Applied Mathematical Sciences, Vol. 6, 2012, no. 24, 1181 - 1190

An Empirical Study of the ϵ-Algorithm for Accelerating Numerical Sequences

Oana Bumbariu

North University of Baia Mare
Department of Mathematics and Computer Science
Victoriei 76, 430122, Baia Mare, Romania
oanabumbariu@yahoo.com

Vasile Berinde

North University of Baia Mare
Department of Mathematics and Computer Science
Victoriei 76, 430122, Baia Mare, Romania
vberinde@ubm.ro

Abstract

In this paper we use Wynn's ϵ-algorithm to accelerate certain numerical sequences, in order to study empirically their rate of convergence, for the sequence of successive approximations associated with some test functions and ordinary differential equations. For the great majority of the test sequences, the ϵ-algorithm does not improve the convergence rate.

Subject Classification: [2000] 47H10, 65B99

Keywords: fixed point, Picard iterations, ϵ-algorithm

1 Introduction

In nonlinear analysis, the most important methods for solving functional equations are the iterative methods. Amongst these methods, the fixed point iterative methods play an important role.
For a recent survey on fixed point iterative methods we refer to [2]. The problem with the fixed point iterative methods, that is, the Picard, Krasnoselskij, Mann and Ishikawa iterations, is that, generally they converge slowly. In order to accelerate them we try to transform the slowly converging sequence into a

new one that converge to the exact limit as the first one, but faster.
Two of the most well known sequence transformations are Aitken's Δ^2 process and Richardson's extrapolation algorithm. The first acceleration method was obtained by A. C. Aitken in 1926 [1], who used it to accelerate the convergence of Bernoulli's method for computing the dominant zero of a polynomial, the method being given by

$$T_n = \frac{S_n S_{n+2} - S_{n+1}^2}{S_{n+2} - 2S_{n+1} + S_n} = S_n - \frac{(S_{n+1} - S_n)^2}{S_{n+2} - 2S_{n+1} + S_n}, \quad n = 0, 1, \dots, \tag{1}$$

where $\{S_n\}$ is the sequence to be accelerated.

In 1927, L. F. Richardson [14] proposed a technique for solving a 6th order differential eigenvalue problem, this extrapolation method being given by

$$T_{k+1}^{(n)} = \frac{x_{n+k+1} T_k^{(n)} - x_n T_k^{(n+1)}}{x_{n+k+1} - x_n}, \quad \kappa, n = 0, 1, \dots \tag{2}$$

with $T_0^{(0)} = S(x_n)$ for $n = 0, 1, \dots$. Extensions and applications of the Richardson extrapolation process can be found in [8], [9], [21].
Later, in 1955, D. Shanks [16] and W. Romberg [15] considered two transformations, the first was latter called Shanks transformation and is given by

$$T_n = e_k(S_n) = \frac{\begin{vmatrix} S_n & S_{n+1} & \dots & S_{n+k} \\ S_{n+1} & S_{n+2} & \dots & S_{n+k+1} \\ \vdots & \vdots & & \vdots \\ S_{n+k} & S_{n+k+1} & \dots & S_{n+2k} \end{vmatrix}}{\begin{vmatrix} \Delta^2 S_n & \dots & \Delta^2 S_{n+k-1} \\ \vdots & & \vdots \\ \Delta^2 S_{n+k-1} & \dots & \Delta^2 S_{n+2k-2} \end{vmatrix}}. \tag{3}$$

When $k = 1$, Shanks transformation reduces to Aitken's Δ^2 process. Romberg transformation is given by

$$T_{k+1}^{(n)} = \frac{4^{k+1} T_k^{(n+1)} - T_k^{(n)}}{4^{k+1} - 1},$$

where $T_0^{(n)}$ is obtained by trapezoidal rule with the stepzise $h_0/2^n$. It was proved by P.J. Laurent, in 1963 [11] that the Romberg process is the same as the Richardson extrapolation process when taking $x_n = h_n^2$ and $h_n = h_0/2^n$.

Then, in 1956 [23] Wynn introduced an important sequence transformation called the ϵ-algorithm and used it to computing the $e_k(S_n)$'s without computing the determinants involved in their definition. We will expose this algorithm in the next section and in the last section of the paper we shall give some numerical examples. In 1971 Brezinski [5] proved some limitations of Wynn's ϵ-algorithm and improved it with the so called θ-algorithm, which is given by

$$\theta_{2k+1}^{(n)} = \theta_{2k-1}^{(n+1)} + \frac{1}{\Delta\theta_{2k}^{(n)}}, \quad \theta_{2k+2}^{(n)} = \theta_{2k}^{(n+1)} + \frac{\Delta\theta_{2k}^{(n+1)}\Delta\theta_{2k+1}^{(n+1)}}{\Delta^2\theta_{2k+1}^{(n)}}, \tag{4}$$

where $\theta_{-1}^{(n)} = 0$, $\theta_0^{(n)} = S_n$, $\Delta\theta_k^{(n)} = \theta_k^{(n+1)} - \theta_k^{(n)}$, $\Delta^2\theta_k^{(n)} = \Delta\theta_k^{(n+1)} - \Delta\theta_k^{(n)}$ and $k, n \in \mathbb{N}$. For this algorithm particular rules were also obtained by R. Zaglia [25].
In 1973 D. Levin [12] introduced a new algorithm and then he extended it in collaboration with A. Sidi [17], [18], [19]. This algorithm is given by

$$l(n,k) = \frac{\sum_{i=0}^{k}(-1)^i \begin{pmatrix} k \\ i \end{pmatrix} (n+i)^{k-2}(\Delta_{n+i+1})(\Delta S_{n+i})^{-1}}{\sum_{i=0}^{k}(-1)^i \begin{pmatrix} k \\ i \end{pmatrix} (n+i)^{k-2}(\Delta S_{n+i})^{-1}}, \tag{5}$$

where $S_n = \sum_{i=0}^{n} c_i$ and $n, k \in \mathbb{N}$.
For a general historical development of sequence transformation see [6], [7].
On the other hand, in a recent paper [3], we used a Padé type acceleration technique for studying the possibility to accelerate the Picard iteration. The aim of this paper is to describe the Wynn's ϵ-algorithm for scalar sequences and to perform with this algorithm a similar empirical study, by considering the sequences arising in solving nonlinear equations and ordinary differential equations for some test functions that can be found in [3], [4], [20], [13].

2 Wynn's ϵ-algorithm

Peter Wynn's ϵ-algorithm is based on a recursive computation of a triangular array, called the epsilon array, in which each term of the sequence is determined by the three previous terms by a simple arithmetic. For any scalar sequence ($m = 1$), the e-transform gives the exact limit, s, of the sequence (S_n) that must be accelerated, and it is defined by

$$e_k(S_n) = \frac{\begin{vmatrix} S_n & S_{n+1} & \dots & S_{n+k} \\ \Delta S_n & \Delta S_{n+1} & \dots & \Delta S_{n+k} \\ \vdots & \vdots & & \vdots \\ \Delta S_{n+k-1} & \Delta S_{n+k} & \dots & \Delta S_{n+2k-1} \end{vmatrix}}{\begin{vmatrix} 1 & 1 & \dots & 1 \\ \Delta S_n & \Delta S_{n+1} & \dots & \Delta S_{n+k} \\ \vdots & & \vdots & \\ \Delta S_{n+k-1} & \Delta S_{n+k} \dots & \Delta S_{n+2k-1} & \end{vmatrix}} \tag{6}$$

where the errors satisfy the relation

$$S_n - s = \sum_{i=1}^{k} \alpha_i \lambda_i^n, \tag{7}$$

and α_i, λ_i are fixed. The same results occur for the vector sequence ($m > 1$), where α_i's become fixed vectors.
It is generally difficult to apply the e-transform to direct applications of (6), due to computations in determinants. Wynn discovered an algorithm without requiring the computation of determinants, called the ϵ-algorithm. It begins with

$$\epsilon_{-1}^{(n)} = 0, \quad \epsilon_0^{(n)} = S_n$$

and then compute

$$\epsilon_{k+1}^{(n)} = \epsilon_{k-1}^{(n+1)} + \frac{1}{\epsilon_k^{(n+1)} - \epsilon_k^{(n)}}. \tag{8}$$

The ϵ-algorithm is related with Shanks transformation by

$$\epsilon_{2k}^{(n)} = e_k(S_n) \quad \epsilon_{2k+1}^{(n)} = 1/e_k(\Delta S_n).$$

When $k = 1$ it reduces to Aitken's Δ^2 process. Wynn 's algorithm is one of the most important nonlinear acceleration procedure, Wynn wrote many papers about the properties and the applications of this algorithm, even a vector generalization in [22].
The ϵ's with an odd lower index are auxiliary quantities, so they can be eliminated from the algorithm. When applying the ϵ-algorithm division by zero can occur, if this divisions by zero are for some adjacent values we can ignore

them and continue the algorithm, otherwise the algorithm must be stopped. When computations are done divisions by a number close to zero can occur and the algorithm becomes numerically unstable because of the cancellation errors. In 1963 Wynn [24] proposed some particular rules for the ϵ-algorithm who are more precise.
Our aim in this paper is to test the ϵ-algorithm capabilities of accelerating some numerical sequences arising in solving nonlinear equations and ordinary differential equations by fixed point methods and by iterative methods, respectively.

3 Numerical examples

In this section we apply the ϵ-algorithm to some nonlinear equations mentioned in [3], [4] and for some ordinary differential equations that can be found in [20], [13]. To see the efficiency of the aforementioned algorithm we shall compare the sequence of ϵ's with the sequence of successive approximations, in each case. We recall that when applying the ϵ-algorithm the ϵ's with an odd lower index are only auxiliary quantities so they can be eliminated from the algorithm. All the numerical computations listed in the tables were done with Maple 13.

Example 3.1 *[3], [4] Test function: $f(x) = x^3 + 4x^2 - 10$, which has a unique root $x^* = 1.3652300135\ldots$. This equation can be rewritten into a fixed point problem by $g(x) = \frac{1}{2}\sqrt{10 - x^3}$. To apply the ϵ-algorithm we shall take the initial guess $x_0 \in \{1.3, 1.4, 1.5, 1.6, 1.7\}$. The results for Example 3.1 with the initial value $x_0 = 1.5$ are listed in Table 1. When taking the other initial values $x_0 \in \{1.3, 1.4, 1.6, 1.7\}$ we obtained the same results as for $x_0 = 1.5$.*

Table 1

Example 3.1 shows that ϵ-algorithm accelerates the sequence of successive approximations.

Example 3.2 *[3], [4] Test function $f(x) = x - \tan x = 0$, which has a unique root $x^* = 4.493410\ldots$. This equation can be rewritten as a fixed point problem by $g(x) = \tan x$. To apply the ϵ-algorithm we shall take the initial guess $x_0 \in \{4.3, 4.4, 4.5, 4.6, 4.7\}$. The results for Example 3.2 with the initial value $x_0 = 4.5$ are listed in Table 2. When taking the other initial values $x_0 \in \{4.3, 4.4, 4.6, 4.7\}$ we obtained the same results as for $x_0 = 4.5$.*

	$\epsilon_0^{(k)},\ k=\overline{0,9}$	$x_{n+1}=g(x_n)$	$x^*-\epsilon_0^{(k)},\ k=\overline{0,9}$	$x^*-x_i,\ i=\overline{0,9}$
0	1.5	1.5		
1	$\epsilon_1^{(0)}$=-4.6938168713	1.2869537676	6.0590468848	0.0782762459
2	$\epsilon_2^{(0)}$=1.3618864810	1.4025408036	0.0033435325	-0.0373107901
3	$\epsilon_3^{(0)}$=418.04269424	1.3454583740	-416.67746423	0.0197716395
4	$\epsilon_4^{(0)}$=1.3652752188	1.3751702528	-0.0000452053	-0.0099402393
5	$\epsilon_5^{(0)}$=-18040.784097	1.3600941928	18042.149327	0.0051358207
6	$\epsilon_6^{(0)}$=1.3652298573	1.3678469676	$1.562*10^{-7}$	-0.0026169541
7	$\epsilon_7^{(0)}$=$7.0343696250*10^6$	1.3638870039	$-7.0343682598*10^6$	0.0013430096
8	$\epsilon_8^{(0)}$=1.3652300135	1.3659167334	0	-0.0006867199
9	$\epsilon_9^{(0)}$=$1.0098930618*10^{10}$	1.3648782172	$-1.0098930617*10^{10}$	0.0003517963

Table 2

	$\epsilon_0^{(k)},\ k=\overline{0,9}$	$x_{n+1}=g(x_n)$	$x^*-\epsilon_0^{(k)},\ k=\overline{0,9}$	$x^*-x_i,\ i=\overline{0,9}$
0	4.5	4.5		
1	$\epsilon_1^{(0)}$=7.2816211984	4.6373320546	-2.7882111984	-0.1439220546
2	$\epsilon_2^{(0)}$=4.4977872904	13.298192493	-0.0043772904	-8.804782493
3	$\epsilon_3^{(0)}$=0.38564800862	0.89820387108	4.1077619914	3.5952061289
4	$\epsilon_4^{(0)}$=0.38564800862	1.2555203306	-2.0664745301	3.2378896694
5	$\epsilon_5^{(0)}$=-0.47856890351	3.0660290311	4.9719789035	1.4273809689
6	$\epsilon_6^{(0)}$=-0.4573035622	-0.075707770987	4.9507135622	4.5691177710
7	$\epsilon_7^{(0)}$=-3.7044172403	-0.075852747279	8.1978272403	4.5692627473
8	$\epsilon_8^{(0)}$=-0.21927926104	-0.075998559317	4.7126892610	4.5694085593
9	$\epsilon_9^{(0)}$=-3.1621097969	-0.076145215159	7.6555197969	4.5695552152

Example 3.2 shows that neither ϵ-algorithm nor the sequence of successive approximations does converge to the unique root of the proposed equation.

Example 3.3 *[3], [4] Test function $f(x)=x-3^{-x}=0$, which has a unique root $x^*=0.54780862125\ldots$. This equation can be rewritten into a fixed point problem by $g(x)=3^{-x}$. To apply the ϵ-algorithm we shall take the initial guess $x_0\in\{0.3, 0.3333333333, 0.4, 0.5, 0.6\}$. The results for Example 3.3 with the initial value $x_0=0.3333333333$ are listed in Table 3. When taking the other initial values $x_0\in\{0.3, 0.4, 0.5, 0.6\}$ we obtained the same results as for $x_0=0.3333333333$.*

Table 3

	$\epsilon_0^{(k)},\ k=\overline{0,9}$	$x_{n+1}=g(x_n)$	$x^*-\epsilon_0^{(k)},\ k=\overline{0,9}$	$x^*-x_i,\ i=\overline{0,9}$
0	0.3333333333	0.3333333333		
1	$\epsilon_1^{(0)}$=2.7775621998	0.69336127438	-2.2297535786	-0.1455526532
2	$\epsilon_2^{(0)}$=0.55432684700	0.46685562816	-0.00651822575	0.0809529930
3	$\epsilon_3^{(0)}$=-247.671590202	0.59876065792	248.21939882	-0.0509520367
4	$\epsilon_4^{(0)}$=0.54764601830	0.51798664611	0.00016260295	0.0298219751
5	$\epsilon_5^{(0)}$=4445.6521292	0.56605360613	-4445.1043206	-0.0182449849
6	$\epsilon_6^{(0)}$=0.54780972336	0.53693757026	-0.00000110211	0.0108710509
7	$\epsilon_7^{(0)}$=-1073084.621	0.55439036360	23138936.825146809	-0.065817424
8	$\epsilon_8^{(0)}$=0.54780861807	0.54386182303	$3.18 * 10^{-9}$	0.0039467982
9	$\epsilon_9^{(0)}$=251346683.3	0.55018907774	-251346682.75	-0.0023804565

Example 3.3 shows that ϵ-algorithm accelerates the sequence of successive approximations.
In what follows we apply the ϵ-algorithm to some Cauchy problems. To apply this method, the Cauchy problem should be rewritten as a corresponding Volterra integral equation.

Example 3.4 *[20] Let be the Cauchy problem for the Riccati equation*

$$y' = \frac{y^2+y-1}{x+1}, \quad y(0)=0.5. \tag{9}$$

When taking the initial guess $x_0 = 0.05$ the solution of the differential equation is $x^* = 0.4871901653\dots$. The results of numerical tests for Example 3.4 are listed in Table 4.
Table 4

	$\epsilon_0^{(k)},\ k=\overline{0,4}$	$y_i(0.05), i=\overline{0,4}$	$y-\epsilon_0^{(k)},\ k=\overline{0,4}$	$y-y_i,\ i=\overline{0,4}$
0	0.05	0.05		
1	$\epsilon_1^{(0)}$=2.284135183	0.4878024590	-1.796945018	-0.0006122937
2	$\epsilon_2^{(0)}$=0.4872105600	0.4872097587	-0.0000203947	-0.0000195934
3	$\epsilon_3^{(0)}$=-52305.63227	0.4872097587	50333.14750	$-4.750 * 10^{-7}$
4	$\epsilon_4^{(0)}$=0.4871901661	0.4871901775	$-8. * 10^{-10}$	$-1.22 * 10^{-8}$

Example 3.4 shows that ϵ-algorithm is not better than the sequence of successive approximations. Taking some other initial values $x_0 = 0.06$ and $x_0 = 0.07$ we obtained the same results as for $x_0 = 0.05$.

Example 3.5 *[13] Let be the Cauchy problem for the Riccati equation*

$$y' = ycosx, \quad y(0)=1. \tag{10}$$

When taking the initial guess $x_0 = 0.05$ the solution of the differential equation is $x^* = 1.0512491980\ldots$. The results of numerical tests for Example 3.5 are listed in Table 5.
Table 5

	$\epsilon_0^{(k)},\ k=\overline{0,4}$	$y_i(0.05), i=\overline{0,4}$	$y-\epsilon_0^{(k)},\ k=\overline{0,4}$	$y-y_i,\ i=\overline{0,4}$
0	0.05	0.05		
1	$\epsilon_1^{(0)}$=1.0000208311	1.0499791692	0.0512283668	0.0012700287
2	$\epsilon_2^{(0)}$=1.0512296898	1.0512281279	0.0000195081	0.0000210700
3	$\epsilon_3^{(0)}$=51826.4341525851	1.0512489352	-51825.3829033870	$2.6275921740 * 10^{-7}$
4	$\epsilon_4^{(0)}$=1.0512491985	1.0512491952	$-5.4326957 * 10^{-10}$	$2.77625314 * 10^{-9}$

Example 3.5 shows that ϵ-algorithm is not better than the sequence of successive approximations. Taking some other initial values $x_0 = 0.06$, $x_0 = 0.1$, $x_0 = 0.2$, $x_0 = 0.3$, $x_0 = 0.4$ and $x_0 = 0.5$ we obtained the same results as for $x_0 = 0.05$.

4 Conclusions

In this work we presented a short survey of the development of convergence acceleration methods for scalar sequences. For an extensive bibliography see the papers of Brezinski [6], [7] and Joyce [9] in the reference list.
One of the well known and important sequence acceleration method is Wynn's ϵ-algorithm. In this work we performed an empirical study between the sequence of ϵ's and the sequence of successive approximations in order to see which one has the best convergence speed. From the results presented in the last section we can say that Wynn's ϵ-algorithm does not improve significantly the convergence speed of a scalar sequence and when it does the convergence speed is improved only with a few decimals.

References

[1] A. C. Aitken, On Bernoulli's numerical solution of algebraic equations, *Proc. Roy. Soc. Edinburgh,* **46** (1926), 289-305.

[2] V. Berinde, Iterative Approximation of Fixed Points, *2nd Ed., Springer Verlag,* Berlin Heidelberg New York, 2007.

[3] V. Berinde, O. Bumbariu, Empirical study of a Padé type accelerating method of Picard iteration, *Creative Math. Inform.,* **19** No. 2 (2010), 149-159.

[4] J. Biazar and A. Amirteimoori, An improvement to the fixed point iterative method, *Appl. Math. Comput.,* **182** (2006), 567-571.

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

[6] C. Brezinski, Convergence acceleration during the 20th century, *J. Comput. Appl. Math.,* **122** No. 1-2 (2000), 1-21.

[7] C. Brezinski, A. Bultheel, Cools R., Some pioneers of extrapolation methods, The Birth of Numerical Analysis, *World Scientific Publ. Co., Singapore,* (2009), 1-22.

[8] J. Dutka, Richardson-extrapolation and Romberg-integration, *Hist. Math.,* **11** (1984), 3-21.

[9] D. C. Joyce, Survey of extrapolation processes in numerical analysis, *SIAM Rev.,* **13** (1971), 435-490.

[10] E. Kokiopoulou, P. Frossard, Accelerating distributed consensus using extrapolation, *IEEE Signal Proc. Let.,* **14** No. 10 (2007), 665-668.

[11] P. J. Laurent, Un théorème de convergence pour le procédé d'extrapolation de Richardson, *C. R. Acad. Sci. Paris,* **256** (1963), 1435-1437.

[12] D. Levin, Development of non-linear transformations for improving convergence of sequences, *Int. J. Comput. Math.,* **3** (1973), 371-388.

[13] J. I. Ramos, C. M. Garcia-López, Linearized θ-methods, I. Ordinary differential equations, *Comp. Meth. Appl. Mech. Eng.,* **129** (1996), 255-269.

[14] L. F. Richardson, The deferred approach to the limit. I: Single lattice, *Philos. Trans. Roy. Soc. London, Ser. A,* **226** (1927), 299-349.

[15] W. Romberg, Vereinfachte numerische Integration, *Norske Vid. Selsk. Forh.,* **28** (1955), 30-36.

[16] D. Shanks, Non-linear transformations of divergent and slowly convergent sequences, *J. Math. and Phys. (Cambridge, Mass.),* **34** (1955), 1-42.

[17] A. Sidi, Some properties of a generalization of the Richardson extrapolation process, *J. Inst. Math. Appl.,* **24** (1979), 327-346.

[18] A. Sidi, D. Levin, Two new classes of nonlinear transformations for accelerating the convergence of infinite integrals and series, *Appl. Math. Comput.,* **9** (1981), 175-215.

[19] A. Sidi, D. Levin, Rational approximations from the d-transformation, *IMA J. Numer. Anal.,* **2** (1982), 153-167.

[20] A. S. Telyakovskiy, Regarding polynomial approximation for ordinary differential equations, *Comput. Math. Appl.,* **55** (2008), 1122-1128.

[21] G. Walz, The history of extrapolation methods in numerical analysis, Report **130**, Universität Mannheim, Facultät für Mathematik und Informatik, 1991.

[22] P. Wynn, Acceleration technique for iterated vector and matrix problems, *Math. Comp.,* **16** (1962), 301-322.

[23] P. Wynn, On a device for computing the em(Sn) transformation, *MTAC,* **10** (1956), 91-96.

[24] P. Wynn, Singular rules for certain non-linear algorithms, *BIT,* **3** (1963), 175-195.

[25] M. Zaglia Redivo, Particular rules for the θ-algorithm, *Numer. Alg.,* **3** (1992), 353-370.

Received: August, 2011