Euro. Jnl of Applied Mathematics (2016), *vol.* 27, *pp.* 194–212.
doi:10.1017/S0956792515000479

A new method to generate non-autonomous discrete integrable systems via convergence acceleration algorithms

YI HE[1], XING-BIAO HU[2], HON-WAH TAM[3] and YING-NAN ZHANG[4]

[1]*Wuhan Institute of Physics and Mathematics, Chinese Academy of Sciences, Wuhan 430071, PR China*
[2]*LSEC, Institute of Computational Mathematics and Scientific Engineering Computing, AMSS, Chinese Academy of Sciences, P.O.Box 2719, Beijing 100190, PR China*
[3]*Department of Computer Science, Hong Kong Baptist University, Kowloon Tong, Hong Kong, PR China*
[4]*School of Mathematical Sciences, Nanjing Normal University, Nanjing 210023, PR China*
email: *heyi@lsec.cc.ac.cn; hxb@lsec.cc.ac.cn; tam@comp.hkbu.edu.hk; zhangyingnan@lsec.cc.ac.cn*

(*Received 19 June 2013; revised 13 August 2015; accepted 13 August 2015; first published online 7 September 2015*)

In this paper, we propose a new algebraic method to construct non-autonomous discrete integrable systems. The method starts from constructing generalizations of convergence acceleration algorithms related to discrete integrable systems. Then the non-autonomous version of the corresponding integrable systems are derived. The molecule solutions of the systems are also obtained. As an example of the application of the method, we propose a generalization of the multistep ε-algorithm, and then derive a non-autonomous discrete extended Lotka–Volterra equation. Since the convergence acceleration algorithm from the lattice Boussinesq equation is just a particular case of the multistep ε-algorithm, we have therefore arrived at a generalization of this algorithm. Finally, numerical experiments on the new algorithm are presented.

Key words: multistep ε-algorithm, G-transformation, Lotka–Volterra equation, Lattice Boussinesq equation

1 Introduction

Let $\{S_n\}$ be a sequence converging to some limit S. Sometimes the convergence is slow, then one can turn to *sequence transformation* which transforms the original sequence into a new one with hopefully better numerical properties. It is commonly said that the transformation $T : \{S_n\} \to \{T_n\}$ accelerates the convergence of the sequence $\{S_n\}$ if

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

In the literature, many sequence transformations have been studied for the acceleration of different kinds of sequences [5, 30, 34, 36, 37]. For most sequence transformations, the terms of the new sequences can be expressed as ratios of determinants. Computationally, it is possible to implement recursive algorithms to avoid actually evaluating the determinants.

Recently it has found that integrable systems have close connections with numerical algorithms [21–24, 26, 32, 33, 38]. Some integrable equations have been applied to design

new numerical algorithms. For example, the continuous-time Toda equation can be used to design a new algorithm for computing the Laplace transform of a given analytic function [24]. The discrete Lotka–Volterra system has applications in numerical algorithms for computing singular values [18, 19, 33]. (Consult [3, 10, 11, 20] and references therein.) Reciprocally, one can naturally expect to derive new integrable systems from some numerical algorithms. Observe that some celebrated convergence acceleration algorithms are found to be fully discrete integrable systems [21, 26]. These facts motivate us to think of constructing new integrable systems from the view point of convergence acceleration algorithms.

Before discussing more details about sequence transformations and integrable systems, it is helpful to recall some facts about the two subjects and the connection between them.

Discrete integrable systems can be considered as a specific class of discrete systems which possesses an important property usually called *integrability*. In particular, the existence of a Lax pair and a τ-function are two of the most important features of integrability shared by some famous numerical algorithms, such as Rutishauser's *qd*-algorithm [27] and Wynn's ε-algorithm [38]. We remark that corresponding to different boundary conditions different versions of discrete integrable systems are available. An example is the famous Toda equation

$$\frac{d^2 x_k}{dt^2} = e^{x_{k-1}-x_k} - e^{x_k - x_{k+1}}. \tag{1.1}$$

In the infinite chain case with $k = 0, \pm 1, \pm 2, \ldots$, we call (1.1) the *infinite Toda lattice equation*. Under the periodic boundary condition $x_{k+K_0} = x_k$ with $k = 0, \pm 1, \pm 2, \ldots$ and fixed $K_0 > 0$, equation (1.1) is referred to as the *periodic Toda lattice equation*. If $k = 0, 1, 2, \ldots$ with the boundary condition $x_0(t) = -\infty$, we call (1.1) the *semi-infinite Toda equation* or *infinite Toda molecule equation*. If $x_0(t) = -\infty$ and $x_{N+1} = +\infty$, we call (1.1) the *finite non-periodic Toda equation* or *finite molecule Toda equation*.

In connection with convergence acceleration algorithms, we are only interested in the semi-infinite or infinite molecule case of (1.1), corresponding to the semi-infinite or infinite Toda molecule equation. The solutions obtained in this way are called *molecule solutions*. As it turns out, our molecule solutions are closely related to sequence transformations.

On the other hand, integrable systems have become a hot topic since the discovery of solitons, and the search for new integrable equations is an important subject in this field. Up to now, there have been several approaches to generate new extended integrable systems. In this regard, we would like to mention two algebraic approaches to generate new extended integrable systems: the pfaffianization method to construct coupled integrable systems [7–9, 13, 14, 17, 25, 40] and the source generation method to generate soliton equations with self-consistent sources [15, 16, 35]. Recall that the pfaffianization method and the source generation method are similar in some points. They both start from determinant or pfaffian soliton solutions of the original integrable equations. Then the new solutions are constructed by pfaffianizing the dispersion relation of the elements given in the determinant solutions (for pfaffianization method) or by replacing arbitrary constants with arbitrary functions of some variables in the determinant or pfaffian (for source generation method). Finally new coupled bilinear equations satisfied by the new solutions are sought. While non-autonomous integrable systems form an important class of integrable equations, yet a systematic algebraic way to construct non-autonomous

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

integrable systems has not been found. In this paper, we present such a new algebraic method to generate non-autonomous discrete integrable systems by generalizing the molecule solutions of some discrete autonomous integrable systems that are related to sequence transformations.

In Section 2, we review the ε-algorithm and a generalization of this algorithm proposed by Brezinski, which inspires the main procedure of our method to generate non-autonomous integrable systems from some convergence acceleration methods. In Section 3, we present a generalization of the multistep ε-algorithm, and then derive a non-autonomous discrete extended Lotka–Volterra equation. Section 4 is devoted to a generalization of the convergence acceleration method obtained from the lattice Boussinesq equation, which is a particular case of the generalized multistep ε-algorithm given in Section 3. Lastly, we apply the new algorithm to some numerical examples.

2 Introduction of the new method

In this section, we recall the ε-algorithm and one of its generalizations, together with the corresponding sequence transformations. We then show the details of the procedure to construct non-autonomous discrete integrable systems. Finally we review a generalization of the G-transformation to exemplify this method.

The Shanks transformation [28, 29] is a generalization of Aitken's Δ^2 process which transforms a given sequence $\{S_n\}$ into the set of sequences $\{(e_k(S_n))\}$, whose terms are defined by

$$e_k(S_n) = \frac{\begin{vmatrix} S_n & S_{n+1} & \cdots & S_{n+k} \\ \Delta S_n & \Delta S_{n+1} & \cdots & \Delta S_{n+k} \\ \vdots & \vdots & & \vdots \\ \Delta^k S_n & \Delta^k S_{n+1} & \cdots & \Delta^k S_{n+k} \end{vmatrix}}{\begin{vmatrix} 1 & 1 & \cdots & 1 \\ \Delta S_n & \Delta S_{n+1} & \cdots & \Delta S_{n+k} \\ \vdots & \vdots & & \vdots \\ \Delta^k S_n & \Delta^k S_{n+1} & \cdots & \Delta^k S_{n+k} \end{vmatrix}}, \tag{2.1}$$

where Δ is the usual forward difference operator satisfying

$$\Delta^{i+1} S_n = \Delta^i S_{n+1} - \Delta^i S_n$$

with $\Delta^0 S_n = S_n$.

The ε-algorithm is a well-known convergence acceleration algorithm proposed by Wynn [38] to implement the Shanks transformation

$$\varepsilon_{-1}^{(n)} = 0, \quad \varepsilon_0^{(n)} = S_n, \quad n = 0, 1, \ldots, \tag{2.2}$$

$$\varepsilon_{k+1}^{(n)} = \varepsilon_{k-1}^{(n+1)} + \frac{1}{\varepsilon_k^{(n+1)} - \varepsilon_k^{(n)}}, \quad k,\ n = 0, 1, \ldots \tag{2.3}$$

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

The connection between the Shanks transformation and the ε-algorithm is given by

$$\varepsilon_{2k}^{(n)} = e_k(S_n),$$

whereas the elements with odd subscripts are only auxiliary quantities satisfying

$$\varepsilon_{2k+1}^{(n)} = \frac{1}{e_k(\Delta S_n)}.$$

In [2], the author proposed two generalizations of the ε-algorithm. The first generalization is

$$\varepsilon_{-1}^{(n)} = 0, \quad \varepsilon_0^{(n)} = S_n, \quad n = 0, 1, \ldots, \tag{2.4}$$

$$\varepsilon_{k+1}^{(n)} = \varepsilon_{k-1}^{(n+1)} + \frac{x_n}{\varepsilon_k^{(n+1)} - \varepsilon_k^{(n)}}, \quad k,\ n = 0, 1, \ldots \tag{2.5}$$

Here noted that the variable x_n is just dependent on n and has no dependence on k, which is different from Wynn's ρ- algorithm [39] as discussed in [2].

The corresponding sequence transformation can be expressed by [2, 5]

$$e_k(S_n) = \frac{\begin{vmatrix} u_n & u_{n+1} & \cdots & u_{n+k} \\ Ru_n & Ru_{n+1} & \cdots & Ru_{n+k} \\ \vdots & \vdots & & \vdots \\ R^k u_n & R^k u_{n+1} & \cdots & R^k u_{n+k} \end{vmatrix}}{\begin{vmatrix} x_n & x_{n+1} & \cdots & x_{n+k} \\ Ru_n & Ru_{n+1} & \cdots & Ru_{n+k} \\ \vdots & \vdots & & \vdots \\ R^k u_n & R^k u_{n+1} & \cdots & R^k u_{n+k} \end{vmatrix}}, \tag{2.6}$$

where $u_n = x_n S_n$. The operator R is defined by

$$R^{k+1} u_n = \Delta\left(\frac{R^k u_n}{x_n}\right) = R^k\left(\Delta\left(\frac{u_n}{x_n}\right)\right), \tag{2.7}$$

with $R^0 u_n = u_n$. For example,

$$R^2 u_n = \Delta\left(\frac{Ru_n}{x_n}\right) = \Delta\left(\frac{\Delta\left(\frac{u_n}{x_n}\right)}{x_n}\right) = \frac{u_{n+2}}{x_{n+1}x_{n+2}} - \frac{u_{n+1}}{x_n x_{n+1}} - \frac{u_{n+1}}{x_{n+1}^2} + \frac{u_n}{x_n^2}.$$

Obviously, the condition $x_n \neq 0$ for all n has to be enforced. Note that $R^k u_n$ can be expressed by the forward difference operator Δ with the sequence $\{x_n\}$ according the recursion formula (2.7). Since the expression $R^k u_n$ seems to be much simpler compared with the latter one, so the author [2] introduce the operator R to get the compact formula (2.6) for the algorithm (2.4)–(2.5).

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

Similarly, the connection between the generalized ε-algorithm (2.4)–(2.5) and the transformation (2.6) is given by

$$\varepsilon_{2k}^{(n)} = e_k(S_n), \quad \varepsilon_{2k+1}^{(n)} = \frac{1}{e_k(\Delta S_n)}.$$

Obviously, if $x_n = 1$ for all n, the following relation holds for any sequence $\{u_n\}$:

$$R^k u_n = \Delta^k u_n, \quad k = 0, 1, \ldots$$

Then transformation (2.6) reduces to the Shanks transformation (2.1) and equation (2.5) reduces to the recursive relation (2.3) of the ε-algorithm. We know that equation (2.3) is equivalent to the discrete potential KdV equation [26]. Hence equation (2.5) can be considered as the non-autonomous discrete potential KdV equation.

As mentioned in [5], the generalization of the ε-algorithm (2.4)–(2.5) can be obtained by modifying directly the rule of the ε-algorithm (2.3), and the new sequence transformation (2.6) can be derived by studying the properties of the algorithm. Here we consider the ε-algorithm and its first generalization from another point of view. Note that the Shanks transformation (2.1) has a special determinantal structure. The transformation (2.6) can be derived from (2.1) by substituting the operator R for Δ, replacing the sequence $\{S_n\}$ with $\{u_n\}$, and making some modifications on the first row of the denominator using an auxiliary sequence $\{x_n\}$. Then the recursive rule (2.5) can be constructed by determinantal identities to compute the transformation (2.6). Thus the non-autonomous form of equation (2.3) is achieved. We refer to this approach as the new method to generate non-autonomous integrable systems via convergence acceleration algorithms. We now review a generalization of the G-transformation that we present in [4] by applying this method.

First, we observe that the original G-transformation can be written in the form

$$G_k^{(n)} = \frac{\begin{vmatrix} S_n & S_{n+1} & \cdots & S_{n+k} \\ x_n & x_{n+1} & \cdots & x_{n+k} \\ \vdots & \vdots & & \vdots \\ \Delta^{k-1}x_n & \Delta^{k-1}x_{n+1} & \cdots & \Delta^{k-1}x_{n+k} \end{vmatrix}}{\begin{vmatrix} 1 & 1 & \cdots & 1 \\ x_n & x_{n+1} & \cdots & x_{n+k} \\ \vdots & \vdots & & \vdots \\ \Delta^{k-1}x_n & \Delta^{k-1}x_{n+1} & \cdots & \Delta^{k-1}x_{n+k} \end{vmatrix}}. \tag{2.8}$$

where $\{x_n\}$ is a given auxiliary sequence which can depend on some terms of the sequence $\{S_n\}$. If we introduce another auxiliary sequences $\{z_n\}$, define $u_n = S_n z_n$, $v_n = x_n z_n$, replace S_n and x_n by u_n and v_n, respectively, substitute the operator R for Δ, and change the first row in the determinant of the denominator to some consecutive terms of the sequence

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

$\{z_n\}$ in the above formula, we can obtain the generalized G-transformation in [4]:

$$G_k^{(n)} = \frac{\begin{vmatrix} u_n & u_{n+1} & \cdots & u_{n+k} \\ v_n & v_{n+1} & \cdots & v_{n+k} \\ \vdots & \vdots & & \vdots \\ R^{k-1}v_n & R^{k-1}v_{n+1} & \cdots & R^{k-1}v_{n+k} \end{vmatrix}}{\begin{vmatrix} z_n & z_{n+1} & \cdots & z_{n+k} \\ v_n & v_{n+1} & \cdots & v_{n+k} \\ \vdots & \vdots & & \vdots \\ R^{k-1}v_n & R^{k-1}v_{n+1} & \cdots & R^{k-1}v_{n+k} \end{vmatrix}}.$$

Note that when $z_n = 1$ for all n, the general G-transformation reduces to the original one.

A generalization of the G-algorithm has been proposed in [4], together with the corresponding auxiliary rs-algorithm for implementing the generalized G-transformation. A generalized qd-algorithm equivalent to the auxiliary rs-algorithm is also obtained in the form

$$e_0^{(n)} = 0, \quad q_1^{(n)} = x_{n+1}/x_n,$$

$$e_k^{(n)} + \frac{1}{z_{n+k}}\, q_k^{(n)} - \frac{1}{z_n} = e_{k-1}^{(n+1)} + \frac{1}{z_{n+k+1}}\, q_k^{(n+1)} - \frac{1}{z_{n+1}}, \tag{2.9}$$

$$q_k^{(n)} e_k^{(n)} = e_k^{(n-1)} q_{k+1}^{(n-1)}. \tag{2.10}$$

In particular, when $z_n = 1$ for all n, this algorithm reduces to the usual qd-algorithm [27], which has been shown to be simply a discrete Toda system [12] from the integrable systems' point of view. Therefore equations (2.9)–(2.10) can be considered as a non-autonomous discrete Toda system.

3 A generalization of the multistep ε-algorithm

We move on to apply the approach in Section 2 to the multistep ε-algorithm proposed in [3], which is related to an extended discrete Lotka–Volterra system.

3.1 The multistep ε-algorithm

We start with some facts on the multistep ε-algorithm [3]

$$\varepsilon_{k+1,m}^{(n)} = \varepsilon_{k-m,m}^{(n+1)} + \frac{1}{\prod_{i=1}^{m} (\varepsilon_{k-m+i,m}^{(n+1)} - \varepsilon_{k-m+i,m}^{(n)})}, \quad k, n = 0, 1, \ldots, \tag{3.1}$$

with initial values

$$\varepsilon_{-m,m}^{(n)} = 0, \ \varepsilon_{-m+1,m}^{(n)} = \varepsilon_{-m+2,m}^{(n)} = \cdots = \varepsilon_{-1,m}^{(n)} = n, \ \varepsilon_{0,m}^{(n)} = S_n, \ n = 0, 1, \ldots, \tag{3.2}$$

where m is a fixed positive integer.

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

As shown in [3], the molecule solution of equation (3.1) can be obtained by Hirota's bilinear method. By setting

$$\varepsilon_{k,m}^{(n)} = \frac{G_{k,m}^n}{F_{k,m}^n},$$

equation (3.1) is transformed into the bilinear form (for simplicity, from now on we omit the subscripts that indicate the dependence on the fixed integer m in $G_{k,m}^n$ and $F_{k,m}^n$).

$$F_{k(m+1)+1}^n G_{k(m+1)+1}^{n+1} - F_{k(m+1)+1}^{n+1} G_{k(m+1)+1}^n = -F_{k(m+1)+2}^n F_{k(m+1)}^{n+1}, \tag{3.3}$$

$$F_{k(m+1)+1}^n G_{(k-1)(m+1)+1}^{n+1} - F_{(k-1)(m+1)+1}^{n+1} G_{k(m+1)+1}^n = -F_{(k-1)(m+1)+2}^n F_{k(m+1)}^{n+1}, \tag{3.4}$$

and, for $i = 2, \ldots, m+1$,

$$F_{k(m+1)+i}^n G_{k(m+1)+i}^{n+1} - F_{k(m+1)+i}^{n+1} G_{k(m+1)+i}^n = F_{k(m+1)+i+1}^n F_{k(m+1)+i-1}^{n+1}, \tag{3.5}$$

$$F_{k(m+1)+i}^n G_{(k-1)(m+1)+i}^{n+1} - F_{(k-1)(m+1)+i}^{n+1} G_{k(m+1)+i}^n = F_{(k-1)(m+1)+i+1}^n F_{k(m+1)+i-1}^{n+1}. \tag{3.6}$$

We set

$$\mathscr{H}_k(u_n) = \begin{vmatrix} u_n & u_{n+1} & \cdots & u_{n+k-1} \\ \Delta^m u_n & \Delta^m u_{n+1} & \cdots & \Delta^m u_{n+k-1} \\ \Delta^{2m} u_n & \Delta^{2m} u_{n+1} & \cdots & \Delta^{2m} u_{n+k-1} \\ \vdots & \vdots & & \vdots \\ \Delta^{(k-1)m} u_n & \Delta^{(k-1)m} u_{n+1} & \cdots & \Delta^{(k-1)m} u_{n+k-1} \end{vmatrix},$$

$$k = 1, 2, \ldots, \quad n = 0, 1, \ldots,$$

with $\mathscr{H}_{-1}(u_n) = 0$ and $\mathscr{H}_0(u_n) = 1$. We also let

$$\psi_k(u_n) = \begin{vmatrix} n & n+1 & \cdots & n+k-1 \\ u_n & u_{n+1} & \cdots & u_{n+k-1} \\ \Delta^m u_n & \Delta^m u_{n+1} & \cdots & \Delta^m u_{n+k-1} \\ \Delta^{2m} u_n & \Delta^{2m} u_{n+1} & \cdots & \Delta^{2m} u_{n+k-1} \\ \vdots & \vdots & & \vdots \\ \Delta^{(k-2)m} u_n & \Delta^{(k-2)m} u_{n+1} & \cdots & \Delta^{(k-2)m} u_{n+k-1} \end{vmatrix},$$

$$k = 1, 2, \ldots, \quad n = 0, 1, \ldots,$$

with $\psi_{-1}(u_n) = 0$ and $\psi_0(u_n) = 1$.

The following result holds [3] for the bilinear equations (3.3)–(3.6) with specific initial values:

Lemma 1 *Given the initial values*

$$F_{-m}^n = F_{-m+1}^n = \cdots = F_0^n = 1, \quad G_{-m}^n = 0, G_{-m+1}^n = G_{-m+2}^n = \cdots = G_{-1}^n = n, \quad G_0^n = S_n,$$

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

the solution of the bilinear equations (3.3)–(3.6) *can be expressed as*

$$F^n_{(k-1)(m+1)+i} = \mathscr{H}_k(\Delta^i S_n), \quad i = 1, 2, \ldots, m+1$$
$$G^n_{(k-1)(m+1)+1} = \mathscr{H}_{k-1}(\Delta^{m+2} S_n), \quad G^n_{k(m+1)} = \mathscr{H}_{k+1}(S_n),$$
$$G^n_{(k-1)(m+1)+i} = \psi_{k+1}(\Delta^{i-1} S_n), \quad i = 2, 3, \ldots, m.$$

From Lemma 1, the determinantal expressions of the multistep ε-algorithm are obtained

$$\varepsilon^{(n)}_{k(m+1),m} = \frac{\mathscr{H}_{k+1}(S_n)}{\mathscr{H}_k(\Delta^{m+1} S_n)}, \quad k = 1, 2, \ldots, \quad n = 0, 1, \ldots$$

These expressions correspond to the multistep Shanks transformation in [3], and the intermediate quantities are given by

$$\varepsilon^{(n)}_{(k-1)(m+1)+1,m} = \frac{\mathscr{H}_{k-1}(\Delta^{m+2} S_n)}{\mathscr{H}_k(\Delta S_n)},$$
$$\varepsilon^{(n)}_{(k-1)(m+1)+i,m} = \frac{\psi_{k+1}(\Delta^{i-1} S_n)}{\mathscr{H}_k(\Delta^i S_n)}, \quad i = 2, 3, \ldots, m.$$

3.2 A generalization of the multistep ε-algorithm and the non-autonomous discrete extended Lotka–Volterra equation

We now turn to construct a generalization of the multistep ε-algorithm. First, we set

$$H_k(u_n) = \begin{vmatrix} u_n & u_{n+1} & \cdots & u_{n+k-1} \\ R^m u_n & R^m u_{n+1} & \cdots & R^m u_{n+k-1} \\ R^{2m} u_n & R^{2m} u_{n+1} & \cdots & R^{2m} u_{n+k-1} \\ \vdots & \vdots & & \vdots \\ R^{(k-1)m} u_n & R^{(k-1)m} u_{n+1} & \cdots & R^{(k-1)m} u_{n+k-1} \end{vmatrix},$$

$$k = 1, 2, \ldots, \quad n = 0, 1, \ldots,$$

$$\Phi_k(u_n) = \begin{vmatrix} x_n & x_{n+1} & \cdots & x_{n+k-1} \\ u_n & u_{n+1} & \cdots & u_{n+k-1} \\ R^m u_n & R^m u_{n+1} & \cdots & R^m u_{n+k-1} \\ R^{2m} u_n & R^{2m} u_{n+1} & \cdots & R^{2m} u_{n+k-1} \\ \vdots & \vdots & & \vdots \\ R^{(k-2)m} u_n & R^{(k-2)m} u_{n+1} & \cdots & R^{(k-2)m} u_{n+k-1} \end{vmatrix},$$

$$k = 1, 2, \ldots, \quad n = 0, 1, \ldots,$$

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

and

$$\Psi_k(u_n) = \begin{vmatrix} y_n & y_{n+1} & \cdots & y_{n+k-1} \\ u_n & u_{n+1} & \cdots & u_{n+k-1} \\ R^m u_n & R^m u_{n+1} & \cdots & R^m u_{n+k-1} \\ R^{2m} u_n & R^{2m} u_{n+1} & \cdots & R^{2m} u_{n+k-1} \\ \vdots & \vdots & & \vdots \\ R^{(k-2)m} u_n & R^{(k-2)m} u_{n+1} & \cdots & R^{(k-2)m} u_{n+k-1} \end{vmatrix},$$

$$k = 1, 2, \ldots, \quad n = 0, 1, \ldots,$$

where $\{x_n\}$ and $\{y_n\}$ are auxiliary sequences and $u_n = x_n S_n$. The operator R is defined by (2.7), with $\{y_n\}$ satisfying

$$R y_n = \Delta\left(\frac{y_n}{x_n}\right) = x_n. \tag{3.7}$$

We now show the relationship between the determinants H and Φ. We first rewrite the determinant Φ in another form, then subtract the $j-1$th column to jth column for $j = n, n-1, \ldots, 2$, and apply the definition of the operator R (2.7), then we have

$$\Phi_k(u_n) = \prod_{j=0}^{k-1} x_{n+j} \begin{vmatrix} 1 & 1 & \cdots & 1 \\ \frac{u_n}{x_n} & \frac{u_{n+1}}{x_{n+1}} & \cdots & \frac{u_{n+k-1}}{x_{n+k-1}} \\ \frac{R^m u_n}{x_n} & \frac{R^m u_{n+1}}{x_{n+1}} & \cdots & \frac{R^m u_{n+k-1}}{x_{n+k-1}} \\ \frac{R^{2m} u_n}{x_n} & \frac{R^{2m} u_{n+1}}{x_{n+1}} & \cdots & \frac{R^{2m} u_{n+k-1}}{x_{n+k-1}} \\ \vdots & \vdots & & \vdots \\ \frac{R^{(k-2)m} u_n}{x_n} & \frac{R^{(k-2)m} u_{n+1}}{x_{n+1}} & \cdots & \frac{R^{(k-2)m} u_{n+k-1}}{x_{n+k-1}} \end{vmatrix}$$

$$= \prod_{j=0}^{k-1} x_{n+j} \begin{vmatrix} 1 & 0 & \cdots & 0 \\ \frac{u_n}{x_n} & \Delta\left(\frac{u_n}{x_n}\right) & \cdots & \Delta\left(\frac{u_{n+k-2}}{x_{n+k-2}}\right) \\ \frac{R^m u_n}{x_n} & \Delta\left(\frac{R^m u_n}{x_n}\right) & \cdots & \Delta\left(\frac{R^m u_{n+k-2}}{x_{n+k-2}}\right) \\ \frac{R^{2m} u_n}{x_n} & \Delta\left(\frac{R^{2m} u_n}{x_n}\right) & \cdots & \Delta\left(\frac{R^{2m} u_{n+k-2}}{x_{n+k-2}}\right) \\ \vdots & \vdots & & \vdots \\ \frac{R^{(k-2)m} u_n}{x_n} & \Delta\left(\frac{R^{(k-2)m} u_n}{x_n}\right) & \cdots & \Delta\left(\frac{R^{(k-2)m} u_{n+k-2}}{x_{n+k-2}}\right) \end{vmatrix}$$

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

$$= \prod_{j=0}^{k-1} x_{n+j} \begin{vmatrix} 1 & 0 & \cdots & 0 \\ \frac{u_n}{x_n} & Ru_n & \cdots & Ru_{n+k-2} \\ \frac{R^m u_n}{x_n} & R^{m+1}u_n & \cdots & R^{m+1}u_{n+k-2} \\ \frac{R^{2m} u_n}{x_n} & R^{2m+1}u_n & \cdots & R^{2m+1}u_{n+k-2} \\ \vdots & \vdots & & \vdots \\ \frac{R^{(k-2)m} u_n}{x_n} & R^{(k-2)m+1}u_n & \cdots & R^{(k-2)m+1}u_{n+k-2} \end{vmatrix}$$

That is,

$$\Phi_k(u_n) = \prod_{j=0}^{k-1} x_{n+j} H_{k-1}(Ru_n). \tag{3.8}$$

However, sometimes we still use the symbol Φ instead of H for simplicity in the subsequent expressions.

We now define functions f_k^n and g_k^n by

$$f^n_{(k-1)(m+1)+1} = H_k(Ru_n), \tag{3.9}$$

$$f^n_{(k-1)(m+1)+i} = \Phi_{k+1}(R^{i-1}u_n), \quad i = 2, 3, \ldots, m+1, \tag{3.10}$$

$$g^n_{(k-1)(m+1)+1} = \Phi_k(R^{m+1}u_n), \quad g^n_{k(m+1)} = H_{k+1}(u_n), \tag{3.11}$$

$$g^n_{(k-1)(m+1)+i} = \Psi_{k+1}(R^{i-1}u_n), \quad i = 2, 3, \ldots, m, \tag{3.12}$$

for $k = 1, 2, \ldots$ Here we also omit the subscripts that indicate the dependence on the fixed integer m in g_k^n and f_k^n.

Notice that when $x_n = 1$ and $y_n = n$ $(\forall n)$, the determinants $H_k(u_n)$ and $\Psi_k(u_n)$ are identical to $\mathscr{H}_k(u_n)$ and $\psi_k(u_n)$, respectively. Therefore f_k^n and g_k^n reduce to F_k^n and G_k^n, respectively.

Now the functions f_k^n and g_k^n do not satisfy the bilinear relations (3.3)–(3.6) any more. By similar determinantal identities as the proof of equations (3.3)–(3.6), we can obtain the new equations f_k^n and g_k^n satisfy. The new results are as follows:

Theorem 1 *The functions f_k^n and g_k^n defined by* (3.9)–(3.12) *satisfy the bilinear equations*

$$\prod_{j=1}^{k+1} x_{n+j} (f^n_{k(m+1)+1} g^{n+1}_{k(m+1)+1} - f^{n+1}_{k(m+1)+1} g^n_{k(m+1)+1}) = -f^n_{k(m+1)+2} f^{n+1}_{k(m+1)}, \tag{3.13}$$

$$f^n_{k(m+1)+2} g^{n+1}_{k(m+1)+2} - f^{n+1}_{k(m+1)+2} g^n_{k(m+1)+2} = \prod_{j=0}^{k+2} x_{n+j} f^n_{k(m+1)+3} f^{n+1}_{k(m+1)+1}, \tag{3.14}$$

$$\prod_{j=1}^{k+1} x_{n+j} (f^n_{k(m+1)+1} g^{n+1}_{(k-1)(m+1)+1} - f^{n+1}_{(k-1)(m+1)+1} g^n_{k(m+1)+1}) = -f^n_{(k-1)(m+1)+2} f^{n+1}_{k(m+1)}, \tag{3.15}$$

$$f^n_{k(m+1)+2} g^{n+1}_{(k-1)(m+1)+2} - f^{n+1}_{(k-1)(m+1)+2} g^n_{k(m+1)+2} = \prod_{j=0}^{k+1} x_{n+j} f^n_{(k-1)(m+1)+3} f^{n+1}_{k(m+1)+1}, \tag{3.16}$$

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

and for $i = 3, 4, \ldots, m+1$,

$$f^{n}_{k(m+1)+i} g^{n+1}_{k(m+1)+i} - f^{n+1}_{k(m+1)+i} g^{n}_{k(m+1)+i} = x_n f^{n}_{k(m+1)+i+1} f^{n+1}_{k(m+1)+i-1}, \tag{3.17}$$

$$x_{n+k+2}(f^{n}_{k(m+1)+i} g^{n+1}_{(k-1)(m+1)+i} - f^{n+1}_{(k-1)(m+1)+i} g^{n}_{k(m+1)+i}) = x_n f^{n}_{(k-1)(m+1)+i+1} f^{n+1}_{k(m+1)+i-1}. \tag{3.18}$$

Proof We need to use the determinantal relation

$$\begin{vmatrix} x_n & x_{n+1} & \cdots & x_{n+k+2} \\ y_n & y_{n+1} & \cdots & y_{n+k+2} \\ R^i u_n & R^i u_{n+1} & \cdots & R^i u_{n+k+2} \\ \vdots & \vdots & & \vdots \\ R^{i+km} u_n & R^{i+km} u_{n+1} & \cdots & R^{i+km} u_{n+k+2} \end{vmatrix}$$

$$= \prod_{j=0}^{k+2} x_{n+j} \begin{vmatrix} 1 & 1 & \cdots & 1 \\ \frac{y_n}{x_n} & \frac{y_{n+1}}{x_{n+1}} & \cdots & \frac{y_{n+k+2}}{x_{n+k+2}} \\ \frac{R^i u_n}{x_n} & \frac{R^i u_{n+1}}{x_{n+1}} & \cdots & \frac{R^i u_{n+k+2}}{x_{n+k+2}} \\ \vdots & \vdots & & \vdots \\ \frac{R^{i+km} u_n}{x_n} & \frac{R^{i+km} u_{n+1}}{x_{n+1}} & \cdots & \frac{R^{i+km} u_{n+k+2}}{x_{n+k+2}} \end{vmatrix}$$

$$= \prod_{j=0}^{k+2} x_{n+j} \begin{vmatrix} 1 & 0 & \cdots & 0 \\ \frac{y_n}{x_n} & \Delta\left(\frac{y_n}{x_n}\right) & \cdots & \Delta\left(\frac{y_{n+k+1}}{x_{n+k+1}}\right) \\ \frac{R^i u_n}{x_n} & \Delta\left(\frac{R^i u_n}{x_n}\right) & \cdots & \Delta\left(\frac{R^i u_{n+k+1}}{x_{n+k+1}}\right) \\ \vdots & \vdots & & \vdots \\ \frac{R^{i+km} u_n}{x_n} & \Delta\left(\frac{R^{i+km} u_n}{x_n}\right) & \cdots & \Delta\left(\frac{R^{i+km} u_{n+k+1}}{x_{n+k+1}}\right) \end{vmatrix}$$

$$= \prod_{j=0}^{k+2} x_{n+j} \begin{vmatrix} x_n & x_{n+1} & \cdots & x_{n+k+1} \\ R^{i+1} u_n & R^{i+1} u_{n+1} & \cdots & R^{i+1} u_{n+k+1} \\ R^{i+1+m} u_n & R^{i+1+m} u_{n+1} & \cdots & R^{i+1+m} u_{n+k+1} \\ \vdots & \vdots & & \vdots \\ R^{i+1+km} u_n & R^{i+1+km} u_{n+1} & \cdots & R^{i+1+km} u_{n+k+1} \end{vmatrix} \tag{3.19}$$

in subsequent calculations which can be obtained in similar way to the formula (3.8) with

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

the relation (3.7) . We first prove the bilinear identity (3.17) for $i = 3, 4, \ldots, m$. If we define

$$D \equiv \begin{vmatrix} x_n & x_{n+1} & \cdots & x_{n+k+2} \\ y_n & y_{n+1} & \cdots & y_{n+k+2} \\ R^{i-1}u_n & R^{i-1}u_{n+1} & \cdots & R^{i-1}u_{n+k+2} \\ \vdots & \vdots & & \vdots \\ R^{i-1+km}u_n & R^{i-1+km}u_{n+1} & \cdots & R^{i-1+km}u_{n+k+2} \end{vmatrix},$$

then by equations (3.8) and (3.19) we have

$$\begin{aligned} &D = \prod_{j=0}^{k+2} x_{n+j}\Phi_{k+2}(R^i u_n) = \prod_{j=0}^{k+2} x_{n+j} f^n_{k(m+1)+i+1}, \\ &D(1,2|1,k+3) = \frac{\Phi_{k+2}(R^{i-2}u_{n+1})}{\prod_{j=1}^{k+2} x_{n+j}} = \frac{f^{n+1}_{k(m+1)+i-1}}{\prod_{j=1}^{k+2} x_{n+j}}, \\ &D(1|1) = \Psi_{k+2}(R^{i-1}u_{n+1}) = g^{n+1}_{k(m+1)+i}, \\ &D(2|k+3) = \Phi_{k+2}(R^{i-1}u_n) = f^n_{k(m+1)+i}, \\ &D(1|k+3) = \Psi_{k+2}(R^{i-1}u_n) = g^n_{k(m+1)+i}, \\ &D(2|1) = \Phi_{k+2}(R^{i-1}u_{n+1}) = f^{n+1}_{k(m+1)+i}, \end{aligned}$$

where $D(j|k)$ and $D(j,k|p,q)$ are $(m+2)$th order and $(m+1)$th order determinants obtained by eliminating the jth row and the kth column from the determinant D and by eliminating the jth and kth rows and the pth and qth columns from D, respectively.

From the above results, we see that the bilinear equation (3.17) is equivalent to the Jacobi identity [6]

$$DD(1,2|1,k+3) = D(1|1)D(2|k+3) - D(1|k+3)D(2|1).$$

Bilinear equations (3.13) and (3.14) can be proved in a similar way. The case when $i = m+1$ for equation (3.17) can also be proved directly by the Jacobi identity.

Next we prove equation (3.18) for $i = 3, 4, \ldots, m$. By equations (3.8), (3.19) and Schwein's determinantal identity [1], we have

$$\begin{vmatrix} x_n & x_{n+1} & \cdots & x_{n+k+1} \\ R^{i-1}u_n & R^{i-1}u_{n+1} & \cdots & R^{i-1}u_{n+k+1} \\ \vdots & \vdots & & \vdots \\ R^{i-1+km}u_n & R^{i-1+km}u_{n+1} & \cdots & R^{i-1+km}u_{n+k+1} \end{vmatrix}$$

$$\begin{vmatrix} y_{n+1} & y_{n+2} & \cdots & y_{n+k+1} \\ R^{i-1}u_{n+1} & R^{i-1}u_{n+2} & \cdots & R^{i-1}u_{n+k+1} \\ \vdots & \vdots & & \vdots \\ R^{i-1+(k-1)m}u_{n+1} & R^{i-1+(k-1)m}u_{n+2} & \cdots & R^{i-1+(k-1)m}u_{n+k+1} \end{vmatrix}$$

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

$$
-\begin{vmatrix}
x_{n+1} & x_{n+2} & \cdots & x_{n+k+1} \\
R^{i-1}u_{n+1} & R^{i-1}u_{n+2} & \cdots & R^{i-1}u_{n+k+1} \\
\vdots & \vdots & & \vdots \\
R^{i-1+(k-1)m}u_{n+1} & R^{i-1+(k-1)m}u_{n+2} & \cdots & R^{i-1+(k-1)m}u_{n+k+1}
\end{vmatrix}
\begin{vmatrix}
y_n & y_{n+1} & \cdots & y_{n+k+1} \\
R^{i-1}u_n & R^{i-1}u_{n+1} & \cdots & R^{i-1}u_{n+k+1} \\
\vdots & \vdots & & \vdots \\
R^{i-1+km}u_n & R^{i-1+km}u_{n+1} & \cdots & R^{i-1+km}u_{n+k+1}
\end{vmatrix}
$$

$$
=\begin{vmatrix}
x_n & x_{n+1} & \cdots & x_{n+k+1} \\
y_n & y_{n+1} & \cdots & y_{n+k+1} \\
R^{i-1}u_n & R^{i-1}u_{n+1} & \cdots & R^{i-1}u_{n+k+1} \\
\vdots & \vdots & & \vdots \\
R^{i-1+(k-1)m}u_n & R^{i-1+(k-1)m}u_{n+1} & \cdots & R^{i-1+(k-1)m}u_{n+k+1}
\end{vmatrix}
\begin{vmatrix}
R^{i-1}u_{n+1} & R^{i-1}u_{n+2} & \cdots & R^{i-1}u_{n+k+1} \\
R^{i-1+m}u_{n+1} & R^{i-1+m}u_{n+2} & \cdots & R^{i-1+m}u_{n+k+1} \\
\vdots & \vdots & & \vdots \\
R^{i-1+km}u_{n+1} & R^{i-1+km}u_{n+2} & \cdots & R^{i-1+km}u_{n+k+1}
\end{vmatrix}.
$$

In other words,

$$
\begin{aligned}
&\Phi_{k+2}(R^{i-1}u_n)\Psi_{k+1}(R^{i-1}u_{n+1}) - \Phi_{k+1}(R^{i-1}u_{n+1})\Psi_{k+2}(R^{i-1}u_n) \\
&\quad = \prod_{j=0}^{k+1} x_{n+j}\Phi_{k+1}(R^i u_n)\frac{\Phi_{k+2}(R^{i-2}u_{n+1})}{\prod_{j=1}^{k+2} x_{n+j}},
\end{aligned}
$$

which is equivalent to equation (3.18). The case when $i = m + 1$ for equation (3.18) and the bilinear equations (3.15)–(3.16) can be proved in a similar way. □

Notice that equations (3.13)–(3.18) are non-autonomous form of (3.3)–(3.6). When $x_n = 1$ ($\forall n$), then equations (3.13)–(3.18) reduce to (3.3)–(3.6).

Next we consider the non-linear equation of the bilinear form (3.13)–(3.18). If we set

$$
\varepsilon_{k,m}^{(n)} = \frac{g_k^n}{f_k^n}, \tag{3.20}
$$

we have

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

Theorem 2 *Given the initial conditions*

$$f_{-m}^n = f_{-m+1}^n = \cdots = f_0^n = x_n, \quad g_{-m}^n = 0, g_{-m+1}^n = g_{-m+2}^n = \cdots = g_{-1}^n = y_n, \quad g_0^n = u_n,$$

if we define $\varepsilon_{k,m}^{(n)}$ *by* (3.20), *then bilinear equations* (3.13)–(3.18) *lead to the generalized multistep* ε*-algorithm*

$$\varepsilon_{k+1,m}^{(n)} = \varepsilon_{k-m,m}^{(n+1)} + \frac{(x_n)^m}{\prod_{i=1}^m (\varepsilon_{k-m+i,m}^{(n+1)} - \varepsilon_{k-m+i,m}^{(n)})}, \quad k, n = 0, 1, \ldots \tag{3.21}$$

with initial values

$$\varepsilon_{-m,m}^{(n)} = 0, \ \varepsilon_{-m+1,m}^{(n)} = \varepsilon_{-m+2,m}^{(n)} = \cdots = \varepsilon_{-1,m}^{(n)} = \frac{y_n}{x_n}, \ \varepsilon_{0,m}^{(n)} = S_n, \quad n = 0, 1, \ldots \tag{3.22}$$

For all k *and* n, *it holds that*

$$\varepsilon_{k(m+1),m}^{(n)} = \frac{H_{k+1}(u_n)}{\Phi_{k+1}(R^m u_n)}, \tag{3.23}$$

which can be considered as a generalization of the multistep Shanks transformation proposed in [3]. For $k = 1, 2, \ldots$ *and* $n = 0, 1, \ldots$, *the intermediate quantities can be expressed by*

$$\varepsilon_{(k-1)(m+1)+1,m}^{(n)} = \frac{\Phi_k(R^{m+1} u_n)}{H_k(R u_n)}, \tag{3.24}$$

$$\varepsilon_{(k-1)(m+1)+i,m}^{(n)} = \frac{\Psi_{k+1}(R^{i-1} u_n)}{\Phi_{k+1}(R^{i-1} u_n)}, \quad i = 2, 3, \ldots, m. \tag{3.25}$$

Proof The validity of equation (3.21) can be proved directly by bilinear relations (3.13)–(3.18). The expressions for $\varepsilon_{k,m}^{(n)}$ in (3.23)–(3.25) are just direct results of Theorem 1.

Note that when $x_n = 1$ and $y_n = n$ $(\forall n)$, the algorithm (3.21)–(3.22) is identical to the multistep ε-algorithm (3.1)–(3.2). If $m = 1$, the algorithm (3.21)–(3.22) reduces to the first generalization of the ε-algorithm (2.4)–(2.5) proposed by Brezinski. □

We see that the kernel of the generalized multistep Shanks transformation (3.23), i.e., the set of sequences that are transformed into a constant sequence, is given by

Theorem 3 *A necessary and sufficient condition that, for all* n, $\varepsilon_{k(m+1),m}^{(n)} = S$ *is that there exist constants* $a_1, \ldots, a_k$, $a_k \neq 0$, *such that, for all* n,

$$(S_n - S)x_n = a_1 R^m u_n + a_2 R^{2m} u_n + \cdots + a_k R^{km} u_n.$$

Lastly, we turn to consider the relationship of equation (3.21) to integrable systems. Setting $\left(a_{k-\frac{m-1}{2}}^{(n)}\right)^{-1} = \varepsilon_{k,m}^{(n+1)} - \varepsilon_{k,m}^{(n)}$, then from equation (3.21) we have

$$(x_{n+1})^m \prod_{i=0}^{m-1} a_{k-\frac{m-1}{2}+i}^{(n+1)} - (x_n)^m \prod_{i=0}^{m-1} a_{k-\frac{m-1}{2}+i}^{(n)} = \frac{1}{a_{k+\frac{m+1}{2}}^{(n)}} - \frac{1}{a_{k-\frac{m+1}{2}}^{(n+1)}}. \tag{3.26}$$

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

When $x_n = 1$ $(\forall n)$, equation (3.26) is identical to a discretization of the extended Lotka–Volterra equation [3]

$$\prod_{i=0}^{m-1} a^{(n+1)}_{k-\frac{m-1}{2}+i} - \prod_{i=0}^{m-1} a^{(n)}_{k-\frac{m-1}{2}+i} = \frac{1}{a^{(n)}_{k+\frac{m+1}{2}}} - \frac{1}{a^{(n+1)}_{k-\frac{m+1}{2}}}, \tag{3.27}$$

which can be derived from the multistep ε-algorithm (3.1)–(3.2). Thus we may consider equation (3.26) as the non-autonomous version of equation (3.27).

By Theorem 1 and equations (3.9)–(3.12), we obtain the molecule solution of (3.26):

$$\begin{aligned}
a^{(n)}_{k(m+1)+1-\frac{m-1}{2}} &= -\prod_{j=1}^{k+1} x_{n+j} \frac{H_{k+1}(Ru_n)H_{k+1}(Ru_{n+1})}{\Phi_{k+2}(Ru_n)\Phi_{k+1}(R^m u_{n+1})}, \\
a^{(n)}_{k(m+1)+2-\frac{m-1}{2}} &= \frac{1}{\prod_{j=0}^{k+2} x_{n+j}} \frac{\Phi_{k+2}(Ru_n)\Phi_{k+2}(Ru_{n+1})}{\Phi_{k+2}(R^2 u_n)H_{k+1}(Ru_{n+1})}, \\
a^{(n)}_{k(m+1)+i-\frac{m-1}{2}} &= \frac{1}{x_n} \frac{\Phi_{k+2}(R^{i-1}u_n)\Phi_{k+2}(R^{i-1}u_{n+1})}{\Phi_{k+2}(R^i u_n)\Phi_{k+2}(R^{i-2}u_{n+1})}, \quad i = 3, 4, \ldots, m, \\
a^{(n)}_{(k+1)(m+1)-\frac{m-1}{2}} &= \frac{1}{x_n} \frac{\Phi_{k+2}(R^m u_n)\Phi_{k+2}(R^m u_{n+1})}{H_{k+2}(Ru_n)\Phi_{k+2}(R^{m-1}u_{n+1})},
\end{aligned}$$

for $k = 0, 1, \ldots$, with initial values

$$a^{(n)}_{-m-\frac{m-1}{2}} = \infty, \quad a^{(n)}_{-m+1-\frac{m-1}{2}} = \cdots = a^{(n)}_{-1-\frac{m-1}{2}} = \frac{1}{x_n}, \quad a^{(n)}_{-\frac{m-1}{2}} = \frac{1}{\Delta S_n}.$$

4 A generalization of the convergence acceleration algorithm related to the lattice Boussinesq equation and a non-autonomous integrable system

It has been shown in [3] that when $m = 2$, the multistep ε-algorithm (3.1)–(3.2) is just the convergence acceleration algorithm related to the lattice Boussinesq equation [10]. Therefore we derive here a generalization of this latter algorithm from the generalized multistep ε-algorithm (3.21)–(3.22) by setting $m = 2$.

The generalization is given by

$$U^{(n)}_k = U^{(n+1)}_{k-3} + \frac{(x_n)^2}{(U^{(n+1)}_{k-2} - U^{(n)}_{k-2})(U^{(n+1)}_{k-1} - U^{(n)}_{k-1})}, \quad k = 1, 2, \ldots, \quad n = 0, 1, \ldots, \tag{4.1}$$

with initial conditions

$$U^{(n)}_{-2} = 0, \quad U^{(n)}_{-1} = \frac{y_n}{x_n}, \quad U^{(n)}_0 = S_n, \quad n = 0, 1, \ldots \tag{4.2}$$

The corresponding sequence transformations $\{T^{(n)}_k\}$ are

$$T^{(n)}_k = U^{(n)}_{3k}.$$

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

Table 1. *Numerical results of Example 1*

n	$\lvert S_n - S\rvert$	$\lvert T_1^{(n)} - S\rvert$	$\lvert T_2^{(n)} - S\rvert$	$\lvert T_3^{(n)} - S\rvert$	$\lvert T_4^{(n)} - S\rvert$
0	1.000	0.222	0.052	7.132×10^{-10}	3.044×10^{-10}
1	0.500	0.138	0.033	1.785×10^{-9}	
2	0.333	0.096	0.023	5.230×10^{-9}	
3	0.250	0.072	0.017	1.962×10^{-8}	
6	0.143	0.038	7.662×10^{-3}		
9	0.100	0.024			
12	0.077				

Notice that when $x_n = 1$ $(\forall n)$, equation (4.1) is the recursion rule of the convergence acceleration algorithm from the lattice Boussinesq equation, which can be seen as a simplified lattice Boussinesq equation. Therefore equation (4.1) can be seen as the non-autonomous simplified lattice Boussinesq equation.

Before presenting numerical examples, we show how to choose the initial values for $U_{-1}^{(n)} = y_n/x_n$ in (4.2). Recall that the sequence $\{y_n\}$ satisfies the relation (3.7). Then if $\{x_n\}$ is given, we can choose $U_{-1}^{(n)}$ to be

$$U_{-1}^{(n)} = \frac{y_n}{x_n} = \sum_{k=0}^{n-1} x_k.$$

We now perform some numerical experiments to test convergence acceleration of the algorithm (4.1)–(4.2).

Example 1. We consider the sequence

$$S_n = 1 + \frac{1}{n+1},$$

which converges to 1 as $n \to \infty$. Here we choose $x_n = \frac{1}{n+4}$, the corresponding transformation results are tabulated in Table 1.

Example 2. We consider the sequence

$$S_n = \sum_{k=1}^{n} \frac{(-1)^{k-1}}{k}$$

of partial sums of an alternating series, which converges to $S = \ln 2$. Here we choose $x_n = 1 + \frac{1}{n+4}$. The corresponding transformation results are given in Table 2.

Remark that similar to the first generalization of the ε-algorithm (2.5), for the new algorithm (3.21) the variable x_n is also just dependent on n and has no dependence on k. But this factor affects the acceleration properties of the new algorithm compared with the original multistep ε-algorithm (3.1). As it can be seen that [31] the multistep ε-algorithm (3.1) is effective for the linearly convergent sequences but not for the logarithmically convergent sequences. But from the above numerical results of example 1, we can see that

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

Table 2. *Numerical results of Example 2*

n	$\lvert S_n - S\rvert$	$\lvert T_1^{(n)} - S\rvert$	$\lvert T_2^{(n)} - S\rvert$	$\lvert T_3^{(n)} - S\rvert$	$\lvert T_4^{(n)} - S\rvert$	$\lvert T_5^{(n)} - S\rvert$
1	0.307	0.019	1.582×10^{-3}	1.396×10^{-4}	1.267×10^{-5}	1.163×10^{-6}
4	0.110	2.547×10^{-3}	1.038×10^{-4}	5.463×10^{-6}	3.311×10^{-7}	2.187×10^{-8}
7	0.066	8.025×10^{-4}	1.910×10^{-5}	6.427×10^{-7}	2.670×10^{-8}	1.276×10^{-9}
10	0.048	3.536×10^{-4}	5.515×10^{-6}	1.283×10^{-7}	3.851×10^{-9}	1.380×10^{-10}
13	0.037	1.868×10^{-4}	2.058×10^{-6}	3.503×10^{-8}	7.941×10^{-10}	
16	0.030	1.106×10^{-4}	9.073×10^{-7}	1.180×10^{-8}		
19	0.026	7.091×10^{-5}	4.500×10^{-7}			

the non-autonomous form of the original algorithm is effective for some logarithmically convergent sequences when the variable x_n is chosen appropriately.

5 Conclusions and discussions

In this article, we propose a new method to construct the non-autonomous form of discrete integrable systems that related to convergence acceleration algorithms. By the method, we can also obtain generalizations of the original convergence acceleration algorithms which involves an auxiliary sequence $\{x_n\}$. From the numerical examples given in Section 4, we can see that the sequence $\{x_n\}$ plays an important role when the algorithm applying to different kinds of sequences. How about the acceleration properties of the generalized algorithms for different choices of $\{x_n\}$? In addition, how about the convergence and stability analysis of the generalized algorithms as they are being applied to some important classes of linearly, logarithmically and hyperlinearly convergent sequences? These problems need further studies, and we will consider these problems in the future.

Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (Grant no. 11331008, Grant no. 11201469), the China Postdoctoral Science Foundation funded project (Grant no. 2012M510186, Grant no. 2013T60761), and the Hong Kong Research Grant Council (Grant no. GRF HKBU 202512).

References

[1] Aitken, A. C. (1965) *Determinants and Matrices*, Oliver and Boyd, Edinburgh and London.
[2] Brezinski, C. (1972) Conditions d'application et de convergence de procédés d'extrapolation. *Numer. Math.* **20**(1), 64–79.
[3] Brezinski, C., He, Y., Hu, X. B., Redivo-Zaglia, M. & Sun, J. Q. (2012) Multistep ε-algorithm, Shanks' transformation, and the Lotka–Volterra system by Hirota's method. *Math. Comp.* **81**(279), 1527–1549.

https://doi.org/10.1017/S0956792515000479 Published online by Cambridge University Press

[4] Brezinski, C., He, Y., Hu, X. B. & Sun, J. Q. (2010) A generalization of the G-transformation and the related algorithms. *Appl. Numer. Math.* **60**(12), 1221–1230.
[5] Brezinski, C. & Redivo-Zaglia, M. (1991) *Extrapolation Methods: Theory and Practice*, North-Holland, Amsterdam.
[6] Brualdi, R. A. & Schneider, H. (1983) Determinantal identities: Gauss, Schur, Cauchy, Sylvester, Kronecker, Jacobi, Binet, Laplace, Muir and Cayley. *Linear Algebra Appl.* **52/53**, 769–791.
[7] Gilson, C. R. (2001) Generalizing of the KP hierarchies: Pfaffian hierarchies. *Theor. Math. Phys.* **133**(3), 1663–1674.
[8] Gilson, C. R. & Nimmo, J. J. C. (2001) Pfaffianization of the Davey-Stewartson equation. *Theor. Math. Phys.* **128**(1), 870–882.
[9] Gilson, C. R., Nimmo, J. J. C. & Tsujimoto, S. (2001) Pfaffianization of the discrete KP equation. *J. Phys. A: Math. Gen.* **34**(48), 10569–10575.
[10] He, Y., Hu, X. B., Sun, J. Q. & Weniger, E. J. (2011) Convergence acceleration algorithm via an equation related to the lattice Boussinesq equation. *SIAM J. Sci. Comput.* **33**(3), 1234–1245.
[11] He, Y., Hu, X. B. & Tam, H. T. (2009) A q-difference version of the ε-algorithm. *J. Phys. A: Math. Theor.* **42**(9), 095202.
[12] Hirota, R. (1981) Discrete analogue of a generalized Toda equation. *J. Phys. Soc. Jpn.* **50**(11), 3785–3791.
[13] Hirota, R. (2004) *Direct Method in Soliton Theory*, Cambridge University Press, Cambridge.
[14] Hirota, R. & Ohta, Y. (1991) Hierarchies of coupled soliton equations. I. *J. Phys. Soc. Jpn.* **60**(3), 798–809.
[15] Hu, X. B. & Wang, H. Y. (2006) Construction of dKP and BKP equations with self-consistent sources. *Inverse Problems* **22**(3), 1903–1920.
[16] Hu, X. B. & Wang, H. Y. (2007) New type of Kadomtsev-Petviashvili equation with self-consistent sources and its bilinear Bäcklund transformation. *Inverse Problems* **23**(4), 1433–1444.
[17] Hu, X. B., Zhao, J. X. & Tam, H. W. (2004) Pfaffianization of the two-dimensional Toda lattice. *J. Math. Anal. Appl.* **296**(1), 256–261.
[18] Iwasaki, M. & Nakamura, Y. (2002) On the convergence of a solution of the discrete Lotka-Volterra system. *Inverse Problems* **18**(6), 1569–1578.
[19] Iwasaki, M. & Nakamura, Y. (2004) An application of the discrete Lotka-Volterra system with variable step-size to singular value computation. *Inverse Problems* **20**(2), 553–563.
[20] Minesaki, Y. & Nakamura, Y. (2001) The discrete relativistic Toda molecule equation and a Padé approximation algorithm. *Numer. Algorithms* **27**(3), 219–235.
[21] Nagai, A. & Satsuma, J. (1995) Discrete soliton equations and convergence acceleration algorithms. *Phys. Lett. A* **209**(5–6), 305–312.
[22] Nagai, A., Tokihiro, T. & Satsuma, J. (1998) The Toda molecule equation and the ε-algorithm. *Math. Comp.* **67**(224), 1565–1575.
[23] Nakamura, Y. ed. (2000) *Applied Integrable Systems (in Japanese)*, Syokabo, Tokyo.
[24] Nakamura, Y. (1999) Calculating Laplace transforms in terms of the Toda molecule. *SIAM J. Sci. Comput.* **20**(1), 306–317.
[25] Ohta, Y., Nimmo, J. J. C. & Gilson, C. R. (2001) A bilinear approach to a Pfaffian self-dual Yang-Mills equation. *Glasg. Math. J.* **43**A, 99–108.
[26] Papageorgiou, V., Grammaticos, B. & Ramani, A. (1993) Integrable lattices and convergence acceleration algorithms. *Phys. Letters A* **179**(2), 111–115.
[27] Rutishauser, H. (1954) Der Quotienten-Differenzen-Algorithmus. *Z. Angew. Math. Pysik.* **5**(3), 233–251.
[28] Shanks, D. (1949) An analogy between transient and mathematical sequences and some nonlinear sequence-to-sequence transforms suggested by it. Part I. *Memorandum 9994,* Naval Ordnance Laboratory, White Oak.

[29] Shanks, D. (1955) Non linear transformations of divergent and slowly convergent sequences. *J. Math. Phys.* **34**, 1–42.
[30] Sidi, A. (2003) *Practical Extrapolation Methods. Theory and Applications*, Cambridge University Press, Cambridge.
[31] Sun, J. Q., Chang, X. K., He, Y. & Hu, X. B. (2013) An extended multistep shanks transformation and convergence acceleration algorithm with their convergence and stability analysis. *Numer. Math.* **125**(4), 785–809.
[32] Symes, W. W. (1981/1982) The QR algorithm and scattering for the nonperiodic Toda lattice. *Phys. D* **4**(2), 275–280.
[33] Tsujimoto, S., Nakamura, Y. & Iwasaki, M. (2001) The discrete Lotka-Volterra system computes singular values. *Inverse Problems* **17**(1), 53–58.
[34] Walz, G. (1996) *Asymptotics and Extrapolation*, Akademie Verlag, Berlin.
[35] Wang, H. Y., Hu, X. B. & Tam, H. W. (2007) A 2+1-dimensional Sasa-Satsuma equation with self-consistent sources. *J. Phys. Soc. Jpn.* **76**(2), 024007.
[36] Weniger, E. J. (1989) Nonlinear sequence transformations for the acceleration of convergence and the summation of divergent series. *Comp. Phys. Reports* **10**(5–6), 189–371.
[37] Wimp, J. (1981) *Sequence Transformations and Their Applications*, Academic Press, New York.
[38] Wynn, P. (1956) On a device for computing the $e_m(S_n)$ transformation. *MTAC* **10**(54), 91–96.
[39] Wynn, P. (1956) On a procrustean technique for the numerical transformation of slowly convergent sequences and series. *Proc. Cambridge Phil. Soc.* **52**(4), 663–671.
[40] Zhao, J. X., Li, C. X. & Hu, X. B. (2004) Pfaffianization of the differential-difference KP equation. *J. Phys. Soc. Jpn.* **73**(5), 1159–1162.