Symbolic computation of Jacobi elliptic function solutions to nonlinear differential-difference equations

Xuelin Yong\(^a\), Xin Zeng\(^b\), Zhiyong Zhang\(^c\), Yufu Chen\(^c\)
\(^a\) Department of Mathematics and Physics, North China Electric Power University, Beijing, 102206, China
\(^b\) Department of Mathematics, Zhengzhou University, Zhengzhou, 450052, China
\(^c\) School of Mathematical Science, Graduate University of Chinese Academy of Sciences, Beijing, 100049, China

A R T I C L E   I N F O

Article history:
Received 17 September 2007
Received in revised form 17 November 2007
Accepted 12 January 2009

Keywords:
Jacobi elliptic functions
Exact solutions
Differential-difference equations
Symbolic computation

A B S T R A C T

In this paper, an algorithm is presented to find exact polynomial solutions of nonlinear differential-difference equations (DDEs) in terms of the Jacobi elliptic functions. The key steps of the algorithm are illustrated by the discretized mKdV lattice. A Maple package \textsc{JACOBI} is developed based on the algorithm to automatically compute special solutions of nonlinear DDEs. The effectiveness of the package is demonstrated by applying it to a variety of equations.

© 2009 Elsevier Ltd. All rights reserved.

1. Introduction

Since the work of Fermi, Pasta, and Ulam in the 1950s \cite{1}, the investigation of exact solutions of the nonlinear differential-difference equations (DDEs) has played a crucial role in the modeling of many physical phenomena, such as particle vibrations in lattices, currents in electrical networks, pulses in biological chains, etc. Moreover, DDEs play an important role in the numerical simulations of nonlinear partial differential equations, queuing problems, discretions in solid state and quantum physics. Unlike difference equations, which are fully discretized, DDEs are semi-discretized with time usually kept continuous. They are hard to solve and only a few techniques exist, for example, the inverse scattering method \cite{2,3}, the Bäcklund transformation method \cite{4}, symmetries method \cite{5}, Hirota’s bilinear method \cite{6}, the multilinear variable separation approach \cite{7} and so on.

Due to the availability of symbolic manipulation programs such as Reduce, Mathematica and Maple, which allow the user to perform the tedious algebra and routine computations, the search for special exact solutions of nonlinear DDEs becomes more and more attractive. Recently, Baldwin et al. presented an algorithm to find exact solutions of DDEs in terms of tanh function \cite{8}. A Mathematica package \textit{DDESpecialSolutions.m} is designed by them. On the other hand, the Jacobi elliptic function expansion method plays an important role in seeking period solutions of nonlinear wave equations. Motivated by the idea, Dai \cite{9,10} and Xu \cite{11} extended the method to derive explicit periodic wave solutions for nonlinear DDEs. However, it is very difficult to complete this procedure by hand, especially when the discrete variables occur. Consequently, the aim of this paper is to develop a fully automated software package which can compute Jacobi elliptic function solutions of DDEs automatically. A complete implementation package \textit{JACOBI.mws} written in Maple will be presented in detail.

The paper is organized as follows. In Section 2, we present the algorithm with a selected example of discretized mKdV lattice. In Section 3, we describe our package \textit{JACOBI.mws}, in which the operator \texttt{Jasc} can output exact Jacobi elliptic function solutions...
2. Outline of the algorithm

In this section, we outline the steps of the algorithm and illustrate each step by the discretized mKdV lattice.

Given a system of polynomial DDEs, say, one discrete variable \( n \),

\[
\Delta(u_{n+p_1}(x), \ldots, u_{n+p_m}(x), u_{n+p_1}'(x), \ldots, u_{n+p_m}'(x), u_{n+p_1}^{(r)}(x), \ldots, u_{n+p_m}^{(r)}(x)) = 0,
\]

where the dependent variable \( u_i \) have \( M \) components \( u_{i,p} \), the continuous variable \( x \) has \( N \) components \( x_i \), the \( m \) shift vectors \( p_i \in \mathbb{Z} \), and \( u^{(r)}(x) \) denotes the collection of mixed derivative terms of order \( r \). For the simplicity of notation, usually we denote the dependent variables by \( u, v, w, \ldots \) instead of \( u_{1,n}, u_{2,n}, u_{3,n}, \ldots \), and the independent variables by \( x, y, t, \ldots \) instead of \( x_1, x_2, \ldots \).

**Example:** The discretized mKdV lattice [4] can be written as:

\[
\frac{du_i}{dt} = (\alpha - u_i^2)(u_{i+1} - u_{i-1}),
\]

where the two-component Volterra equation [8] reads

\[
\frac{du_i}{dt} = u_i(u_{i-1} - u_{i+1}), \quad \frac{dv_i}{dt} = v_i(u_{i+1} - u_{i}).
\]

**Step 1:** Transform the original DDEs into nonlinear ODEs.

We introduce the transformation \( u_{n+p}(x) = \Phi(\xi_{n+p}) = (\Phi_1(\xi_{n+p}), \Phi_2(\xi_{n+p}), \ldots, \Phi_M(\xi_{n+p})), \Phi_{n+p} = \sum_{i=1}^{N} k_i \xi_i + k_{N+1}(n + p) \) for any \( s = 1, \ldots, m \), where the coefficients \( k_1, k_2, \ldots, k_{N+1} \) are all nonzero constants. In this way, Eq. (1) becomes

\[
\Delta(\Phi(\xi_{n+p}), \ldots, \Phi(\xi_{n+p}), \ldots, \Phi'(\xi_{n+p}), \ldots, \Phi'(\xi_{n+p}), \ldots, \Phi^{(r)}(\xi_{n+p}), \ldots, \Phi^{(r)}(\xi_{n+p})) = 0.
\]

**Example:** For the discretized mKdV lattice, let \( u_{n+i} = \Phi(\xi_{n+i}), \xi_{n+i} = k_1 t + k_2 (d + i)(i = -1, 0, 1) \), we obtain

\[
k_1 \Phi' (\xi_{n+i}) = (\alpha - \Phi^2 (\xi_{n+i})) (\Phi (\xi_{n+i+1}) - \Phi (\xi_{n+i-1})).
\]

**Step 2:** Make the ansätze and determine the degrees of the polynomial solutions.

We seek solutions of the polynomial form

\[
\Phi_1(\xi_{n+p}) = a_0 + \sum_{j=1}^{L_1} [a_j \text{sn}^j(\xi_{n+p}) + b_j \text{sn}(\xi_{n+p})^{j-1} \text{cn}(\xi_{n+p})],
\]

where \( a_0, a_i, b_i \ (i = 1, \ldots M, j = 1, \ldots, L_i) \) are constants to be determined and \( L_i \) are fixed according to balancing the terms of highest degree as follows.

First of all, it is important to note that

\[
\text{sn} (\xi_{n+p}) = \frac{\text{sn} (\xi) \text{cn} (k_{N+1} p) \text{dn} (k_{N+1} p) + \text{sn} (k_{N+1} p) \text{cn} (\xi) \text{dn} (\xi)}{1 - m^2 \text{sn}^2 (\xi) \text{sn}^2 (k_{N+1} p)},
\]

and \( m \) is the Jacobi elliptic modulus. By means of the relations

\[
\text{cn}^2 (\xi) = 1 - \text{sn}^2 (\xi), \quad \text{dn}^2 (\xi) = 1 - m^2 \text{sn}^2 (\xi),
\]

we can say \( \Phi_1(\xi_{n+p}) \) is of highest degree \( L_1 \) in \( \text{sn}(\xi_{n+p}) \) and is of highest degree zero in \( \text{sn}(\xi) \) if \( p_i \neq 0 \). This also implies that \( \Phi_1(\xi_{n+p}) \) is of highest degree zero in \( \text{sn}(\xi_{n+p}) \) if \( p_i \neq 0 \) since \( n + p_s = n + p_r + p_s - p_r = n' + p_s \).

Furthermore,

\[
\Phi'_1(\xi_{n+p}) = \sum_{j=1}^{L_1} [a_j \text{sn}^{j-1} (\xi_{n+p}) \text{cn} (\xi_{n+p}) \text{dn} (\xi_{n+p}) - b_j \text{sn}^j (\xi_{n+p}) \text{dn} (\xi_{n+p})] + b_j (j - 1) \text{sn}^{j-2} (\xi_{n+p}) \text{cn}^2 (\xi_{n+p}) \text{dn} (\xi_{n+p}).
\]
so we can take the highest degree of $\Phi^J(\xi_{n+\beta})$ as $L_i + 1$ in $\text{sn}(\xi_{n+\beta})$ in view of Eq. (7). In the same way, the highest degree of $\Phi^J(\xi_{n+\beta})$ is taken as $L_i + 2$, and in general, the highest degree of $\Phi^J(\xi_{n+\beta})$ is taken as $\alpha L_i + \beta$ in $\text{sn}(\xi_{n+\beta})$.

After substituting $\Phi(\xi_{n+\beta})$ into Eq. (4), the coefficients of like power of $\text{sn}(\xi_{n+\beta})$ in every equation must vanish. In particular, the highest degree must equal zero. However, the terms with shifts other than $p_i$, say $p_j$, will not effect the highest degree since $\Phi(\xi_{n+\beta})$ can be interpreted as being of the highest degree zero in $\text{sn}(\xi_{n+\beta})$. Therefore, if we are interested in balancing terms with shift $p_i$, it suffices to equate every two possible highest exponents in every component. It generates a linear system for determining the $L_i$. Solving the linear system one can get $L_i$ or some relations between them. If one or more exponents $L_i$ remain undetermined, we assign a strictly positive integer value to the free $L_i$ to avoid the trivial cases.

**Example:** For Eq. (5), we have the maximal exponents are $L_1 + 1$ (from the $\Phi^J(\xi_{n})$ term) and $2L_1$ (from the $\Phi^J(\xi_{n+1}) - \Phi(\xi_{n-1})$ term). Therefore, we get the linear system $L_1 + 1 = 2L_1$. Then $L_1 = 1$, and Eq. (6) becomes

$$
\Phi(\xi_{n+i}) = a_{10} + a_{11}\text{sn}(\xi_{n+i}) + b_{11}\text{cn}(\xi_{n+i}),
$$

for $i = -1, 0, 1$.

**Step 3:** Derive the algebraic system for the coefficients.

To generate the system for the unknown coefficients, substitute Eq. (6) into Eq. (4) and do some simplifications according to Eqs. (7)–(9). Then we get a rational polynomial system. Take the numerators and set the coefficients of all powers like $\text{sn}^m(\xi_{n})$ and $\text{sn}^m(\xi_{n})\text{cn}(\xi_{n})\text{dn}(\xi_{n})$ to zero, we will get an algebraic system. In order to avoid the trivial solutions, it demands $k_1k_2 \cdots k_{n+1} \neq 0$, and $a_{1i}^2 + b_{1i}^2 \neq 0$ for $i = 1, \ldots, M$. These constraints and the algebraic system together determine the unknowns.

**Example:** For the discretized mKdV lattice, now we have

$$
k_1k_2(a_{11}^2 + b_{11}^2) \neq 0,$$

$$
-4a_{10}b_{11}\text{sn}(k_2, m) = 0,$$

$$
4a_{10}b_{11}\text{sn}(k_2, m)\text{dn}(k_2, m) + 1 = 0,$$

$$
4a_{10}\text{sn}(k_2, m)(b_{11}^2\text{dn}(k_2, m) - a_{11}^2) = 0,$$

$$
2\text{sn}(k_2, m)\text{dn}(k_2, m) + b_{11}^2 - 4b_{11}\text{sn}(k_2, m) + a_{11}^2 + b_{11}k_1 = 0,$$

$$
-2\alpha a_{11}\text{dn}(k_2, m) + a_{11}k_1 - 2b_{11}\text{sn}(k_2, m) + a_{11}^2 = 0,$$

$$
a_{11}(2\alpha^2\text{sn}(k_2, m) + 4b_{11}\text{sn}(k_2, m) + a_{11}^2 + b_{11}) = 0,$$

$$
-2b_{11}\text{sn}(k_2, m)(2b_{11}^2\text{dn}(k_2, m) + k_1^2\text{sn}(k_2, m) - 2\text{dn}(k_2, m) + a_{11}^2 - 4a_{11}^2) = 0.
$$

**Step 4:** Solve the nonlinear algebraic system and build the solutions.

Solving the nonlinear algebraic system obtained in **Step 3**, then substitute the obtained solutions in Eq. (6) and reverse **Step 1** to obtain the explicit solutions in the original variables. Then, test the solutions by substitution into Eq. (1).

**Example:** Eq. (12) have two different solutions,

$$
a_{11} = \pm m\sqrt{\alpha}\text{sn}(k_2, m), \quad a_{10} = b_{11} = 0, \quad k_1 = 2\alpha\text{sn}(k_2, m), \quad k_2 = k_2,
$$

and

$$
a_{10} = a_{11} = 0, \quad k_1 = \frac{2\alpha\text{sn}(k_2, m)}{\text{dn}(k_2, m)}, \quad b_{11} = \pm m\sqrt{\alpha}\frac{\text{sn}(k_2, m)}{\text{dn}(k_2, m)}, \quad k_2 = k_2.
$$

Therefore, the elliptic function solutions of Eq. (2) are

$$
u_n(t) = \pm m\sqrt{\alpha}\text{sn}(k_2, m)\text{sn}[2\alpha\text{sn}(k_2, m)t + 2k_2n, m],
$$

$$
u_n(t) = \pm m\sqrt{\alpha}\frac{\text{sn}(k_2, m)}{\text{dn}(k_2, m)}\text{cn}\left[\frac{2\alpha\text{sn}(k_2, m)}{\text{dn}(k_2, m)}t + 2k_2n, m\right].
$$

When $m \to 1$, the solutions Eq. (15) degenerate as

$$
u_n(t) = \pm \frac{\sqrt{\alpha}}{\text{tanh}(k_2)\text{tanh}[2\alpha\text{tanh}(k_2)t + k_2n]},
$$

which are the kink-type excitation of the discretized mKdV lattice in [8,9]. The solutions Eq. (16) degenerate as

$$
u_n(t) = \pm \sqrt{\alpha}\text{sech}[2\alpha\text{sech}(k_2)t + k_2n],
$$

which are the bell-type solitary wave solutions given in [10].
3. Description of *JACOBI*

Now we present the program *JACOBI.mws* that was developed based on the steps in Section 2 using the computer algebra system Maple. In our program, the following operators are available. *(Note: The dependent variables $u_{n+p}(x)$ are input as $u(x, n + p)$, and the Jacobi elliptic functions are output in the familiar forms $sn, cn, dn$ rather than the complicated expressions $JacobiSN, JacobiCN, JacobiDN$ in Maple. The expressions after the prompt “* > ” are input ones, the rest are output results.)*

3.1. **init**

This operator is used to load the available packages used in our package.

3.2. **Normal**

This operator is used to convert the input as a list whose elements having the vanishing right parts. It is used with the following syntax: **Normal(eqs)**. The input can be some equations or a list. The output is a list.

**Example:**
```maple
DmKdV := diff(u(t, n), t) = (alpha - u(t, n)^2) * (u(t, n + 1) - u(t, n - 1));
Normal(DmKdV);
```

3.3. **Extract**

This operator is used to decompose the input system from which the variables contained in each required function, the parameters contained in the input system, and the required function names are separated respectively. It is used with the syntax: **Extract(eqs)**. The output is a list with two or three elements depending on whether there are some parameters of the input system.

**Example:**
```maple
Extract(DmKdV);
```

3.4. **Tran**

This operator is used to convert the simplified system obtained in **Normal** into an ordinary differential-difference system. It is used with the syntax: **Tran(eqs)**. The output is also a list.

**Example:**
```maple
Tran(DmKdV);
```

3.5. **Balcon**

The operator **Balcon** is used to determine degrees of the required solutions. There may be several sets of values, which correspond to several different types of solutions. It is used with the syntax: **Balcon(eqs)**. We must use the operator **Tran** before using this operator. The output is a set.

**Example:**
```maple
Balcon(DmKdV);
```

3.6. **Jasc**

This operator includes all the calculations in **Step 3** and **Step 4**. Firstly it produces the algebraic system for the unknowns, and then solves it. If there is no result, it will deliver the information “Sorry, there isn’t any solution”. Otherwise, build the solution in the original variables and test it. Then output the solutions in a sequence of lists. This operator is used with the syntax: **Jasc(eqs)**.
Example: Consider the following lattice equation introduced by Wadati [12]

\[ u = \sqrt{\alpha} \text{sn} (k_2, m) \text{sn} (\xi_n, m), \xi_n = 2 \alpha \text{sn} (k_2, m) t + k_2 n \]

\[ u = -\sqrt{\alpha} \text{sn} (k_2, m) \text{sn} (\xi_n, m), \xi_n = 2 \alpha \text{sn} (k_2, m) t + k_2 n \]

\[ u = \text{cn} (\xi_n, m) \sqrt{\frac{\alpha}{-\text{dn} (k_2, m)^2}} \text{msn} (k_2, m), \xi_n = \frac{2 \alpha \text{sn} (k_2, m) t}{\text{dn} (k_2, m)} + k_2 n \]

\[ u = -\text{cn} (\xi_n, m) \sqrt{\frac{\alpha}{-\text{dn} (k_2, m)^2}} \text{msn} (k_2, m), \xi_n = \frac{2 \alpha \text{sn} (k_2, m) t}{\text{dn} (k_2, m)} + k_2 n \].

4. Examples

In this section, we shall give some examples to explain the applications and effectiveness of our package.

Example 1. Consider the following lattice equation introduced by Wadati [12]

\[ \frac{du_n}{dt} = (\alpha + \beta u_n + \gamma u_n^3)(u_{n-1} - u_{n+1}). \]  

where \(\alpha, \beta\) and \(\gamma \neq 0\) are constants. Eq. (19) obviously contains the following famous DDEs, namely,

(1) Hybrid lattice [13]

\[ \frac{du_n}{dt} = (1 + \alpha u_n + \beta u_n^2)(u_{n-1} - u_{n+1}). \]

(2) Discretized mKdV lattice [8]

\[ \frac{du_n}{dt} = (\alpha - u_n^2)(u_{n-1} - u_{n+1}). \]

(3) Modified Volterra lattice [14]

\[ \frac{du_n}{dt} = u_n^2(u_{n-1} - u_{n+1}). \]

In [8], one can find the solutions of Eqs. (20) and (21) which are polynomial in the hyperbolic tangent function. Eq. (22) is a very well studied integrable model. It is a bi-Hamiltonian, possesses an L–A pair, recursion operator, local master-symmetry, and an infinite hierarchy of higher symmetries and conservation laws [14].

For this equation, the package JACOBI.mws is used as follows:

>HL := diff(u (t, n), t) = (alpha + beta * u (t, n) + gamma * u (t, n)^3) * (u (t, n-1) - u (t, n+1));

>SC(HL);

\[ u (n, t) = -\frac{\beta}{2} + \frac{1}{2} \sqrt{\beta^2 - 4 \alpha \gamma} \text{sn} (k_2, m) \text{sn} (\xi_n, m), \xi_n = -\frac{1}{2} \frac{\text{sn} (k_2, m) (-\beta^2 + 4 \alpha \gamma) t}{\gamma} + k_2 n \]

\[ u (n, t) = -\frac{\beta}{2} + \frac{1}{2} \frac{\text{cn} (\xi_n, m) \sqrt{-\beta^2 + 4 \alpha \gamma}}{\text{dn} (k_2, m)^2} \text{sn} (k_2, m)m, \xi_n = -\frac{1}{2} \frac{\text{sn} (k_2, m) (-\beta^2 + 4 \alpha \gamma) t}{\text{dn} (k_2, m) \gamma} + k_2 n \]

\[ u (n, t) = -\frac{\beta}{2} - \frac{1}{2} \sqrt{\beta^2 - 4 \alpha \gamma} \text{sn} (k_2, m) \text{sn} (\xi_n, m), \xi_n = -\frac{1}{2} \frac{\text{sn} (k_2, m) (-\beta^2 + 4 \alpha \gamma) t}{\gamma} + k_2 n \]

\[ u (n, t) = -\frac{\beta}{2} - \frac{1}{2} \frac{\text{cn} (\xi_n, m) \sqrt{-\beta^2 + 4 \alpha \gamma}}{\text{dn} (k_2, m)^2} \text{sn} (k_2, m)m, \xi_n = -\frac{1}{2} \frac{\text{sn} (k_2, m) (-\beta^2 + 4 \alpha \gamma) t}{\text{dn} (k_2, m) \gamma} + k_2 n \].

When \(m \to 1\) the solutions degenerate as

\[ u_n = -\frac{1}{2} \frac{\beta}{\gamma} + \frac{\sqrt{\beta^2 - 4 \alpha \gamma}}{\gamma} \tanh(k_2) \tanh \left[k_2 n - \frac{1}{2} \frac{\tanh (k_2) (-\beta^2 + 4 \alpha \gamma) t}{\gamma}\right]. \]
which coincide with the results given in [8], and when $m \to 0$ they degenerate as

$$u_n = \frac{1}{2} \beta \pm \frac{\sqrt{\beta^2 - 4 \alpha} \gamma}{\gamma} \sin(k_2) \sin \left[ k_2 n - \frac{1}{2} \sin(k_2) \left( -\beta^2 + 4 \alpha \gamma \right) t \right],$$

(24)

which haven't been obtained before.

**Example 2.** The semi-discrete coupled mKdV system [15]

\[
\begin{align*}
\frac{du_n}{dt} &= (1 + \alpha u_n^2 + \beta u_n v_n + \gamma v_n^2) (u_{n-1} - u_{n+1}), \\
\frac{dv_n}{dt} &= (1 + \alpha u_n^2 + \beta u_n v_n + \gamma v_n^2) (v_{n-1} - v_{n+1}),
\end{align*}
\]

(25)

which is a special case of the general form

\[
\frac{du_{i,n}}{dt} = \left( 1 + \sum_{j,k=1}^{M} c_{jk} u_{i,n} u_{k,n} \right) (u_{i,n+1} - u_{i,n-1}), \quad i = 1, 2, \ldots, M
\]

(26)

investigated in [16].

For the semi-discrete coupled mKdV system, the package JACOBI.mws is used as follows:

\[
\begin{align*}
&D\text{CMKdV} := \text{diff}(u(t,n), t) = (1 + \alpha u(t,n)^2 + \beta u(t,n) v(t,n) + \gamma (v(t,n))^2) (u(t,n+1) - u(t,n-1)), \\
&\text{diff}(v(t,n), t) = (1 + \alpha u(t,n)^2 + \beta u(t,n) v(t,n) + \gamma (v(t,n))^2) (v(t,n+1) - v(t,n-1)).
\end{align*}
\]

Several solutions are output within 25 seconds:

1. When $\alpha \neq 0$,

\[
\begin{align*}
&u(t,n) = -\beta a_{21} \pm \sqrt{\beta^2 a_{21}^2 - 4 \alpha \gamma a_{21}^2 - 4 \alpha + 4 \alpha \gamma^2 (k_2, m)} \text{sn}[2 \text{sn}(k_2, m) t + k_2 n, m], \\
v(t,n) = a_{21} \text{sn}[2 \text{sn}(k_2, m) t + k_2 n, m],
\end{align*}
\]

(27)

(28)

where $\beta, \gamma, k_2, a_{21}$ are arbitrary. They are also discovered in [15].

2. When $\alpha, \gamma \neq 0$ satisfy the constraint $4 \alpha \gamma - \beta^2 \neq 0$,

\[
\begin{align*}
&u(t,n) = -2 \gamma \sqrt{-1 + (1 + \gamma b_{21}^2) \text{dn}^2(k_2, m)} \text{sn}[2 \gamma b_{21}^2 + 2 \text{sn}(k_2, m) t + k_2 n, m], \\
v(t,n) = \pm \beta \sqrt{-1 + (1 + \gamma b_{21}^2) \text{dn}^2(k_2, m)} \text{sn}[2 \gamma b_{21}^2 + 2 \text{sn}(k_2, m) t + k_2 n, m] + b_{21} \text{cn}[2 \gamma b_{21}^2 + 2 \text{sn}(k_2, m) t + k_2 n, m],
\end{align*}
\]

(29)

(30)

where $b_{21}, k_2$ are arbitrary. Here the expressions of $v(t,n)$ are different from case 1 and are nontrivial. When $m \to 0$ they degenerate as

\[
\begin{align*}
&u(t,n) = -2 \gamma \sqrt{b_{21}^2 \frac{1}{4 \gamma^2 \alpha - \beta^2}} \sin \left[ (2 \gamma b_{21}^2 + 2) \sin(k_2) t + k_2 n \right], \\
v(t,n) = \pm \beta \sqrt{b_{21}^2 \frac{1}{4 \gamma^2 \alpha - \beta^2}} \sin \left[ (2 \gamma b_{21}^2 + 2) \sin(k_2) t + k_2 n \right] + b_{21} \cos \left[ (2 \gamma b_{21}^2 + 2) \sin(k_2) t + k_2 n \right],
\end{align*}
\]

(31)

(32)

which haven't been obtained before.

3. When $\beta \gamma \neq 0$,

\[
\begin{align*}
&u(t,n) = -\frac{2b_{21} \gamma}{\beta} \text{cn} \left[ -2 \text{sn}(k_2, m) \text{dn}(k_2, m) \left( -4 \alpha b_{21}^2 \gamma^2 - \beta^2 + \beta^2 b_{21}^2 \gamma \right) t + k_2 n, m \right],
\end{align*}
\]

(33)
\[ v(t, n) = b_{21} \text{cn} \left[ \frac{-2 \sin(k_2, m) \text{dn}(k_2, m) \left( -4 \alpha b_{21}^2 \gamma^2 - \beta^2 + \beta^2 b_{21}^2 \gamma \right) t}{\beta^2} + k_2 n, m \right] + \sqrt{\frac{4 \gamma \text{dn}^2(k_2, m) \alpha^2 b_{21}^2}{\beta^2} + \frac{\text{dn}^2(k_2, m) - 1}{\gamma}} - \frac{\text{dn}^2(k_2, m) b_{21}^2}{\gamma} \times \text{sn} \left[ \frac{-2 \sin(k_2, m) \text{dn}(k_2, m) \left( -4 \alpha b_{21}^2 \gamma^2 - \beta^2 + \beta^2 b_{21}^2 \gamma \right) t}{\beta^2} + k_2 n, m \right], \] (34)

where \( \alpha, b_{21}, k_2 \) are arbitrary. When \( m \to 0, \alpha = 0 \) they degenerate as
\[ u(t, n) = -\frac{2b_{21} \gamma}{\beta} \cos[2 \sin(k_2) (1 - \beta^2 b_{21}^2 \gamma) t + k_2 n]. \] (35)
\[ v(t, n) = b_{21} \cos[2 \sin(k_2) (1 - \beta^2 b_{21}^2 \gamma) t + k_2 n] + ib_{21} \sin \left[ 2 \sin(k_2) \left( 1 - \beta^2 b_{21}^2 \gamma \right) t + k_2 n \right], \] (36)

where \( i^2 = -1, b_{21} > 0. \)

4. When \( \beta, \gamma, m, k_2 \) satisfy \( \beta \gamma \text{msn}(k_2, m) \neq 0, \)
\[ u(n, t) = -2 \frac{b_{21} \gamma}{\beta} \text{cn} \left[ \frac{8 \alpha b_{21}^2 \gamma^2 - 2 \beta^2 b_{21}^2 \gamma}{\beta^2 \text{sn}(k_2, m) m^2} + k_2 n, m \right], \] (37)
\[ v(n, t) = \frac{1}{\beta \text{msn}(k_2, m)} \sqrt{\frac{4 \alpha b_{21}^2 \gamma^2 + \beta^2 - \beta^2 b_{21}^2 \gamma}{\gamma} \text{dn}(k_2, m) - \beta^2} + b_{21} \text{cn} \left[ \frac{8 \alpha b_{21}^2 \gamma^2 - 2 \beta^2 b_{21}^2 \gamma}{\beta^2 \text{sn}(k_2, m) m^2} + k_2 n, m \right], \] (38)

where \( \alpha, b_{21} \) are arbitrary. When \( m \to 1, \alpha = 0 \) they degenerate as the solitary wave
\[ u(n, t) = -2 \frac{b_{21} \gamma}{\beta} \text{sech}[2 b_{21}^2 \gamma \sinh(k_2) t - k_2 n], \] (39)
\[ v(n, t) = \frac{1}{\beta \text{tanh}(k_2)} \sqrt{\frac{\left( \beta^2 - \beta^2 b_{21}^2 \gamma \right) \text{sech}^2(k_2) - \beta^2}{\gamma}} + b_{21} \text{sech}[2 b_{21}^2 \gamma \sinh(k_2) t - k_2 n]. \] (40)

5. Discussions

In this paper, we presented a straightforward algorithm to compute special solutions of nonlinear DDEs without using explicit integration. We also designed the symbolic package to find traveling wave solutions of nonlinear DDEs involving the Jacobi elliptic functions. Two discrete systems are investigated as illustrative examples to show the application of the package. While the software can produce some solutions for many equations, there is no guarantee that the code will compute the complete solution set of all polynomial solutions involving the Jacobi elliptic functions, especially when the DDEs have parameters. This is due to the restrictions on the form of the solutions and the limitations of the algebraic solver. Whereas we believe that this method can be applied to other DDEs and further work to improve the efficiencies of the package is worthy of study.

Acknowledgements

We gratefully thank the referees for their valuable comments. The first author is supported by Natural Science Foundation of China under the Grant Number 10726063.

References


