An extended procedure for extrapolation to the limit

Work in progress, not yet finished

Michela Redivo Zaglia
University of Padua - Italy
Claude Brezinski
University of Lille - France

- **The E–algorithm**
- **The extended procedure**
- **Some particular cases**

Let (S_n) be a sequence of real or complex numbers converging to a limit S.

If the convergence is slow, and if one has no access to the process producing the sequence (that is, if it is a black box), (S_n) can be transformed into a new sequence (T_n) converging to the same limit by a **sequence transformation** T.

Under some assumptions on (S_n) and T, (T_n) can converge to S faster than (S_n), that is

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

The idea behind a sequence transformation is **extrapolation to the limit**.

It is assumed that (S_n) behaves as a **model sequence** $(\widetilde{S}_n)$ depending on p parameters, and belonging to a given class $\mathcal{K}_{\mathcal{T}}$ of sequences.

These p parameters are obtained **by interpolation**, requiring that $S_i = \widetilde{S}_i$ for $i = n, \ldots, n+p-1$, thus defining a unique model sequence in $\mathcal{K}_{\mathcal{T}}$ depending on the index n (the first index used in the interpolation process).

Then, the limit of this model sequence is considered as an **approximation of** S. Since this limit depends on n, it is denoted by T_n, and, therefore, the sequence (S_n) has been transformed into the new sequence (T_n).

An example: Aitken's Δ^2 process

$$\mathcal{K}_{\mathcal{T}} = \{\widetilde{\mathbf{S}}_{\mathbf{i}} = \mathbf{S} + \alpha\lambda^{\mathbf{i}}\}.$$

$$\begin{aligned} S_n &= S + \alpha\lambda^n \\ S_{n+1} &= S + \alpha\lambda^{n+1} \\ S_{n+2} &= S + \alpha\lambda^{n+2} \end{aligned}$$

Solve this system for the unknowns α, λ and S.

They depend on n.

Thus, set

$$\mathbf{T_n} = \mathbf{S}$$

$(\mathbf{S_n})$ has been transformed into $(\mathbf{T_n})$

For any transformation, if the sequence (S_n) to be accelerated **belongs** itself to $\mathcal{K}_{\mathcal{T}}$, then, **by construction, for all** n**,** $T_n = S$, the limit of the sequence (S_n) if it converges, its antilimit otherwise.

The set $\mathcal{K}_{\mathcal{T}}$ is called the **kernel of the transformation** T. It is the set of sequences which are transformed into a constant sequence (S) (usually their limit, or their antilimit).

The study of the kernel of a transformation is based on the notion of **linear annihilation operator** for a sequence introduced by Weniger (1989).

There exist **many approaches** to sequence transformations

- by defining a kernel, then constructing the corresponding transformation, and constructing a recursive algorithm for its implementation.
- the construction can be based on error estimates,
- they can be obtained by modifying the rules of existing algorithms,
- it can make use of annihilation operators,
- it can be based on the relation between extrapolation and asymptotic expansions,
- by means of the theory of triangular recursive schemes,
- by composing together several transformations,
- by Schur complements.

We will make use of the following abbreviate notation for determinants

$$|a_i \;\; b_i \;\; c_i \;\; \cdots|_{i=n}^{i=n+k} = \begin{vmatrix} a_n & b_n & c_n & \cdots \\ a_{n+1} & b_{n+1} & c_{n+1} & \cdots \\ \vdots & \vdots & \vdots & \\ a_{n+k} & b_{n+k} & c_{n+k} & \cdots \end{vmatrix}.$$

The symbol Δ will denote the usual **forward difference operator** whose powers are defined by $\Delta^0 u_n = u_n$, and

$$\Delta^k u_n = \Delta(\Delta^{k-1} u_n) = \sum_{i=0}^{k} (-1)^i C_k^i u_{n+k-i} \quad \text{with} \quad C_k^i = \frac{k!}{i!(k-i)!}.$$

The E–algorithm

The E–algorithm is the most general extrapolation algorithm known so far. It is built from the **kernel**

$$S_n = S + a_1 g_1(n) + \cdots + a_k g_k(n), \quad n = 0, 1, \ldots$$

where the $(g_i(n))$'s are given **auxiliary sequences**.

Writing this relation for $n, \ldots, n+k$ leads to a system of $k+1$ linear equations in the $k+1$ unknowns $a_1, \ldots, a_k$, and S. Since these unknowns depend on n and k, S will be denoted by $E_k^{(n)}$, and it is given as a ratio of determinants

$$E_k^{(n)} = \frac{|S_i \;\; g_1(i) \;\; \cdots \;\; g_k(i)|_{i=n}^{i=n+k}}{|1 \;\; g_1(i) \;\; \cdots \;\; g_k(i)|_{i=n}^{i=n+k}}.$$

$E_k^{(n)} = S$ for all n if and only if (S_n) satisfies the preceding relation (kernel).

These quantities can be recursively computed by the

E–algorithm

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

$$g_{k,i}^{(n)} = \frac{\Delta(g_{k-1,i}^{(n)}/g_{k-1,k}^{(n)})}{\Delta(1/g_{k-1,k}^{(n)})}, \quad i > k \quad \textbf{(auxiliary rule)},$$

with $E_0^{(n)} = S_n$ and $g_{0,i}^{(n)} = g_i(n)$.

The operator Δ acts on the upper index n:

$$\Delta u^{(n)} = u^{(n+1)} - u^{(n)}.$$

THE EXTENDED PROCEDURE

For $k = 1, 2, \ldots$, let L_k be a **linear operator** on a set of real or complex functions $\mathcal{L}_k$ on $\mathbb{R}$ or $\mathbb{C}$, such that

$$\forall a_k \in \mathcal{L}_k, \quad L_k(a_k(x_n)) = 0, \quad n = 0, 1, \ldots,$$

where (x_n) is a sequence of points in $\mathbb{R}$ or $\mathbb{C}$.
L_k is an **annihilation operator** for $\mathcal{L}_k$.

For $k = 1, 2, \ldots$, we consider the **linear operators** Λ_k, acting on a sequence (u_n), which are recursively defined by

$$\Lambda_k(u_n) = \frac{L_k(\Lambda_{k-1}(u_n)/\Lambda_{k-1}(g_k(n)))}{L_k(1/\Lambda_{k-1}(g_k(n)))}, \quad n = 0, 1, \ldots,$$

with $\Lambda_0(u_n) = u_n$, for $n = 0, 1, \ldots$

Then, the extended transformation, called the Λ_k**–transformation** (we can identify the operator and the transformation without a risk of confusion), is defined by

$$\Lambda_k : (S_n) \longmapsto (\Lambda_k^{(n)} = \Lambda_k(S_n)), \quad k = 0, 1, \dots,$$

for $k \geq 0$ fixed, where (S_n) is the sequence to the accelerated (that is extrapolated).

The implementation of this transformation requires the computation of the auxiliary quantities $\Lambda_k(g_i(n))$ for different values of the three indexes. Thus, any algorithm for its implementation will depend on the properties (in general, the recursive ones) of the operators L_k, and it does not exist in the general case.

Denote $\Lambda_k^{(n)}$ by $\Lambda_k(S_n; g_k)$ for indicating its dependence on the auxiliary sequence g_k. By linearity of the operators L_k, and the definition of Λ_0, it holds, by induction,

Property 1 *(Quasi-linearity)*

$$\Lambda_k(aS_n + b; \alpha g_k) = a\Lambda_k(S_n; g_k) + b, \quad \forall a, b, \text{and } \alpha \neq 0.$$

According to the theory of sequence transformations, it could be written under the form

$$\Lambda_k^{(n)} = \frac{f(S_n, \ldots, S_m)}{Df(S_n, \ldots, S_m)},$$

for some function f depending on the operators L_k, where n and m are respectively the first and the last indexes of the terms used for computing $\Lambda_k^{(n)}$, where Df denotes the sum of the partial derivatives of f, and where $D^2 f$ is identically zero.

Property 2
The kernel of the Λ_k–transformation ($k \geq 1$) is the set of sequences satisfying, for all n,

$$\Lambda_{k-1}(S_n - S) = a_k(x_n)\Lambda_{k-1}(g_k(n)).$$

Property 3 *(by replacing Λ_{k-1} by its definition)*
The kernel of the Λ_k–transformation ($k \geq 2$) is the set of sequences satisfying, for all n,

$$L_{k-1}(\Lambda_{k-2}(S_n - S)/\Lambda_{k-2}(g_{k-1}(n)))$$
$$= a_k(x_n)L_{k-1}(\Lambda_{k-2}(g_k(n))/\Lambda_{k-2}(g_{k-1}(n))).$$

Property 4 *(Remainder formula)*

Assume that, for all n, $S_n = \widetilde{S}_n + r_n$, *where* $(\widetilde{S}_n)$ *belongs to the kernel of the* Λ_{k-1}*–transformation. Then, for all n,*

$$\Lambda_k^{(n)} = \Lambda_k(S_n) = S + \frac{L_k(\Lambda_{k-1}(r_n)/\Lambda_{k-1}(g_k(n))}{L_k(1/\Lambda_{k-1}(g_k(n))}.$$

Property 5

For all k, *the kernel of the* Λ_k*–transformation includes the kernel of the* Λ_{k-1}*–transformation.*

Particular cases

E-algorithm:

For the choice

$\forall k, L_k = \Delta$

we recover the E-algorithm, and we have

$\Lambda_k^{(n)} = \Lambda_k(S_n) = E_k^{(n)}$

and

$\Lambda_k(g_{k+1}(n)) = g_{k,k+1}^{(n)}.$

DRUMMOND'S PROCESS:

In 1972, Drummond proposed the sequence transformation $\Delta_m : (S_n) \longmapsto (\Delta_m^{(n)})$, for m fixed, where

$$\Delta_m^{(n)} = \frac{\Delta^m(S_n/\Delta S_n)}{\Delta^m(1/\Delta S_n)}, \quad m, n = 0, 1, \ldots.$$

It corresponds to $k = 1, L_1 = \Delta^m$

Property 6

The kernel of Drummond's Δ_m–transformation is the set of sequences such that there exist S and a polynomial P_{m-1} of degree at most $m-1$ satisfying

$$S_n - S = P_{m-1}(n)\Delta S_n, \quad n = 0, 1, \ldots$$

The kernel of the Δ_{m+1}–transformation includes the kernel of the Δ_m–transformation.

Solving this difference equation, gives

Property 7

The kernel of Drummond's Δ_m–transformation is the set of sequences of the form (assuming that $\forall i \geq 0, P_{m-1}(i) \neq -1, 0$)

$$S_n = S + \alpha \prod_{i=0}^{n-1} \left(1 + \frac{1}{P_{m-1}(i)} \right), \quad n = 0, 1, \ldots$$

Property 8

For all $m, n = 0, 1, \ldots$, it holds

$$\Delta_m^{(n)} = \frac{\left| S_i \;\; \Delta S_i \;\; i\Delta S_i \;\; \cdots \;\; i^{m-1}\Delta S_i \right|_{i=n}^{i=n+m}}{\left| 1 \;\; \Delta S_i \;\; i\Delta S_i \;\; \cdots \;\; i^{m-1}\Delta S_i \right|_{i=n}^{i=n+m}}.$$

APPLICATION TO FORMAL POWER SERIES:

We consider the formal power series

$$S(x)=\sum_{i=0}^{\infty} c_i x^i,$$

and apply Drummond's Δ_m–transformation to its partial sums

$$S_n(x)=\sum_{i=0}^{n} c_i x^i, \quad n=0,1,\ldots$$

$\Delta_m^{(n)}(x)=N_m^{(n)}(x)/D_m^{(n)}(x)$ is a rational function with a numerator of degree $n+m$ and a denominator of degree m at most, and we have

$$N_m^{(n)}(x)-S(x)D_m^{(n)}(x)=\mathcal{O}(x^{n+m+1}),$$

which shows that $\Delta_m^{(n)}(x)$ is a **Padé–type approximant** of S.

EXTENSIONS OF DRUMMOND'S PROCESS:

Drummond's transformation can be generalized by

- replacing ΔS_n by a known quantity D_n, called an error estimate,
- replacing the operator Δ^m by the divided difference operator δ^m (for $x_n = 1/(n+b)$, Levin's t–transformation is recovered),
- or both (that we call Drummond's δ_m–transformation).

Property 9

The kernel of Drummond's δ_m–transformation is the set of sequences such that there exist S and a polynomial P_{m-1} of degree at most $m-1$ satisfying

$$S_n - S = P_{m-1}(x_n)D_n, \quad n = 0, 1, \ldots$$

For the Drummond's δ_m–transformation, we also have

Property 10

For all $m, n = 0, 1, \ldots$, it holds

$$\delta_m^{(n)} = \frac{|S_i \;\; D_i \;\; x_i D_i \;\; \cdots \;\; x_i^{m-1} D_i|_{i=n}^{i=n+m}}{|1 \;\; D_i \;\; x_i D_i \;\; \cdots \;\; x_i^{m-1} D_i|_{i=n}^{i=n+m}}.$$

Remark 1 *If $D_n = x_n$, the kernel of the* **Richardson's extrapolation process** *and the corresponding ratio of determinants are recovered. Moreover, Richardson's process can be written as*

$$T_m^{(n)} = \delta^m(S_n/x_n)/\delta^m(1/x_n).$$

Remark 2 *When applied to the partial sums of a formal power series S, $\delta_m^{(n)}(x)$ is a* **Padé–type approximant** *of S for the choice $D_n = \delta S_n$.*

OTHER EXTENSIONS:

- using the q–difference operator (X.-B. Hu)

$$\Delta_q u_n = \frac{u_{qn} - u_n}{q - 1},$$

- using a linear differential operator,
- using a general linear annihilation operator,
- using the reciprocal differences operator,
- combinations of various operators....

Conclusions

We gave a general framework for the construction of extrapolation methods by using annihilation operators.

Recursive rules for the implementation have to be obtained in each particular case, since they depend on properties of the annihilation operators used.

All the preceding extensions have yet to be studied from the theoretical and the numerical point of view.

Thank you for your participation.

Have a nice trip back.

Arrivederci !