g_{k-1,k}^{(n)})}{\Delta(1/g_{k-1,k}^{(n)})}$$

with $g_{0,i}^{(n)} = g_i(n)$.

Overholt's process (example 3.7) and the θ–algorithm (example 3.10) also fit into this category.

Let us set

$$\begin{aligned} b_k^{(n)} &= \frac{1}{D_k^{(n)} L^{(k)}(1/D_{k-1}^{(n)})}, \\ a_k^{(n)} &= b_k^{(n)} L^{(k)}(a_{k-1}^{(n)}). \end{aligned}$$

We have

$$T_k^{(n)} = \frac{L^{(k)}(b_{k-1}^{(n)} L^{(k-1)}(\cdots L^{(2)}(b_1^{(n)} L^{(1)}(S_n/D_n)\cdots)}{L^{(k)}(b_{k-1}^{(n)} L^{(k-1)}(\cdots L^{(2)}(b_1^{(n)} L^{(1)}(1/D_n)\cdots)}.$$

Let us denote, in this general case, by T_k the transformation $(S_n) \longmapsto (T_k^{(n)})$ for a fixed value of k. The kernel of such a composed transformation was studied in [22] where the following results were proved

Theorem 1 :
The kernel of T_{k+1} contains the kernel of T_k.

Theorem 2 :
The kernel of T_k is the set of sequences such that $\forall n$,

$$S_\infty - S_n = a_n D_n \text{ with } L^{(k)}(b_{k-1}^{(n)} L^{(k-1)}(\cdots L^{(2)}(b_1^{(n)} L^{(1)}(a_n)\cdots) = 0.$$

When the operators $L^{(i)}$ are all identical to Δ then, by the theory developed in [26], we know that the quantities $T_k^{(n)}$ can be expressed as a ratio of two determinants. However, these determinants are only known in a few particular cases and for many transformations they still have to be found.

Let us now give three examples.

i) We consider the sequence of transformations given by

$$\begin{aligned} T_0^{(n)} &= S_n \\ T_{k+1}^{(n)} &= T_k^{(n)} - \frac{a_n(\Delta T_k^{(n)})^2}{a_{n+1}\Delta T_k^{(n+1)} - a_n \Delta T_k^{(n)}}. \end{aligned}$$

These transformations were studied by Matos [34] in the cases $a_n = n$ and $a_n = n \log n$. It is easy to see that they can be obtained by composition of operators in two different ways. First of all, these transformations can be written as

$$\begin{aligned} T_0^{(n)} &= S_n \\ T_{k+1}^{(n)} &= \frac{L(T_k^{(n)}/D_k^{(n)})}{L(1/D_k^{(n)})} \end{aligned}$$

with $L=\Delta$ and $D_k^{(n)}=a_n\Delta T_k^{(n)}$.
The same sequence of transformations can also be obtained as in subsection 4.1 with

$$L^{(k+1)}(u_n)=\frac{D_k^{(n+1)}}{a_{n+1}\Delta T_k^{(n+1)}}u_{n+1}-\frac{D_k^{(n)}}{a_n\Delta T_k^{(n)}}u_n,$$

and

$$D_{k+1}^{(n)}=1/L^{(k+1)}(1/D_k^{(n)}).$$

ii) The transformations defined by

$$\begin{aligned} T_0^{(n)} &= S_n \\ T_{k+1}^{(n)} &= T_k^{(n)}-\frac{\Delta T_k^{(n)}}{\rho_{k+1}-1} \end{aligned}$$

are recovered by the choice $L=\Delta$ and $D_k^{(n)}=\rho_{k+1}^n$. Such transformations were introduced by Matos [34]. They were proved to accelerate the convergence of monotone sequences such that, $\forall n$

$$\Delta S_n=a_1\rho_1^n+a_2\rho_2^n+\cdots$$

with $1>\rho_1>\rho_2>\cdots>0$ and $\forall i, a_i\neq 0$, that is, we have $\forall k$,

$$\lim_{n\to\infty}\left(T_{k+1}^{(n)}-S\right)/\left(T_k^{(n)}-S\right)=0.$$

iii) Let us consider again Levin's transforms as defined in the previous section. If we set $\gamma_n^{(k)}=n\cdots(n+k)=(n)_{k+1}$, the Pochhammer symbol, then we can write

$$L^{(k+1)}(u_n)=\frac{\Delta(L^{(k)}(u_n)\gamma_n^{(k)}(n+k)^{k-1})}{(n+k+1)^{k-1}\gamma_{n+1}^{(k)}}$$

which gives

$$T_k^{(n)}=\frac{L^{(k)}(S_n/D_n)}{L^{(k)}(1/D_n)}=\frac{\Delta\left(\dfrac{L^{(k-1)}(S_n/D_n)}{L^{(k-1)}(1/D_n)}\gamma_n^{(k-1)}(n+k-1)^{k-2}L^{(k-1)}(1/D_n)\right)}{\Delta\left(\gamma_n^{(k-1)}(n+k-1)^{k-2}L^{(k-1)}(1/D_n)\right)}.$$

Setting

$$D_{k-1}^{(n)}=\frac{1}{\gamma_n^{(k-1)}(n+k-1)^{k-2}L^{(k-1)}(1/D_n)}$$

we get

$$T_k^{(n)}=\frac{\Delta\left(T_{k-1}^{(n)}/D_{k-1}^{(n)}\right)}{\Delta\left(1/D_{k-1}^{(n)}\right)}.$$

In this case, the $D_k^{(n)}$'s are not related by $D_k^{(n)} = 1/\Delta(1/D_{k-1}^{(n)})$ but it is easy to see that they satisfy

$$D_k^{(n)} = \frac{1}{n(n+k)\Delta(1/D_{k-1}^{(n)})}$$

which shows that Levin's transforms also fit into the general case with

$$b_k^{(n)} = n(n+k).$$

Iteration of sequence transformations is considered in [52] and their kernels are studied in [22].

5 Conclusions

As can be seen from what precedes, the framework developed in this paper is quite powerful and interesting and it could certainly be extended furthermore. In particular, an important question will be to study the convergence and acceleration properties of extrapolation algorithms within this formalism in order to obtain more general results than those actually known. Some results in this direction have already been obtained by Matos [38]. The mechanism introduced in [23] has been extended to the vector and matrix cases in [24]. Including in this mechanism the recent interpretation of the vector ε–algorithm of Wynn [58] obtained by Salam [46] is under consideration.

Acknowledgements: We would like to thank the two referees for their careful reading of the manuscript and for suggesting several improvements.
This work was supported by the Human Capital and Mobility Programme of the European Community, Project ROLLS, Contract CHRX–CT93–0416 (DG 12 COMA).

References

[1] A.C. Aitken, On Bernoulli's numerical solution of algebraic equations, Proc. R. Soc. Edinb., 46 (1926) 289–305.

[2] G.A. Baker Jr., P.R. Graves–Morris, *Padé Approximants*, 2 vols., Addison–Wesley, Reading, 1981.

[3] S. Banach, *Théorie des Opérations Linéaires*, Monografie Matematyczne, vol.1, Warszawa, 1932.

[4] M.D. Benchiboun, *Etude de Certaines Généralisations du Δ^2 d'Aitken et Comparaison de Procédés d'Accélération de la Convergence*, Thèse 3ème cycle, Université des Sciences et Technologies de Lille, 1987.

[5] C. Brezinski, Accélération de suites à convergence logarithmique, C.R. Acad. Sci. Paris, 273 A (1971) 727–730.

[6] C. Brezinski, Etudes sur les ε et ρ–algorithmes, Numer. Math., 17 (1971) 153–162.

[7] C. Brezinski, Conditions d'application et de convergence de procédés d'extrapolation, Numer. Math., 20 (1972) 64–79.

[8] C. Brezinski, *Padé–Type Approximation and General Orthogonal Polynomials*, ISNM vol.50, Birkhäuser, Basel, 1980.

[9] C. Brezinski, A general extrapolation algorithm, Numer. Math., 35 (1980) 175–187.

[10] C. Brezinski, Some new convergence acceleration methods, Math. Comput., 39 (1982) 133–145.

[11] C. Brezinski, Algorithm 585: A subroutine for the general interpolation and extrapolation problems, ACM Trans. Math. Software, 8 (1982) 290–301.

[12] C. Brezinski, Error control in convergence acceleration processes, IMA J. Numer. Anal., 3 (1983) 65–80.

[13] C. Brezinski, Composite sequence transformations, Numer. Math., 46 (1985) 311–321.

[14] C. Brezinski, Quasi–linear extrapolation processes, in *Numerical Mathematics. Singapore 1988*, R.P. Agarwal et al. eds., ISNM vol. 86, Birkhäuser, Basel, 1988, pp.61–78.

[15] C. Brezinski, A new approach to convergence acceleration methods, in *Nonlinear Numerical Methods and Rational Approximation*, A. Cuyt ed., Reidel, Dordrecht, 1988, pp.373–405.

[16] C. Brezinski, A survey of iterative extrapolation by the E–algorithm, Det Kong. Norske Vid. Selsk. skr., 2 (1989) 1–26.

[17] C. Brezinski, Algebraic properties of the E–transformation, in *Numerical Analysis and Mathematical Modelling*, Banach Center Publications, vol. 24, PWN, Warsaw, 1990, pp.85–90.

[18] C. Brezinski, The asymptotic behavior of sequences and new series transformations based on the Cauchy product, Rocky Mt. J. Math., 21 (1991) 71–84.

[19] C. Brezinski, *Biorthogonality and its Applications to Numerical Analysis*, Marcel Dekker, New York, 1992.

[20] C. Brezinski, M. Redivo Zaglia, *Extrapolation Methods. Theory and Practice*, North–Holland, Amsterdam, 1991.

[21] C. Brezinski, M. Redivo Zaglia, Construction of extrapolation processes, Appl. Numer. Math., 8 (1991) 11–23.

[22] C. Brezinski, M. Redivo Zaglia, On the kernel of sequence transformations, Appl. Numer. Math., to appear.

[23] C. Brezinski, M. Redivo Zaglia, A general extrapolation procedure revisited, Advances Comput. Math., to appear.

[24] C. Brezinski, A. Salam, Matrix and vector sequence transformations revisited, Proc. Edinb. Math. Soc., to appear.

[25] C. Brezinski, J. van Iseghem, Padé approximations, in *Handbook of Numerical Analysis, vol. III*, P.G. Ciarlet and J.L. Lions eds., North–Holland, Amsterdam, 1994, pp.47–222.

[26] C. Brezinski, G. Walz, Sequences of transformations and triangular recursion schemes with applications in numerical analysis, J. Comput. Appl. Math., 34 (1991) 361–383.

[27] F. Cordellier, *Interpolation Rationnelle et Autres Questions: Aspects Algorithmiques et Numériques*, Thèse d'Etat, Université des Sciences et Technologies de Lille, 1989.

[28] J.P. Delahaye, *Sequence Transformations*, Springer Verlag, Berlin, 1988.

[29] J.E. Drummond, A formula for accelerating the convergence of a general series, Bull. Aust. Math. Soc., 6 (1972) 69–74.

[30] B. Germain–Bonne, *Estimation de la Limite de Suites et Formalisation de Procédés d'Accélération de la Convergence*, Thèse d'Etat, Université des Sciences et Technologies de Lille, 1978.

[31] T. Hvie, Generalized Neville type extrapolation schemes, BIT, 19 (1979) 204–213.

[32] H.H.H. Homeier, Some applications of nonlinear convergence accelerators, Int. J. Quantum Chem., 45 (1993) 545–562.

[33] D. Levin, Development of non–linear transformations for improving convergence of sequences, Int. J. Comput. Math., B3 (1973) 371–388.

[34] A.C. Matos, Acceleration methods based on convergence tests, Numer. Math., 58 (1990) 329–340.

[35] A.C. Matos, Construction of new transformations for lacunary power series based on the Cauchy–type approximants, Appl. Numer. Math., 7 (1991) 493–507.

[36] A.C. Matos, On the choice of the denominator in Padé and Cauchy–type approximation, in *Orthogonal Polynomials and their Applications*, C. Brezinski et al. eds., J.C. Baltzer, Basel, 1991, pp.347–352.

[37] A.C. Matos, Some new acceleration methods for periodic–linearly convergent power series, BIT, 31 (1991) 686–696.

[38] A.C. Matos, Linear difference operators and acceleration methods, to appear.

[39] L.M. Milne–Thomson, *The Calculus of Finite Differences*, Chelsea, New York, 1954.

[40] G. Mühlbach, A recurrence formula for generalized divided differences and some applications, J. Approx. Theory, 9 (1973) 165–172.

[41] K.J. Overholt, Extended Aitken acceleration, BIT, 5 (1965) 122–132.

[42] R. Powell, S.M. Shah, *Summability Theory and its Applications*, Van Nostrand Reinhold, London, 1972; 2nd revised edition, Prentice–Hall of India, New Delhi, 1988.

[43] L.F. Richardson, The approximate arithmetic solution by finite differences of physical problems involving differential equations, with an application to the stress in a masonery dam, Phil. Trans. Roy. Soc. London, ser. A, 210 (1910) 307–357.

[44] W. Romberg, Vereinfachte numerische Integration, Kgl. Norske Vid. Selsk. Forsk., 28 (1955) 30–36.

[45] A. Salam, On a generalization of the ε–algorithm, J. Comput. Appl. Math., 46 (1993) 455–464.

[46] A. Salam, Non–commutative extrapolation algorithms, Numerical Algorithms, 7 (1994) 225–251.

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

[48] A. Sidi, An algorithm for a special case of a generalization of the Richardson extrapolation process, Numer. Math., 38 (1982) 299–307.

[49] A.C. Smith, A one–parameter method for accelerating the convergence of sequences and series, Comp. and Maths. with Appls., 4 (1978) 157–171.

[50] D.A. Smith, W.F. Ford, Acceleration of linear and logarithmic convergence, SIAM J. Numer. Anal., 16 (1979) 223–240.

[51] E.J. Weniger, Nonlinear sequence transformations for the acceleration of convergence and the summation of divergent series, Computer Physics Reports, 10 (1989)189–371.

[52] E.J. Weniger, On the derivation of iterated sequence transformations for the acceleration of convergence and the summation of divergent series, Comput. Phys. Commun., 64 (1991) 19–45.

[53] J. Wimp, Toeplitz arrays, linear sequence transformations and orthogonal polynomials, Numer. Math., 23 (1974) 1–17.

[54] J. Wimp, *Sequence Transformations and Their Applications*, Academic Press, New York, 1981.

[55] J. Wimp, *Computation with Recurrence Relations*, Pitman, Boston, 1984.

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

[57] P. Wynn, On a procrustean technique for the numerical transformation of slowly convergent sequences and series, Proc. Cambridge Phil. Soc., 52 (1956) 663–671.

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