Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Symbolic Computation 37 (2004) 669–705
Journal of Symbolic Computation
www.elsevier.com/locate/jsc

Symbolic computation of exact solutions expressible in hyperbolic and elliptic functions for nonlinear PDEs

D. Baldwin[a], Ü. Göktaş[b], W. Hereman[a,c,*], L. Hong[d], R.S. Martino[a], J.C. Miller[e]

[a]*Department of Mathematical and Computer Sciences, Colorado School of Mines, Golden, CO 80401-1887, USA*
[b]*Wolfram Research, Inc., 100 Trade Center Drive, Champaign, IL 61820, USA*
[c]*Department of Applied Mathematics, University of Stellenbosch, Private Bag X1, 7602 Matieland, South Africa*
[d]*Department of Engineering and Applied Sciences, Harvard University, Cambridge, MA 02138, USA*
[e]*Department of Applied Mathematics and Theoretical Physics, Churchill College, University of Cambridge, Cambridge, CB3 0DS, UK*

Received 26 December 2001; accepted 29 September 2003

Abstract

Algorithms are presented for the tanh- and sech-methods, which lead to closed-form solutions of nonlinear ordinary and partial differential equations (ODEs and PDEs). New algorithms are given to find exact polynomial solutions of ODEs and PDEs in terms of Jacobi's elliptic functions.

For systems with parameters, the algorithms determine the conditions on the parameters so that the differential equations admit polynomial solutions in tanh, sech, combinations thereof, Jacobi's sn or cn functions. Examples illustrate key steps of the algorithms.

The new algorithms are implemented in *Mathematica*. The package PDESpecialSolutions.m can be used to automatically compute new special solutions of nonlinear PDEs. Use of the package, implementation issues, scope, limitations, and future extensions of the software are addressed.

A survey is given of related algorithms and symbolic software to compute exact solutions of nonlinear differential equations.

Keywords: Exact solutions; Nonlinear PDEs; Tanh method; Symbolic software

* Corresponding address: Department of Mathematical and Computer Sciences, Colorado School of Mines, Stratton Hall SH 217, Golden, CO 80401-1887, USA. Tel.: +1-303-273-3881; fax: +1-303-273-3875.
E-mail address: whereman@mines.edu (W. Hereman).

0747-7171/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jsc.2003.09.004

1. Introduction

The appearance of solitary wave solutions in nature is quite common. Bell-shaped sech-solutions and kink shaped tanh-solutions model wave phenomena in fluids, plasmas, elastic media, electrical circuits, optical fibers, chemical reactions, bio-genetics, etc. The travelling wave solutions of the Korteweg–de Vries (KdV) and Boussinesq equations, which describe water waves, are famous examples.

Apart from their physical relevance, the knowledge of closed-form solutions of nonlinear ordinary and partial differential equations (ODEs and PDEs) facilitates the testing of numerical solvers, and aids in the stability analysis. Indeed, the exact solutions given in this paper correspond to homoclinic and heteroclinic orbits in phase space, which are the separatrices of stable and unstable regions.

Travelling wave solutions of many nonlinear ODEs and PDEs from soliton theory (and beyond) can often be expressed as polynomials of the hyperbolic tangent and secant functions. An explanation is given in, for example, Hereman and Takaoka (1990). The existence of solitary wave solutions of evolution equations is addressed in Kichenassamy and Olver (1993). The tanh-method provides a straightforward algorithm to compute such particular solutions for a large class of nonlinear PDEs. Consult Malfliet (1992, 2004), Malfliet and Hereman (1996) and Das and Sarma (1999) for a multitude of references to tanh-based techniques and applications.

The tanh-method for, say, a single PDE in $u(x,t)$ works as follows: in a travelling frame of reference, $\xi = c_1 x + c_2 t + \Delta$, one transforms the PDE into an ODE in the new independent variable $T = \tanh \xi$. Since the derivative of tanh is polynomial in tanh, i.e., $T' = 1 - T^2$, all derivatives of T are polynomials of T. Via a chain rule, the polynomial PDE in $u(x,t)$ is transformed into an ODE in $U(T)$, which has polynomial coefficients in T. One then seeks polynomial solutions of the ODE, thus generating a subset of the set of all solutions.

Along the path, one encounters ODEs which are nonlinear, higher-order versions of the ultraspherical differential equation,

$$(1 - x^2)y''(x) - (2\alpha + 1)xy'(x) + n(n + 2\alpha)y(x) = 0, \tag{1}$$

with integer $n \geq 0$ and α real, whose solutions are the Gegenbauer polynomials. Eq. (1) includes the Legendre equation ($\alpha = 1/2$), satisfied by the Legendre polynomials, and the ODEs for Chebeyshev polynomials of type I ($\alpha = 0$) and type II ($\alpha = 1$). Likewise, the associated Legendre equation,

$$(1 - x^2)^2 y''(x) - 2x(x^2 - 1)y'(x) + [n(n + 1)(1 - x^2) - m^2]y(x) = 0, \tag{2}$$

with m and n non-negative integers, appears in solving the Sturm–Liouville problem for the KdV with a sech-square potential (see Drazin and Johnson, 1989).

The appeal and success of the tanh-method lies in the fact that one circumvents integration to get explicit solutions. Variants of the method appear in mathematical physics, plasma physics, and fluid dynamics. For early references see e.g. Malfliet (1992), Yang (1994) and Das and Sarma (1999). Recently, the tanh-methods have been applied to many nonlinear PDEs in multiple independent variables (see Fan, 2002a,b,c, 2003a,b,c; Fan and Hon, 2002, 2003a,b; Gao and Tian, 2001; Li and Liu, 2002; Yao and Li, 2002a,b).

In this paper we present three flavors of tanh- and sech-methods as they apply to nonlinear polynomial systems of ODEs and PDEs. Based on the strategy of the tanh-method, we also present algorithms to compute polynomial solutions in terms of the Jacobi sn and cn functions. Applied to the KdV equation, the so-called cnoidal solution (Drazin and Johnson, 1989) is obtained. For Duffing's equation (Lawden, 1989), we recover known sn and cn-solutions which model vibrations of a nonlinear spring. Sn- and cn-methods are quite effective for symbolically solving nonlinear PDEs as shown in Fu et al. (2001), Parkes et al. (2002), Liu and Li (2002a, submitted for publication), Fan and Zhang (2002), Fan (2003a,b,c), Chen and Zhang (2003a, submitted for publication) and Yan (2003).

We also present our package, `PDESpecialSolutions.m` (Baldwin et al., 2001) in *Mathematica*, which implements the five methods. Without intervention by the user, our software computes travelling wave solutions as polynomials in either $T = \tanh\xi$, $S = \operatorname{sech}\,\xi$, combinations thereof, $\mathrm{CN} = \mathrm{cn}(\xi; m)$, or $\mathrm{SN} = \mathrm{sn}(\xi; m)$ with $\xi = c_1x+c_2y+c_3z+\cdots+c_nt+\Delta = \sum_{j=0}^{N} c_jx_j+\Delta$. The coefficients of the spatial coordinates are the components of the wavevector; the time coefficient is the angular frequency of the wave. The wave travels in the direction of the wavevector; its plane wavefront is perpendicular to that wavevector. Δ is the constant phase. For systems of ODEs or PDEs with constant parameters, the software automatically determines the conditions on the parameters so that the equations might admit polynomial solutions in tanh, sech, both, sn or cn.

Parkes and Duffy (1996) mention the difficulty of using the tanh-method by hand for anything but simple PDEs. Therefore, they automated to some degree the tanh-method using *Mathematica*. Their code ATFM carries out some (but not all) steps of the method. Parkes et al. (1998) also considered solutions to (odd-order generalized KdV) equations in even powers of sech. The code ATFM does not cover solutions involving odd powers of sech. Recently, Parkes et al. (2002) extended their methods to cover the Jacobi elliptic functions. Abbott et al. (2002) produced the function SeriesSn to partially automate the elliptic function method. Li and Liu (2002) designed the *Maple* package RATH to automate the tanh-method. In Liu and Li (2002a) they announce their *Maple* code AJFM for the Jacobi elliptic function method. In Section 8.2 we review the codes ATFM, RATH, AJFM, and SeriesSn and compare them with PDESpecialSolutions.m.

The paper is organized as follows: in Sections 2 and 3, we give the main steps of the algorithms for computing tanh- and sech-solutions of nonlinear polynomial PDEs. We restrict ourselves to polynomial solutions in either tanh or sech. The Boussinesq equation and Hirota–Satsuma system of coupled KdV equations illustrate the steps. For references to both equations see e.g. Ablowitz and Clarkson (1991). In Section 4 we consider a broader class of polynomial solutions involving both tanh and sech. The tanh–sech algorithm is used to solve a system of PDEs due to Gao and Tian (2001). In Section 5 we show how modifying the chain rule allows us to find polynomial solutions in cn and sn. The KdV equation is used to illustrate the steps. In Section 6 we give details of the algorithms to compute the highest-degree of the polynomials, to analyze and solve nonlinear algebraic systems with parameters, and to numerically and symbolically test solutions. The coupled KdV equations illustrate the subtleties of these algorithms. In Section 7 we present exact solutions for several nonlinear ODEs and PDEs. In Section 8 we address other perspectives and extensions of the algorithms, and review related software packages.

We discuss the results and draw some conclusions in Section 9. The use of the package PDESpecialSolutions.m is shown in the Appendix.

2. Algorithm to compute tanh-solutions for nonlinear PDEs

In this section we outline the tanh-method (Malfliet and Hereman, 1996) for the computation of closed-form tanh-solutions for nonlinear PDEs (and ODEs). Each of the five main steps of our algorithm is illustrated for the Boussinesq equation. Details of Steps T2, T4 and T5 are postponed to Section 6.

Given is a system of polynomial PDEs with constant coefficients,

$$\boldsymbol{\Delta}(\mathbf{u}(\mathbf{x}), \mathbf{u}'(\mathbf{x}), \mathbf{u}''(\mathbf{x}), \ldots, \mathbf{u}^{(k)}(\mathbf{x}), \ldots, \mathbf{u}^{(m)}(\mathbf{x})) = \mathbf{0}, \tag{3}$$

where the dependent variable $\mathbf{u}$ has M components u_i, the independent variable $\mathbf{x}$ has N components x_j, and $\mathbf{u}^{(k)}(\mathbf{x})$ denotes the collection of mixed derivative terms of order k. Lower-case Greek letters will denote parameters in (3).

For notational simplicity, in Section 7 we will use dependent variables u, v, w, etc. and independent variables x, y, z, and t.

Example. The classical Boussinesq equation,

$$u_{tt} - u_{xx} + 3uu_{xx} + 3u_x^2 + \alpha u_{xxxx} = 0, \tag{4}$$

with real parameter α, was proposed by Boussinesq to describe surface water waves whose horizontal scale is much larger than the depth of the water (Ablowitz and Clarkson, 1991). Variants of (4) were recently solved by Fan and Hon (2003a).

While one could apply the tanh-method directly to (4), we recast it as a first-order system in time to show the method for a simple system of PDEs. So,

$$\begin{aligned} &u_{1,x_2} + u_{2,x_1} = 0, \\ &u_{2,x_2} + u_{1,x_1} - 3u_1 u_{1,x_1} - \alpha u_{1,3x_1} = 0, \end{aligned} \tag{5}$$

where $x_1 = x$, $x_2 = t$, $u_1(x_1, x_2) = u(x, t)$, and $u_2(x_1, x_2) = u_t(x, t)$. We use

$$u_{i,kx_j} \stackrel{\text{def}}{=} \frac{\partial^k u_i}{\partial x_j^k}, \qquad u_{i,px_jrx_ksx_\ell} \stackrel{\text{def}}{=} \frac{\partial^{p+r+s} u_i}{\partial x_j^p \partial x_k^r \partial x_\ell^s}, \text{ etc.} \tag{6}$$

through out this paper.

Step T1 (Transform the PDE into a Nonlinear ODE). We seek solutions in the travelling frame of reference,

$$\xi = \sum_{j=1}^{N} c_j x_j + \Delta, \tag{7}$$

where c_j and Δ are constant.

The tanh-method seeks polynomial solutions expressible in the hyperbolic tangent, $T = \tanh \xi$. Based on the identity $\cosh^2 \xi - \sinh^2 \xi = 1$ one computes

$$\begin{aligned} \tanh' \xi &= \operatorname{sech}^2 \xi = 1 - \tanh^2 \xi, \\ \tanh'' \xi &= -2 \tanh \xi + 2 \tanh^3 \xi, \text{ etc.} \end{aligned} \tag{8}$$

Therefore, the first and, consequently, all higher-order derivatives are polynomials in T. Since $T' = 1 - T^2$, repeatedly applying the chain rule,

$$\frac{\partial \bullet}{\partial x_j} = \frac{\partial \xi}{\partial x_j} \frac{\mathrm{d}T}{\mathrm{d}\xi} \frac{\mathrm{d}\bullet}{\mathrm{d}T} = c_j (1 - T^2) \frac{\mathrm{d}\bullet}{\mathrm{d}T}, \tag{9}$$

transforms the system of PDEs into a coupled system of nonlinear ODEs,

$$\boldsymbol{\Delta}(T, \mathbf{U}(T), \mathbf{U}'(T), \mathbf{U}''(T), \ldots, \mathbf{U}^{(m)}(T)) = \mathbf{0}, \tag{10}$$

with $\mathbf{U}(T) = \mathbf{u}(\mathbf{x})$. Each component of $\boldsymbol{\Delta}$ is a nonlinear ODE with polynomial coefficients in T.

Example. Substituting

$$\begin{aligned} u_{i,x_j} &= c_j (1 - T^2) U_i', \\ u_{i,2x_j} &= c_j^2 (1 - T^2)[(1 - T^2) U_i']' = c_j^2 (1 - T^2)[-2T U_i' + (1 - T^2) U_i''], \\ u_{i,3x_j} &= c_j^3 (1 - T^2)[-2T(1 - T^2) U_i' + (1 - T^2)^2 U_i'']' \\ &= c_j^3 (1 - T^2)[-2(1 - 3T^2) U_i' - 6T(1 - T^2) U_i'' + (1 - T^2)^2 U_i'''], \end{aligned} \tag{11}$$

into (5), and cancelling common $(1 - T^2)$ factors, yields

$$\begin{aligned} & c_2 U_1' + c_1 U_2' = 0, \\ & c_2 U_2' + c_1 U_1' - 3c_1 U_1 U_1' + \alpha c_1^3 [2(1 - 3T^2) U_1' \\ & \quad + 6T(1 - T^2) U_1'' - (1 - T^2)^2 U_1'''] = 0, \end{aligned} \tag{12}$$

where $U_1(T) = u_1(x_1, x_2)$ and $U_2(T) = u_2(x_1, x_2)$.

Step T2 (Determine the Degree of the Polynomial Solutions). Seeking polynomial solutions of the form

$$U_i(T) = \sum_{j=0}^{M_i} a_{ij} T^j, \tag{13}$$

we must determine the leading exponents M_i before the a_{ij} can be computed. We assume that $M_i \geq 1$ to avoid trivial solutions. Substituting U_i into (10), the coefficients of every power of T in every equation must vanish. In particular, the highest degree terms must vanish. Since the highest degree terms depend only on T^{M_i} in (13), it suffices to substitute $U_i(T) = T^{M_i}$ into the left-hand side of (10). In the resulting polynomial system $\mathbf{P}(T)$, equating every two possible highest exponents in every component P_i gives a linear system for M_i. That linear system is then solved.

If one or more exponents M_i remain undetermined, assign an integer value to the free M_i so that every equation in (10) has at least two different terms with equal highest exponents. Carry each the solution to Step T3.

Example. For the Boussinesq system, substituting $U_1(T) = T^{M_1}$ and $U_2(T) = T^{M_2}$ into (12), and equating the highest exponents of T for each equation, gives

$$M_1 - 1 = M_2 - 1, \qquad 2M_1 - 1 = M_1 + 1. \tag{14}$$

Then, $M_1 = M_2 = 2$, and

$$U_1(T) = a_{10} + a_{11}T + a_{12}T^2, \qquad U_2(T) = a_{20} + a_{21}T + a_{22}T^2. \tag{15}$$

Step T3 (Derive the Algebraic System for the Coefficients a_{ij}). To generate the system for the unknown coefficients a_{ij} and wave parameters c_j, substitute (13) into (10) and set the coefficients of T^i to zero. The resulting nonlinear algebraic system for the unknowns a_{ij} is parameterized by the c_j, and the external parameters (in lower-case Greek letters) of system (3), if any.

Example. Continuing with the Boussinesq system, after substituting (15) into (12), and collecting the terms of like degree in T, we get (in order of complexity)

$$\begin{aligned}
&a_{21}c_1 + a_{11}c_2 = 0,\\
&a_{22}c_1 + a_{12}c_2 = 0,\\
&a_{11}c_1(3a_{12} + 2\alpha c_1^2) = 0,\\
&a_{12}c_1(a_{12} + 4\alpha c_1^2) = 0,\\
&a_{11}c_1 - 3a_{10}a_{11}c_1 + 2\alpha a_{11}c_1^3 + a_{21}c_2 = 0,\\
&-3a_{11}^2c_1 + 2a_{12}c_1 - 6a_{10}a_{12}c_1 + 16\alpha a_{12}c_1^3 + 2a_{22}c_2 = 0,
\end{aligned} \tag{16}$$

with unknowns a_{10}, a_{11}, a_{12}, a_{20}, a_{21}, a_{22}, and parameters c_1, c_2, and α.

Step T4 (Solve the Nonlinear Parameterized Algebraic System). The most difficult step is solving the nonlinear algebraic system. To do so, we designed a customized, yet powerful, nonlinear solver (see Section 6.2 for details).

The nonlinear algebraic system is solved under the following assumptions:

(i) All parameters, α, β, etc., in (3) are strictly positive. Vanishing parameters may change the exponents M_i in Step T2. To compute solutions corresponding to negative parameters, reverse the signs of the parameters in the PDE. For example, replace α by $-\alpha$ in (4).

(ii) The coefficients of the highest power terms $(a_{iM_i}, i = 1, \ldots, M)$ in (13) are all nonzero (for consistency with Step T2).

(iii) All c_j are nonzero (demanded by the physical nature of the solutions).

Example. Assuming c_1, c_2, a_{12}, a_{22}, and α are nonzero, the solution of (16) is

$$\begin{aligned}
&a_{10} = (c_1^2 - c_2^2 + 8\alpha c_1^4)/(3c_1^2), \qquad a_{11} = 0, \qquad a_{12} = -4\alpha c_1^2,\\
&a_{20} = \text{arbitrary}, \qquad a_{21} = 0, \qquad a_{22} = 4\alpha c_1 c_2.
\end{aligned} \tag{17}$$

In this case, there are no conditions on the parameters c_1, c_2 and α.

Step T5 (Build and Test the Solitary Wave Solutions). Substitute the solutions obtained in Step T4 into (13) and reverse Step T1 to obtain the explicit solutions in the

original variables. It is prudent to test the solutions by substituting them into (3). For details about testing see Section 6.3.

Example. Inserting (17) into (15), and replacing $T = \tanh(c_1x + c_2t + \Delta)$, the closed form solution for (5) (or (4)) is

$$\begin{aligned} u(x,t) &= u_1(x,t) = (c_1^2 - c_2^2 + 8\alpha c_1^4)/(3c_1^2) - 4\alpha c_1^2 \tanh^2(c_1x + c_2t + \Delta), \\ u_2(x,t) &= -\int u_{1,t}(x,t)\mathrm{d}x = a_{20} + 4\alpha c_1c_2 \tanh^2(c_1x + c_2t + \Delta), \end{aligned} \tag{18}$$

where a_{20}, c_1, c_2, α and Δ are arbitrary. Steps T1–T5 must be repeated if one or more of the external parameters (lower-case Greeks) are set to zero.

3. Algorithm to compute sech-solutions for nonlinear PDEs

In this section we restrict ourselves to polynomial solutions of (3) in sech. Polynomial solutions involving both sech and tanh are dealt with in Section 4. Details of the algorithms for Steps S2, S4 and S5 are given in Section 6.

Using $\tanh^2\xi + \mathrm{sech}^2\,\xi = 1$, solution (18) of (5) can be expressed as

$$\begin{aligned} u_1(x,t) &= (c_1^2 - c_2^2 - 4\alpha c_1^4)/(3c_1^2) + 4\alpha c_1^2 \mathrm{sech}^2(c_1x + c_2t + \Delta), \\ u_2(x,t) &= a_{20} + 4\alpha c_1c_2 - 4\alpha c_1c_2 \mathrm{sech}^2(c_1x + c_2t + \Delta). \end{aligned} \tag{19}$$

Obviously, any even order solution in tanh can be written in even orders of sech. Some PDEs however have polynomial solutions of odd-order in sech. For example, the modified KdV equation (Ablowitz and Clarkson, 1991),

$$u_t + \alpha u^2 u_x + u_{xxx} = 0, \tag{20}$$

has the solution

$$u(x,t) = \pm c_1\sqrt{6/\alpha}\,\mathrm{sech}(c_1x - c_1^3t + \Delta), \tag{21}$$

which cannot be found using the tanh-method.

Example. The five main steps of the sech-algorithm are illustrated with the Hirota–Satsuma system of coupled KdV equations (Ablowitz and Clarkson, 1991),

$$\begin{aligned} &u_t - \alpha(6uu_x + u_{xxx}) + 2\beta vv_x = 0, \\ &v_t + 3uv_x + v_{xxx} = 0, \end{aligned} \tag{22}$$

with real parameters α, β. Sech-type solutions were reported in Hereman (1991) and Fan and Hon (2002). Variants and generalizations of (22) were solved in Chen and Zhang (2003a) and Yan (2003).

Letting $u_1(x_1, x_2) = u(x,t)$ and $u_2(x_1, x_2) = v(x,t)$, Eq. (22) is then

$$\begin{aligned} &u_{1,x_2} - \alpha(6u_1u_{1,x_1} + u_{1,3x_1}) + 2\beta u_2u_{2,x_1} = 0, \\ &u_{2,x_2} + 3u_1u_{2,x_1} + u_{2,3x_1} = 0. \end{aligned} \tag{23}$$

Step S1 (Transform the PDE into a Nonlinear ODE). Adhering to the travelling frame of reference (7), and using $\tanh^2\xi + \operatorname{sech}^2\xi = 1$,

$$\operatorname{sech}'\xi = -\operatorname{sech}\xi\tanh\xi = -\operatorname{sech}\xi\sqrt{1-\operatorname{sech}^2\xi}. \tag{24}$$

Setting $S = \operatorname{sech}\xi$ and repeatedly applying the chain rule,

$$\frac{\partial\bullet}{\partial x_j} = \frac{\partial\xi}{\partial x_j}\frac{\mathrm{d}S}{\mathrm{d}\xi}\frac{\mathrm{d}\bullet}{\mathrm{d}S} = -c_j S\sqrt{1-S^2}\frac{\mathrm{d}\bullet}{\mathrm{d}S}, \tag{25}$$

(3) is transformed into a system of nonlinear ODEs of the form

$$\boldsymbol{\Gamma}(S,\mathbf{U}(S),\mathbf{U}'(S),\ldots) + \sqrt{1-S^2}\,\boldsymbol{\Pi}(S,\mathbf{U}(S),\mathbf{U}'(S),\ldots) = \mathbf{0}, \tag{26}$$

where $\mathbf{U}(S) = \mathbf{u}(\mathbf{x})$, and all components of $\boldsymbol{\Gamma}$ and $\boldsymbol{\Pi}$ are ODEs with polynomial coefficients in S. If either $\boldsymbol{\Gamma}$ or $\boldsymbol{\Pi}$ are identically $\mathbf{0}$, then

$$\boldsymbol{\Delta}(S,\mathbf{U}(S),\mathbf{U}'(S),\ldots) = \mathbf{0}, \tag{27}$$

where $\boldsymbol{\Delta}$ is either $\boldsymbol{\Gamma}$ or $\boldsymbol{\Pi}$, whichever is nonzero. For this to occur, the order of all terms in any equation in (3) must be even or odd (as is the case in (23)).

Any term in (3) for which the total number of derivatives is even contributes to the first term in (26); while any term of odd order contributes to the second term. Section 4 deals with any case for which neither $\boldsymbol{\Gamma}$ or $\boldsymbol{\Pi}$ is identically $\mathbf{0}$.

Example. Substituting

$$\begin{aligned}
u_{i,x_j} &= -c_j S\sqrt{1-S^2}U_i',\\
u_{i,x_jx_k} &= c_jc_kS\sqrt{1-S^2}\left[S\sqrt{1-S^2}U_i'\right]'\\
&= c_jc_kS[(1-2S^2)U_i' + S(1-S^2)U_i''],\\
u_{i,x_jx_kx_l} &= -c_jc_kc_lS\sqrt{1-S^2}[S(1-2S^2)U_i' + S(1-S^2)U_i'']'\\
&= -c_jc_kc_lS\sqrt{1-S^2}[(1-6S^2)U_i' + 3S(1-2S^2)U_i''\\
&\quad + S^2(1-S^2)U_i'''],
\end{aligned} \tag{28}$$

into (23), and cancelling the common $S\sqrt{1-S^2}$ factors yields

$$\begin{aligned}
&c_2U_1' - 6\alpha c_1U_1U_1' - \alpha c_1^3[(1-6S^2)U_1' + 3S(1-2S^2)U_1''\\
&\qquad + S^2(1-S^2)U_1''']+2\beta c_1U_2U_2' = 0,\\
&c_2U_2' + 3c_1U_1U_2' + c_1^3[(1-6S^2)U_2' + 3S(1-2S^2)U_2'' + S^2(1-S^2)U_2''']=0,
\end{aligned} \tag{29}$$

with $U_1(T) = u_1(x_1,x_2)$ and $U_2(T) = u_2(x_1,x_2)$. Note that (29) matches (27) with $\boldsymbol{\Delta} = \boldsymbol{\Pi}$, since $\boldsymbol{\Gamma} = \mathbf{0}$.

Step S2 (Determine the Degree of the Polynomial Solutions). We seek polynomial solutions of the form,

$$U_i(S) = \sum_{j=0}^{M_i} a_{ij} S^j. \tag{30}$$

To determine the M_i exponents, substitute $U_i(S) = S^{M_i}$ into the left-hand side of (27) and proceed as in Step T2. Continue with Step S3 for each solution of M_i. If some of the M_i exponents are undetermined, try all legitimate values for the free M_i. See Section 6.1 for more details.

Example. For (23), substituting $U_1(S) = S^{M_1}$, $U_2(S) = S^{M_2}$ into (29) and equating the highest exponents in the second equation yields $M_1 + M_2 - 1 = 1 + M_2$, or $M_1 = 2$. The maximal exponents coming from the first equation are $2M_1 - 1$ (from the $U_1 U_1'$ term), $M_1 + 1$ (from U_1'''), and $2M_2 - 1$ (from $U_2 U_2'$). Using $M_1 = 2$, two cases emerge: (i) the third exponent is less than the first two (equal) exponents, i.e., $2M_2 - 1 < 3$, so $M_2 = 1$, or (ii) all three exponents are equal, in which case $M_2 = 2$. For the case $M_1 = 2$ and $M_2 = 1$,

$$U(S) = a_{10} + a_{11}S + a_{12}S^2, \qquad V(S) = a_{20} + a_{21}S, \tag{31}$$

and, for the case $M_1 = M_2 = 2$,

$$U(S) = a_{10} + a_{11}S + a_{12}S^2, \qquad V(S) = a_{20} + a_{21}S + a_{22}S^2. \tag{32}$$

Step S3 (Derive the Algebraic System for the Coefficients a_{ij}). Follow the strategy in Step T3.

Example. After substituting (31) into (29), cancelling common numerical factors, and organizing the equations (according to complexity) one obtains

$$\begin{aligned}
&a_{11}a_{21}c_1 = 0,\\
&\alpha a_{11}c_1(3a_{12} - c_1^2) = 0,\\
&\alpha a_{12}c_1(a_{12} - 2c_1^2) = 0,\\
&a_{21}c_1(a_{12} - 2c_1^2) = 0,\\
&a_{21}(3a_{10}c_1 + c_1^3 + c_2) = 0,\\
&6\alpha a_{10}a_{11}c_1 - 2\beta a_{20}a_{21}c_1 + \alpha a_{11}c_1^3 - a_{12}c_2 = 0,\\
&3\alpha a_{11}^2 c_1 + 6\alpha a_{10}a_{12}c_1 - \beta a_{21}^2 c_1 + 4\alpha a_{12}c_1^3 - a_{12}c_2 = 0.
\end{aligned} \tag{33}$$

Similarly, after substitution of (32) into (29), one gets

$$\begin{aligned}
&a_{22}c_1(a_{12} - 4c_1^2) = 0,\\
&a_{21}(3a_{10}c_1 + c_1^3 + c_2) = 0,\\
&c_1(a_{12}a_{21} + 2a_{11}a_{22} - 2a_{21}c_1^2) = 0,\\
&c_1(3\alpha a_{11}a_{12} - \beta a_{21}a_{22} - \alpha a_{11}c_1^2) = 0,\\
&c_1(3\alpha a_{12}^2 - \beta a_{22}^2 - 6\alpha a_{12}c_1^2) = 0,\\
&6\alpha a_{10}a_{11}c_1 - 2\beta a_{20}a_{21}c_1 + \alpha a_{11}c_1^3 - a_{11}c_2 = 0,\\
&3a_{11}a_{21}c_1 + 6a_{10}a_{22}c_1 + 8a_{22}c_1^3 + 2a_{22}c_2 = 0,\\
&3\alpha a_{11}^2 c_1 + 6\alpha a_{10}^2 a_{12}c_1 - \beta a_{21}^2 c_1 - 2\beta a_{20}a_{22}c_1 + 4\alpha a_{12}c_1^3 - a_{12}c_2 = 0.
\end{aligned} \tag{34}$$

Step S4 (Solve the Nonlinear Parameterized Algebraic System). Similar strategy as in Step T4.

Example. For α, β, c_1, c_2, a_{12} and a_{21} all nonzero, the solution of (33) is

$$\begin{aligned} &a_{10} = -(c_1^3 + c_2)/(3c_1), \qquad a_{11} = 0, \qquad a_{12} = 2c_1^2, \\ &a_{20} = 0, \qquad a_{21} = \pm\sqrt{(4\alpha c_1^4 - 2(1+2\alpha)c_1c_2)/\beta}. \end{aligned} \tag{35}$$

For α, β, c_1, c_2, a_{12} and a_{22} nonzero, the solution of (34) is

$$\begin{aligned} &a_{10} = -(4c_1^3 + c_2)/(3c_1), \qquad a_{11} = 0, \qquad a_{12} = 4c_1^2, \\ &a_{20} = \pm(4\alpha c_1^3 + (1+2\alpha)c_2)/\left(c_1\sqrt{6\alpha\beta}\right), \qquad a_{21} = 0, \\ &a_{22} = \mp 2c_1^2\sqrt{6\alpha/\beta}. \end{aligned} \tag{36}$$

Step S5 (Build and Test the Solitary Wave Solutions). Substitute the result of Step S4 into (30) and reverse Step S1. Test the solutions.

Example. The solitary wave solutions of (23) are

$$\begin{aligned} &u(x,t) = -(c_1^3 + c_2)/(3c_1) + 2c_1^2\mathrm{sech}^2(c_1x + c_2t + \Delta), \\ &v(x,t) = \pm\sqrt{[4\alpha c_1^4 - 2(1+2\alpha)c_1c_2]/\beta}\,\mathrm{sech}(c_1x + c_2t + \Delta), \end{aligned} \tag{37}$$

and

$$\begin{aligned} u(x,t) &= -(4c_1^3 + c_2)/(3c_1) + 4c_1^2\mathrm{sech}^2(c_1x + c_2t + \Delta), \\ v(x,t) &= \pm(4\alpha c_1^3 + (1+2\alpha)c_2)/\left(c_1\sqrt{6\alpha\beta}\right) \\ &\quad \mp 2c_1^2\sqrt{6\alpha/\beta}\,\mathrm{sech}^2(c_1x + c_2t + \Delta). \end{aligned} \tag{38}$$

In both cases c_1, c_2, α, β, and Δ are arbitrary. These solutions contain the solutions reported in Hereman (1991).

Steps S1–S5 must be repeated if any of the parameters in (3) are set to zero.

4. Algorithm for mixed tanh–sech solutions for PDEs

The five main steps of our algorithm to compute mixed tanh–sech solutions for (3) are presented below. Here we seek particular solutions of (26) when $\boldsymbol{\Gamma} \neq \mathbf{0}$ and $\boldsymbol{\Pi} \neq \mathbf{0}$. On could apply the method of Section 3 to (26) in 'squared' form $\boldsymbol{\Gamma}^2(S, \mathbf{U}(S), \mathbf{U}'(S), \ldots) - (1-S^2)\boldsymbol{\Pi}^2(S, \mathbf{U}(S), \mathbf{U}'(S), \ldots) = \mathbf{0}$. For anything but simple cases, the computations are unwieldy. Alternatively, since $T = \sqrt{1-S^2}$, Eq. (26) may admit solutions of the form

$$U_i(S) = \sum_{j=0}^{\tilde{M}_i}\sum_{k=0}^{\tilde{N}_i} \tilde{a}_{i,jk} S^j T^k. \tag{39}$$

However, (39) can always be rearranged such that

$$U_i(S) = \sum_{j=0}^{M_i} a_{ij} S^j + T \sum_{j=0}^{N_i} b_{ij} S^j = \sum_{j=0}^{M_i} a_{ij} S^j + \sqrt{1-S^2} \sum_{j=0}^{N_i} b_{ij} S^j. \tag{40}$$

The polynomial solutions in S from Section 3 are special cases of this broader class. Remarkably, (27) where $\sqrt{1-S^2}$ is not explicitly present also admits solutions of the form (40). See Section 7.6 for an example.

Computing solutions of type (30) with the tanh–sech method is inefficient and costly, as the following example and the examples in Sections 7.5 and 7.6 show.

Example. We illustrate this algorithm with the system (Gao and Tian, 2001):

$$\begin{aligned} &u_t - u_x - 2v = 0, \\ &v_t + 2uw = 0, \\ &w_t + 2uv = 0. \end{aligned} \tag{41}$$

Step ST1 (Transform the PDE into a Nonlinear ODE). Same as Step S1.

Example. Use (25) to transform (41) into

$$\begin{aligned} &(c_1 - c_2) S\sqrt{1-S^2} U_1' - 2U_2 = 0, \\ &c_2 S\sqrt{1-S^2} U_2' - 2U_1 U_3 = 0, \\ &c_2 S\sqrt{1-S^2} U_3' - 2U_1 U_2 = 0 \end{aligned} \tag{42}$$

with $U_i(S) = u_i(x_1, x_2),\ i = 1, 2, 3$.

Step ST2 (Determine the Degree of the Polynomial Solutions). Seeking solutions of form (40), we must first determine the leading M_i and N_i exponents. Substituting $U_i(S) = a_{i0} + a_{iM_i} S^{M_i} + \sqrt{1-S^2}\,(b_{i0} + b_{iN_i} S^{N_i})$ into the left-hand side of (26), we get an expression of the form

$$\mathbf{P}(S) + \sqrt{1-S^2}\mathbf{Q}(S), \tag{43}$$

where $\mathbf{P}$ and $\mathbf{Q}$ are polynomials in S.

Consider separately the possible balances of highest exponents in all P_i and Q_i. Then solve the resulting linear system(s) for the unknowns M_i and N_i. Continue with each solution in Step ST3.

In contrast to Step S2, we no longer assume $M_i \geq 1$, $N_i \geq 1$. Even with some M_i or N_i zero, non-constant solutions $U_i(S)$ often arise. In most examples, however, the sets of balance equations for M_i and N_i are too large or the corresponding linear systems are under-determined (i.e., several leading exponents remain arbitrary). To circumvent the problem, we set all $M_i = 2$ and all $N_i = 1$, restricting the solutions to (at most) quadratic in S and T.

Example. For (42), we set all $M_i = 2$, $N_i = 1$, and continue with

$$U_i(S) = a_{i0} + a_{i1}S + a_{i2}S^2 + \sqrt{1-S^2}(b_{i0} + b_{i1}S), \qquad i = 1, 2, 3. \tag{44}$$

Step ST3 (Derive the Algebraic System for the Coefficients a_{ij} and b_{ij}). Substituting (40) into (26) gives $\widetilde{\mathbf{P}}(S) + \sqrt{1-S^2}\widetilde{\mathbf{Q}}(S)$, which must vanish identically. Hence, equate to zero the coefficients of the power terms in S so that $\widetilde{\mathbf{P}} = \mathbf{0}$ and $\widetilde{\mathbf{Q}} = \mathbf{0}$.

Example. After substitution of (44) into (42), the resulting nonlinear algebraic system for the coefficients a_{ij} and b_{ij} contains 25 equations (not shown).

Step ST4 (Solve the Nonlinear Parameterized Algebraic System). In contrast to Step S4 we no longer assume that a_{iM_i} and b_{iN_i} are nonzero (at the cost of generating some constant solutions, which we discard later).

Example. For (41), there are 11 solutions. Three are trivial, leading to constant U_i. Eight are nontrivial solutions giving the results below.

Step ST5 (Build and Test the Solitary Wave Solutions). Proceed as in Step S5.

Example. The solitary wave solutions of (41) are

$$\begin{aligned} u(x,t) &= \pm c_2 \tanh\xi, \\ v(x,t) &= \mp\tfrac{1}{2}c_2(c_1-c_2)\mathrm{sech}^2\,\xi, \\ w(x,t) &= -\tfrac{1}{2}c_2(c_1-c_2)\mathrm{sech}^2\,\xi, \end{aligned} \tag{45}$$

which could have been obtained with the tanh-method of Section 2;

$$\begin{aligned} u(x,t) &= \pm i c_2 \mathrm{sech}\,\xi, \\ v(x,t) &= \pm\tfrac{1}{2}i c_2(c_1-c_2)\tanh\xi\,\mathrm{sech}\,\xi, \\ w(x,t) &= \tfrac{1}{4}c_2(c_1-c_2)(1-2\mathrm{sech}^2\,\xi), \end{aligned} \tag{46}$$

reported in Gao and Tian (2001); and the two complex solutions

$$\begin{aligned} u(x,t) &= \pm\tfrac{1}{2}i c_2(\mathrm{sech}\,\xi \pm i\tanh\xi), \\ v(x,t) &= \tfrac{1}{4}c_2(c_1-c_2)\mathrm{sech}\,\xi(\mathrm{sech}\,\xi \pm i\tanh\xi), \\ w(x,t) &= -\tfrac{1}{4}c_2(c_1-c_2)\mathrm{sech}\,\xi(\mathrm{sech}\,\xi \pm i\tanh\xi). \end{aligned} \tag{47}$$

In all solutions $\xi = c_1x + c_2t + \Delta$, with c_1, c_2 and Δ arbitrary. The complex conjugates of (47) are also solutions.

5. Algorithms used to compute sn and cn solutions for PDEs

5.1. Computation of solutions involving Cn

In this section we give the main steps (labelled CN1–CN5) of our algorithm used to compute polynomial solutions of (3) in terms of Jacobi's elliptic cosine function (cn). Modifications needed for solutions involving the sn function are given at the end of this section. Details for Steps CN2, CN4 and CN5 are shown in Section 6.

Example. Consider the KdV equation (Ablowitz and Clarkson, 1991),

$$u_t + \alpha u u_x + u_{xxx} = 0, \tag{48}$$

with real constant α. The KdV equation models, among other things, waves in shallow water and ion-acoustic waves in plasmas.

Step CN1 (Transform the PDE into a Nonlinear ODE). Similar to the strategy in S1 and T1, using (Lawden, 1989)

$$\mathrm{sn}^2(\xi; m) = 1 - \mathrm{cn}^2(\xi; m), \qquad \mathrm{dn}^2(\xi; m) = 1 - m + m\mathrm{cn}^2(\xi; m), \tag{49}$$

and

$$\mathrm{cn}'(\xi; m) = -\mathrm{sn}(\xi; m)\mathrm{dn}(\xi; m), \tag{50}$$

one has $\mathrm{CN}' = -\sqrt{(1-\mathrm{CN}^2)(1-m+m\mathrm{CN}^2)}$ where $\mathrm{CN} = \mathrm{cn}(\xi; m)$ is the Jacobi elliptic cosine with argument ξ and modulus $0 \le m \le 1$.

Repeatedly applying the chain rule

$$\frac{\partial \bullet}{\partial x_j} = \frac{\partial \xi}{\partial x_j}\frac{\mathrm{dCN}}{\mathrm{d}\xi}\frac{\mathrm{d}\bullet}{\mathrm{dCN}} = -c_j\sqrt{(1-\mathrm{CN}^2)(1-m+m\mathrm{CN}^2)}\frac{\mathrm{d}\bullet}{\mathrm{dCN}}, \tag{51}$$

system (3) is transformed into a nonlinear ODE system. In addition to the c_j, the algorithm introduces m as an extra parameter.

Example. Using (51) to transform (48) we have

$$\begin{aligned}&(c_1^3(1-2m+6m\mathrm{CN}^2) - c_2 - \alpha c_1 U_1)U_1' \\ &\quad + 3c_1^3\mathrm{CN}(1-2m+2m\mathrm{CN}^2)U_1'' - c_1^3(1-\mathrm{CN}^2)(1-m+m\mathrm{CN}^2)U_1''' = 0.\end{aligned} \tag{52}$$

Step CN2 (Determine the Degree of the Polynomial Solutions). Follow the strategy in Step T2.

Example. For (48), substituting $U_1(\mathrm{CN}) = \mathrm{CN}^{M_1}$ into (52) and equating the highest exponents gives $1 + M_1 = -1 + 2M_1$. Then, $M_1 = 2$, and

$$U_1(\mathrm{CN}) = a_{10} + a_{11}\mathrm{CN} + a_{12}\mathrm{CN}^2. \tag{53}$$

Step CN3 (Derive the Algebraic System for the Coefficients a_{ij}). Proceed as in Step T3.

Example. For (48), after substituting (53) into (52), one finds

$$\begin{aligned}&a_{11}c_1(\alpha a_{12} - 2mc_1^2) = 0,\\ &a_{12}c_1(\alpha a_{12} - 12mc_1^2) = 0,\\ &a_{11}(\alpha a_{10}c_1 - c_1^3 + 2mc_1^3 + c_2) = 0,\\ &\alpha a_{11}^2 c_1 + a_{12}(2\alpha a_{10}c_1 - 16mc_1^3 - 8c_1^3 + 2c_2) = 0.\end{aligned} \tag{54}$$

Step CN4 (Solve the Nonlinear Parameterized Algebraic System). Solve the system as in Step T4.

Example. For c_1, c_2, m, α and a_{12} nonzero, the solution of (54) is

$$a_{10} = [4c_1^3(1-2m) - c_2]/(\alpha c_1), \qquad a_{11} = 0, \qquad a_{12} = (12mc_1^2)/\alpha. \tag{55}$$

Step CN5 (Build and Test the Solitary Wave Solutions). Substitute the results of Step CN4 into (53). Reverse Step CN1. Test the solutions.

Example. The cnoidal wave solution of (48) is

$$u(x,t) = [4c_1^3(1-2m) - c_2]/(\alpha c_1) + (12mc_1^2)/(\alpha)\mathrm{cn}^2(c_1x + c_2t + \Delta; m), \quad (56)$$

where c_1, c_2, α, Δ and modulus m are arbitrary. If any of the parameters in (3) are zero, Steps CN1–CN5 should be repeated.

5.2. Computation of solutions involving Sn

To find solutions in terms of Jacobi's sn function, one uses the identities,

$$\begin{aligned} &\mathrm{cn}^2(\xi; m) = 1 - \mathrm{sn}^2(\xi; m), \qquad \mathrm{dn}^2(\xi; m) = 1 - m\mathrm{sn}^2(\xi; m), \\ &\mathrm{sn}'(\xi; m) = \mathrm{cn}(\xi; m)\mathrm{dn}(\xi; m). \end{aligned} \quad (57)$$

Then, $\mathrm{SN}' = \sqrt{(1-\mathrm{SN}^2)(1-m\mathrm{SN}^2)}$, where $\mathrm{SN} = \mathrm{sn}(\xi; m)$ is the Jacobi elliptic sine with argument ξ and modulus $0 \le m \le 1$. The steps are identical to the cn case, except one uses the chain rule

$$\frac{\partial \bullet}{\partial x_j} = \frac{\partial \xi}{\partial x_j}\frac{\mathrm{dSN}}{\mathrm{d}\xi}\frac{\mathrm{d}\bullet}{\mathrm{dSN}} = c_j\sqrt{(1-\mathrm{SN}^2)(1-m\mathrm{SN}^2)}\frac{\mathrm{d}\bullet}{\mathrm{dSN}}. \quad (58)$$

Since (51) and (58) involve roots, as in Sections 3 and 4 there is no reason to restrict the solutions to polynomials in only cn or sn. Solutions involving both sn and cn (or combinations with dn) are beyond the scope of this paper.

Finally, from the sn and cn solutions, sin, cos, sech, and tanh-solutions can be obtained by taking the appropriate limits for the modulus ($m \to 0$, and $m \to 1$). Indeed, $\mathrm{sn}(\xi; 0) = \sin(\xi)$, $\mathrm{sn}(\xi; 1) = \tanh(\xi)$, $\mathrm{cn}(\xi; 0) = \cos(\xi)$, $\mathrm{cn}(\xi; 1) = \mathrm{sech}(\xi)$. No need to compute solutions in dn explicitly since $\mathrm{cn}(\sqrt{m}\xi; 1/m) = \mathrm{dn}(\xi; m)$.

6. Key algorithms

In this section we present in a uniform manner the details of steps two, four and five of the algorithms in Sections 2–5.

6.1. Algorithm to compute the degree of the polynomials

Step M1 (Substitute the Leading-Order Ansatz). A tracking variable is attached to each term in the original system of PDEs. Let `Tr[i]` denote the tracking variable of the ith term in (3).

The first step of the main algorithms leads to a system of parameterized ODEs in $\mathbf{U}, \mathbf{U}', \mathbf{U}'', \ldots, \mathbf{U}^{(m)}$. These ODEs match the form

$$\boldsymbol{\Gamma}(F, \mathbf{U}(F), \mathbf{U}'(F), \ldots) + \sqrt{R(F)}\boldsymbol{\Pi}(F, \mathbf{U}(F), \mathbf{U}'(F), \ldots) = \mathbf{0}, \quad (59)$$

Table 1
Values for $R(F)$ in (59)

F	$R(F)$
T	0
S	$1 - S^2$
CN	$(1 - \mathrm{CN}^2)(1 - m + m\mathrm{CN}^2)$
SN	$(1 - \mathrm{SN}^2)(1 - m\mathrm{SN}^2)$

where F is either T, S, CN, or SN, and $R(F)$ is defined in Table 1. Since the highest degree term only depends on F^{M_i}, it suffices to substitute

$$U_i(F) \to F^{M_i} \tag{60}$$

into (59).

Example. We use the coupled KdV equations (22) as our leading example:

$$\begin{aligned} &\mathtt{Tr}[1]u_t - 6\alpha\mathtt{Tr}[2]uu_x + 2\beta\mathtt{Tr}[3]vv_x - \alpha\mathtt{Tr}[4]u_{xxx} = 0, \\ &\mathtt{Tr}[5]v_t + 3\mathtt{Tr}[6]uv_x + \mathtt{Tr}[7]v_{xxx} = 0. \end{aligned} \tag{61}$$

Step S1 resulted in (29) with $\boldsymbol{\Pi} = \mathbf{0}$. Substituting (60) into (61), we get

$$\begin{aligned} &(\mathtt{Tr}[1]c_2M_1 - \alpha\mathtt{Tr}[4]c_1^3M_1^3)S^{M_1-1} + \alpha\mathtt{Tr}[4]c_1^3M_1(M_1+1)(M_1+2)S^{M_1+1} \\ &\qquad - 6\alpha\mathtt{Tr}[2]c_1M_1S^{2M_1-1} + 2\beta\mathtt{Tr}[3]c_1M_2S^{2M_2-1} = 0, \\ &(\mathtt{Tr}[5]c_2M_2 + \mathtt{Tr}[7]c_1^3M_2^3)S^{M_2-1} - \mathtt{Tr}[7]c_1^3M_2(M_2+1)(M_2+2)S^{M_2+1} \\ &\qquad + 3\mathtt{Tr}[6]c_1M_2S^{M_1+M_2-1} = 0. \end{aligned} \tag{62}$$

Step M2 (Collect Exponents and Prune Sub-dominant Branches). The balance of highest exponents must come from different terms in (3). For each equation Δ_i and for each tracking variable, collect the exponents of F, remove duplicates, and non-maximal exponents. For example, $M_1 - 1$ can be removed from $\{M_1 + 1, M_1 - 1\}$ because $M_1 + 1 > M_1 - 1$.

Example. Collecting the exponents of S in (62), we get the unpruned list:

Δ_1	Δ_2
$\mathtt{Tr}[1]$: $\{M_1 - 1\}$	$\mathtt{Tr}[5]$: $\{M_2 - 1\}$
$\mathtt{Tr}[2]$: $\{2M_1 - 1\}$	$\mathtt{Tr}[6]$: $\{M_1 + M_2 - 1\}$
$\mathtt{Tr}[3]$: $\{2M_2 - 1\}$	$\mathtt{Tr}[7]$: $\{M_2 + 1, M_2 + 1, M_2 + 1, M_2 - 1\}$
$\mathtt{Tr}[4]$: $\{M_1 + 1, M_1 + 1, M_1 + 1, M_1 - 1\}$	

(63)

We prune by removing duplicates and non-maximal expressions, and get

$$\begin{aligned} &\text{from } \Delta_1\text{: } \{M_1 + 1, 2M_1 - 1, 2M_2 - 1\}, \\ &\text{from } \Delta_2\text{: } \{M_2 + 1, M_1 + M_2 - 1\}. \end{aligned} \tag{64}$$

Step M3 (Combine Expressions and Compute Relations for M_i). For each Δ_i separately, equate all possible combinations of two elements. Construct relations between the M_i by solving for one M_i.

Example. Combining the expressions in (64), we get

$$\begin{array}{l|l} \Delta_1 & \Delta_2 \\ \hline M_1+1=2M_1-1 & M_2+1=M_1+M_2-1 \\ M_1+1=2M_2-1 & \\ 2M_1-1=2M_2-1 & \end{array} \tag{65}$$

We construct relations between the M_i by solving for M_1 (in this case):

$$\begin{array}{l|l} \Delta_1 & \Delta_2 \\ \hline M_1=2 & M_1=2 \\ M_1=2M_2-2 & \\ M_1=M_2 & \end{array} \tag{66}$$

Step M4 (Combine Relations and Solve for Exponents M_i). By combining the lists of expressions in an outer product like fashion, we generate all the possible linear equations for M_i. Solving this linear system, we form a list of all the possible solutions for M_i.

Example. Combining the equations in Δ_1 and Δ_2, we obtain

$$\{M_1=2, M_1=2\},\ \{M_1=2, M_1=M_2\},\ \{M_1=2, M_1=2M_2-2\}. \tag{67}$$

Solving, we find

$$\begin{cases} M_1=2 \\ M_2=2 \end{cases} \qquad \begin{cases} M_1=2 \\ M_2=\text{Free}. \end{cases} \tag{68}$$

Step M5 (Discard Invalid Exponents M_i). The solutions are substituted into the unpruned list of exponents (in Step M2). For every solution (without free exponents) we test whether or not there is a highest-power balance between at least two different tracking variables. If not, the solution is rejected. Non-positive, fractional, and complex exponents are discarded (after showing them to the user). Negative exponents ($M_i=-p_i$) and fractional exponents ($M_i=p_i/q_i$) indicate that a change of dependent variables ($u_1=\tilde{u_i}^{-p_i}$ or $u_i=\tilde{u_i}^{1/q_i}$) should be attempted in (3). Presently, such nonlinear transformations are only carried out automatically for single equations.

Example. Removing the case $\{M_1=2, M_2=\text{Free}\}$ from (68), we substitute $\{M_1=2, M_2=2\}$ into (63). Leading exponent (3 in this case) occurs for `Tr[2]`, `Tr[3]` and `Tr[4]` in Δ_1, and for `Tr[6]` and `Tr[7]` in Δ_2. The solution is accepted.

Step M6 (Fix Undetermined M_i and Generate Additional Solutions). When some solutions involve one or more arbitrary M_i we produce candidate solutions with a countdown procedure and later reject invalid candidates.

Based on the outcome of Step M5, scan for freedom in one or more of M_i by gathering the highest-exponent expressions from the unpruned list in Step M2. If the dominant expressions are free of any M_i, a countdown mechanism generates valid integer values for those M_i. These values of M_i must not exceed those computed in Step M5. Candidate solutions are tested (and rejected, if necessary) by the procedure given in Step M5.

Example. The dominant expressions from (63) with $\{M_1 = 2, M_2 = 2\}$ are

Δ_1	Δ_2
`Tr[2]`: $\{2M_1 - 1\}$	`Tr[6]`: $\{M_1 + M_2 - 1\}$
`Tr[3]`: $\{2M_2 - 1\}$	`Tr[7]`: $\{M_2 + 1\}$
`Tr[4]`: $\{M_1 + 1\}$	

(69)

Substituting $M_1 = 2$, the highest exponent (3 in this case) matches for `Tr[2]` and `Tr[4]` in Δ_1 when $M_2 \leq 2$. The highest exponent $(M_2 + 1)$ matches for `Tr[6]` and `Tr[7]` in Δ_2.

A countdown mechanism then generates the following list of candidates:

$$\begin{cases} M_1 = 1 \\ M_2 = 1 \end{cases} \qquad \begin{cases} M_1 = 1 \\ M_2 = 2 \end{cases} \qquad \begin{cases} M_1 = 2 \\ M_2 = 1 \end{cases} \qquad \begin{cases} M_1 = 2 \\ M_2 = 2. \end{cases} \tag{70}$$

Verifying these candidate solutions, we are left with

$$\begin{cases} M_1 = 2 \\ M_2 = 1 \end{cases} \qquad \begin{cases} M_1 = 2 \\ M_2 = 2. \end{cases} \tag{71}$$

Notice that for the new solution $\{M_1 = 2, M_2 = 1\}$ only the exponents corresponding to `Tr[2]` and `Tr[4]` in Δ_1 are equal.

Currently, for the mixed tanh–sech method, the code sets $M_i = 2$ and $N_i = 1$.

6.2. Algorithm to analyze and solve nonlinear algebraic systems

In this section, we detail our algorithm to analyze and solve nonlinear parameterized algebraic systems (as generated in step 3 of the main algorithms). Our solver is custom designed for systems that are (initially) polynomial in the primary unknowns (a_{ij}), the secondary unknowns (c_i), and parameters $(m, \alpha, \beta, \gamma, \ldots)$.

The goal is to compute the coefficients a_{ij} in terms of the wavenumbers c_i and the parameters m, α, β, etc. In turn, the c_i must be solved in terms of these parameters. Possible compatibility conditions for the parameters (relations amongst them or specific values for them) must be added to the solutions.

Algebraic systems are solved recursively, starting with the simplest equation, and continually back-substituting solutions. This process is repeated until the system is completely solved.

To guide the recursive process, we designed functions to: (i) factor, split, and simplify the equations; (ii) sort the equations according to their complexity; (iii) solve the equations for sorted unknowns; (iv) substitute solutions into the remaining equations; and (v) collect the solution branches and constraints.

This strategy is similar to what one would do by hand. If there are numerous parameters in the system or if it is of high degree, there is no guarantee that our solver will return a suitable result, let alone a complete result.

Step R1 (Split and Simplify Each Equation). For all but the mixed tanh–sech algorithm, we assume that the coefficients a_{iM_i} of the highest power terms are nonzero and that c_i, m, α, β, etc. are nonzero. For the mixed sech–tanh method, $a_{iM_i} = a_{i2}$ and $b_{iN_i} = b_{i1}$ are allowed to be zero.

We first factor equations and set admissible factors equal to zero (after clearing possible exponents). For example, $\{\phi_1\phi_2^3\phi_3^2 = 0\} \to \{\phi_1 = 0, \phi_2 = 0, \phi_3 = 0\}$, where ϕ_i is a polynomial in primary and secondary unknowns along with the parameters. Equations where non-zero expressions are set to zero are disgarded.

Example. Consider (34), which was derived in the search for sech-solutions of (22) for the case $M_1 = M_2 = 2$. Taking a_{12}, a_{22}, c_1, c_2, α, β, to be nonzero, splitting equations, and removing non-zero factors leads to

$$\begin{aligned}
&a_{12} - 4c_1^2 = 0,\\
&a_{21} = 0 \vee (3a_{10}c_1 + c_1^3 + c_2) = 0,\\
&a_{12}a_{21} + 2a_{11}a_{22} - 2a_{21}c_1^2 = 0,\\
&3\alpha a_{11}a_{12} - \beta a_{21}a_{22} - \alpha a_{11}c_1^2 = 0,\\
&3\alpha a_{12}^2 - \beta a_{22}^2 - 6\alpha a_{12}c_1^2 = 0,\\
&6\alpha a_{10}a_{11}c_1 - 2\beta a_{20}a_{21}c_1 + \alpha a_{11}c_1^3 - a_{11}c_2 = 0,\\
&3a_{11}a_{21}c_1 + 6a_{10}a_{22}c_1 + 8a_{22}c_1^3 + 2a_{22}c_2 = 0,\\
&3\alpha a_{11}^2c_1 + 6\alpha a_{10}a_{12}c_1 - \beta a_{21}^2c_1 - 2\beta a_{20}a_{22}c_1 + 4\alpha a_{12}c_1^3 - a_{12}c_2 = 0,
\end{aligned} \tag{72}$$

where $\vee$ is the logical or.

Step R2 (Sort Equations According to Complexity). A heuristic measure of complexity is assigned to each ϕ_i by computing a weighted sum of the degrees of nonlinearity in the primary and secondary unknowns, parameters, and the length of ϕ_i. Linear and quasi-linear equations (with products like $a_{11}a_{21}$) are of lower complexity than polynomial equations of higher degree or non-polynomial equations. Solving the equation of the lowest complexity first, forestalls branching, avoids expression swell, and conserves memory.

Example. Sorting (72), we get

$$\begin{aligned}
&a_{12} - 4c_1^2 = 0,\\
&3\alpha a_{11}a_{12} - \beta a_{21}a_{22} - \alpha a_{11}c_1^2 = 0,\\
&a_{12}a_{21} + 2a_{11}a_{22} - 2a_{21}c_1^2 = 0,\\
&a_{21} = 0 \vee (3a_{10}c_1 + c_1^3 + c_2) = 0,\\
&3\alpha a_{12}^2 - \beta a_{22}^2 - 6\alpha a_{12}c_1^2 = 0,\\
&6\alpha a_{10}a_{11}c_1 - 2\beta a_{20}a_{21}c_1 + \alpha a_{11}c_1^3 - a_{11}c_2 = 0,\\
&3a_{11}a_{21}c_1 + 6a_{10}a_{22}c_1 + 8a_{22}c_1^3 + 2a_{22}c_2 = 0,\\
&3\alpha a_{11}^2c_1 + 6\alpha a_{10}a_{12}c_1 - \beta a_{21}^2c_1 - 2\beta a_{20}a_{22}c_1 + 4\alpha a_{12}c_1^3 - a_{12}c_2 = 0.
\end{aligned} \tag{73}$$

Step R3 (Solve Equations for Ordered Unknowns). The ordering of unknowns is of paramount importance. The unknowns from the first equation from Step R2 are ordered so that the lowest exponent primary-unknowns precede the primary-unknowns that the equation is not polynomial in. If there are not any primary-unknowns, the lowest exponent secondary-unknowns precede the secondary-unknowns that the equation is not

polynomial in. Likewise, in the absence of primary- or secondary-unknowns, the lowest exponent parameters precede the non-polynomial parameters.

The equation is solved using the built-in *Mathematica* function `Reduce`, which produces a list of solutions and constraints. Constraints of the form $a \neq b$ (where neither a or b is zero) are pruned, and the remaining constraints and solutions are collected.

Example. In this example, $a_{12} - 4c_1^2 = 0$ is solved for a_{12} and the solution $a_{12} = 4c_1^2$ is added to a list of solutions.

Step R4 (Recursively Solve the Entire System). The solutions and constraints from Step R3 are applied and added to the previously found solutions and constraints. In turn, all the solutions are then applied to the remaining equations. The updated system is simplified by clearing common denominators in each equation and continuing with the numerators. Steps R1–R4 are then repeated on the simplified system.

Example. Substituting $a_{12} = 4c_1^2$ and clearing denominators, we obtain

$$\begin{aligned}
&\beta a_{21}a_{22} - 11\alpha a_{11}c_1^2 = 0\\
&a_{11}a_{22} + a_{21}c_1^2 = 0,\\
&a_{21} = 0 \vee (3a_{10}c_1 + c_1^3 + c_2) = 0,\\
&\beta a_{22}^2 - 24\alpha c_1^4 = 0,\\
&6\alpha a_{10}a_{11}c_1 - 2\beta a_{20}a_{21}c_1 + \alpha a_{11}c_1^3 - a_{11}c_2 = 0,\\
&3a_{11}a_{21}c_1 + 6a_{10}a_{22}c_1 + 8a_{22}c_1^3 + 2a_{22}c_2 = 0,\\
&3\alpha a_{11}^2 - \beta a_{21}^2 - 2\beta a_{20}a_{22} + 24\alpha a_{10}c_1^2 + 16\alpha c_1^4 - 4c_1c_2 = 0.
\end{aligned} \tag{74}$$

The recursive process terminates when the system is completely solved. The solutions (including possible constraints) are returned.

Repeating Steps R1–R4 seven more times the *global* solution of (34) is obtained:

$$\begin{aligned}
&a_{10} = -(4c_1^3 + c_2)/(3c_1),\, a_{11} = 0,\, a_{12} = 4c_1^2,\\
&a_{20} = \pm(4\alpha c_1^3 + (1+2\alpha)c_2)/(c_1\sqrt{6\alpha\beta}),\, a_{21} = 0,\, a_{22} = \mp 2c_1^2\sqrt{6\alpha/\beta}
\end{aligned} \tag{75}$$

where c_1, c_2, α and β are arbitrary.

This solution of (33), corresponds to the $M_1 = 2$, $M_2 = 1$ case given in (35).

6.3. Algorithm to build and test solutions

The solutions to the algebraic system found in Section 6.2 are substituted into

$$u_i(\mathbf{x}) = \sum_{j=0}^{M_i} a_{ij}F^j(\xi) + \sqrt{R(F)}\sum_{j=0}^{N_i} b_{ij}F^j(\xi), \tag{76}$$

where F and $R(F)$ are defined in Section 6.1. The constraints on the parameters (m, α, β, etc.) are also collected and applied to system (3).

Since the algorithm used to solve the nonlinear algebraic system continually clears denominators, it is important to test the final solutions for u_i. While *Mathematica's* `Reduce`

function generates constraints that should prevent any undetermined or infinite coefficients a_{ij} after back-substitution, it is still prudent to check the solutions.

To present solutions in the simplest format, we assume that all parameters (c_i, m, α, β, etc.) are positive, real numbers. This allows us to repeatedly apply rules such as $\sqrt{\alpha^2} \to \alpha$, $\sqrt{-\alpha^2} \to i\alpha$, $\sqrt{-\beta} \to i\sqrt{\beta}$ and $\sqrt{-(c_1+c_2)^2} \to i(c_1+c_2)$.

We allow for two flavors of testing: a numeric test for complicated solutions and a symbolic test which guarantees the solution. In either test, we substitute the solutions into (3) after casting the solutions into exponential form, i.e., $\tanh\xi \to (\mathrm{e}^{\xi} - \mathrm{e}^{-\xi})/(\mathrm{e}^{\xi} + \mathrm{e}^{-\xi})$ and $\operatorname{sech}\xi \to 2/(\mathrm{e}^{\xi} + \mathrm{e}^{-\xi})$.

For the numeric test of solutions:

- after substituting the solution, substitute random real numbers in $[-1, 1]$ for x_i, c_i, and Δ in the left-hand side of (3),
- expand and factor the remaining expressions,
- substitute random real numbers in $[-1, 1]$ for arbitrary a_{ij}, b_{ij}, m, α, ...,
- expand and factor the remaining expressions,
- if the absolute value of each of the expressions $< \epsilon \approx$ MachinePrecision/2, then accept the solution as valid, else reject the solution (after showing it to the user).

Mathematica evaluates $\sqrt{a^2} \to a$ when a is numeric, but does not evaluate $\sqrt{a^2} \to |a|$ when a is symbolic. Our simplification routines use $\sqrt{a^2} = a$ instead of $|a|$ when a is symbolic. This has two consequences: (i) valid solutions may be missed, and (ii) solutions have a $1/2$ probability of evaluating to matching signs during the numeric test. The numeric test being inconclusive, we perform a symbolic test.

For the symbolic test of solutions:

- after substituting the solution, expand and factor the left-hand side of (3),
- apply simplification rules like $\sqrt{a^2} \to a$, $\sqrt{-a^2} \to ia$, $1 - \operatorname{sech}^2\xi \to \tanh^2\xi$, and $\operatorname{sn}^2(x; m) \to 1 - \operatorname{cn}^2(x; m)$,
- repeat the above simplifications until the expressions are static,
- if the final expressions are identically equal to zero, then accept the solution, else reject the solution and report the unresolved expressions to the user.

7. Examples of solitary wave solutions for ODEs and PDEs

The algorithms from Sections 2–5 were implemented in our *Mathematica* package `PDESpecialSolutions.m`, which was used to solve the equations in this section.

7.1. The Zakharov–Kuznetsov KdV-type equations

The KdV–Zakharov–Kuznetsov (KdV–ZK) equation,

$$u_t + \alpha u u_x + u_{xxx} + u_{xyy} + u_{xzz} = 0, \tag{77}$$

models ion-acoustic waves in magnetized multi-component plasmas including negative ions (see e.g. Das and Verheest, 1989).

With `PDESpecialSolutions` (Tanh and Sech options) we found the solution

$$\begin{aligned} u(x,y,z,t) &= \frac{8c_1(c_1^2+c_2^2+c_3^2)-c_4}{\alpha c_1} - \frac{12(c_1^2+c_2^2+c_3^2)}{\alpha}\tanh^2\xi, \\ &= -\frac{4c_1(c_1^2+c_2^2+c_3^2)+c_4}{\alpha c_1} + \frac{12(c_1^2+c_2^2+c_3^2)}{\alpha}\mathrm{sech}^2\,\xi, \end{aligned} \tag{78}$$

where $\xi = c_1x + c_2y + c_3z + c_4t + \Delta$, with c_1, c_2, c_3, c_4, Δ and α arbitrary.

For $c_2 = c_3 = 0$ and replacing c_4 by c_2, one gets the solitary wave solution

$$\begin{aligned} u(x,t) &= \frac{8c_1^3-c_2}{\alpha c_1} - \frac{12c_1^2}{\alpha}\tanh^2(c_1x+c_2t+\Delta), \\ &= -\frac{4c_1^3+c_2}{\alpha c_1} + \frac{12c_1^2}{\alpha}\mathrm{sech}^2(c_1x+c_2t+\Delta), \end{aligned} \tag{79}$$

of the ubiquitous KdV equation (48).

The function `PDESpecialSolutions` does not take boundary or initial conditions as input. One can *a posteriori* impose conditions on solutions. For example, requiring $\lim_{x\to\pm\infty} u(x,t) = 0$ in (79) would fix $c_2 = -4c_1^3$.

For the modified KdV–ZK equation (Das and Verheest, 1989),

$$u_t + \alpha u^2 u_x + u_{xxx} + u_{xyy} + u_{xzz} = 0, \tag{80}$$

using the Tanh and Sech options, `PDESpecialSolutions` returns

$$u(x,y,z,t) = \pm i\sqrt{6(c_1^2+c_2^2+c_3^2)/\alpha}\,\tanh\xi, \tag{81}$$

with $\xi = c_1x + c_2y + c_3z + 2c_1(c_1^2+c_2^2+c_3^2)t + \Delta$, and

$$u(x,y,z,t) = \pm\sqrt{6(c_1^2+c_2^2+c_3^2)/\alpha}\,\mathrm{sech}\,\xi, \tag{82}$$

with $\xi = c_1x + c_2y + c_3z - c_1(c_1^2+c_2^2+c_3^2)t + \Delta$. For $c_2 = c_3 = 0$, (81) and (82) reduce to the well-known solitary wave solutions

$$u(x,t) = \pm i c_1\sqrt{6/\alpha}\,\tanh(c_1x + 2c_1^3t + \Delta), \tag{83}$$

$$u(x,t) = \pm c_1\sqrt{6/\alpha}\,\mathrm{sech}(c_1x - c_1^3t + \Delta) \tag{84}$$

(c_1, Δ and α arbitrary real numbers) of the modified KdV (mKdV) equation (Ablowitz and Clarkson, 1991),

$$u_t + \alpha u^2 u_x + u_{xxx} = 0. \tag{85}$$

For a three-dimensional modified KdV (3D-mKdV) equation,

$$u_t + 6u^2 u_x + u_{xyz} = 0, \tag{86}$$

one obtains the solitary wave solution

$$u(x,y,z,t) = \pm\sqrt{c_2c_3}\,\mathrm{sech}(c_1x + c_2y + c_3z - c_1c_2c_3t + \Delta), \tag{87}$$

where c_1, c_2, c_3 and Δ are arbitrary.

7.2. The generalized Kuramoto–Sivashinsky equation

Consider the generalized Kuramoto–Sivashinsky (KS) equation (see e.g. Parkes and Duffy, 1996)

$$u_t + uu_x + u_{xx} + \alpha u_{xxx} + u_{xxxx} = 0. \tag{88}$$

Ignoring complex solutions, `PDESpecialSolutions` (Tanh option) automatically determines the special values of the real parameter α and the corresponding closed form solutions. For $\alpha = 4$,

$$u(x,t) = 9 \pm 2c_2 \pm 15 \tanh \xi - 15 \tanh^2 \xi \mp 15 \tanh^3 \xi, \tag{89}$$

with $\xi = \mp(1/2)x + c_2 t + \Delta$. For $\alpha = (12/\sqrt{47})$,

$$u(x,t) = \frac{45 \mp 4418c_2}{47\sqrt{47}} \pm \frac{45}{47\sqrt{47}} \tanh \xi - \frac{45}{47\sqrt{47}} \tanh^2 \xi \pm \frac{15}{47\sqrt{47}} \tanh^3 \xi, \tag{90}$$

where $\xi = \pm(1/2\sqrt{47})x + c_2 t + \Delta$. For $\alpha = (16/\sqrt{73})$,

$$\begin{aligned} u(x,t) = {} & \frac{2(30 \mp 5329c_2)}{73\sqrt{73}} \pm \frac{75}{73\sqrt{73}} \tanh \xi \\ & - \frac{60}{73\sqrt{73}} \tanh^2 \xi \pm \frac{15}{73\sqrt{73}} \tanh^3 \xi, \end{aligned} \tag{91}$$

where $\xi = \pm(1/2\sqrt{73})x + c_2 t + \Delta$.

The remaining solutions produced by `PDESpecialSolutions` are either complex (not shown here) or can be obtained from the solutions above via the inversion symmetry of (88): $u \to -u$, $x \to -x$, $\alpha \to -\alpha$.

A separate run of the code after setting $\alpha = 0$ in (88) yields

$$u(x,t) = -2\sqrt{\frac{19}{11}}c_2 - \frac{135}{19}\sqrt{\frac{11}{19}} \tanh \xi + \frac{165}{19}\sqrt{\frac{11}{19}} \tanh^3 \xi, \tag{92}$$

with $\xi = (1/2)\sqrt{11/19}x + c_2 t + \Delta$. In all the solutions above c_2 is arbitrary.

7.3. Coupled KdV equations

In Section 3 we gave the sech-solutions for the Hirota–Satsuma system (22). Here we list the tanh, cn and sn solutions for (22) computed by `PDESpecialSolutions` (Tanh, JacobiCN and JacobiSN options):

$$\begin{aligned} u(x,t) &= \frac{2c_1^3 - c_2}{3c_1} - 2c_1^2 \tanh^2(\xi), \\ v(x,t) &= \pm\sqrt{[8\alpha c_1^4 + 2(1+2\alpha)c_1c_2]/\beta} \tanh(\xi), \\ u(x,t) &= \frac{8c_1^3 - c_2}{3c_1} - 4c_1^2 \tanh^2(\xi), \\ v(x,t) &= \pm\frac{8\alpha c_1^3 - (1+2\alpha)c_2}{c_1\sqrt{6\alpha\beta}} \mp 2c_1^2\sqrt{6\alpha/\beta} \tanh^2(\xi), \end{aligned} \tag{93}$$

$$
\begin{aligned}
u(x,t) &= \frac{(1+m)c_1^3 - c_2}{3c_1} - 2mc_1^2\mathrm{sn}^2(\xi;m),\\
v(x,t) &= \pm\sqrt{[4\alpha m(1+m)c_1^4 + 2(1+2\alpha)mc_1c_2]/\beta}\,\mathrm{sn}(\xi,m),\\
u(x,t) &= \frac{4(1+m)c_1^3 - c_2}{3c_1} - 4mc_1^2\mathrm{sn}^2(\xi;m),\\
v(x,t) &= \pm\frac{4\alpha(1+m)c_1^3 - (1+2\alpha)c_2}{c_1\sqrt{6\alpha\beta}} \mp 2c_1^2\sqrt{6\alpha/\beta}\,\mathrm{sn}^2(\xi;m),
\end{aligned}
\tag{94}
$$

$$
\begin{aligned}
u(x,t) &= \frac{(1-2m)c_1^3 - c_2}{3c_1} + 2mc_1^2\mathrm{cn}^2(\xi;m),\\
v(x,t) &= \pm\sqrt{[4\alpha m(2m-1)c_1^4 - 2(1+2\alpha)mc_1c_2]/\beta}\,\mathrm{cn}(\xi;m),\\
u(x,t) &= \frac{4(1-2m)c_1^3 - c_2}{3c_1} + 4mc_1^2\mathrm{cn}^2(\xi;m),\\
v(x,t) &= \pm\frac{4\alpha(1-2m)c_1^3 - (1+2\alpha)c_2}{c_1\sqrt{6\alpha\beta}} \pm 2c_1^2\sqrt{6\alpha/\beta}\,\mathrm{cn}^2(\xi;m),
\end{aligned}
\tag{95}
$$

with $\xi = c_1x + c_2t + \Delta$, and c_1, c_2, α, β, Δ, and modulus m arbitrary. These solutions correspond with those given in Fan and Hon (2002).

With the SechTanh option we obtained two dozen (real and complex) solutions. The real solutions coincide with the ones given above.

Another coupled system of KdV-type equations was studied by Guha-Roy (1987)

$$
\begin{aligned}
&u_t + \alpha v v_x + \beta u u_x + \gamma u_{xxx} = 0,\\
&v_t + \delta(uv)_x + \epsilon v v_x = 0,
\end{aligned}
\tag{96}
$$

where α through ϵ are real constants. The package `PDESpecialSolutions` (Sech option) computed:

$$
\begin{aligned}
u(x,t) &= -\frac{4\epsilon^2\gamma c_1^3 + (4\alpha\delta + \epsilon^2)c_2}{Ac_1} + \frac{12\epsilon^2\gamma c_1^2}{A}\mathrm{sech}^2(c_1x + c_2t + \Delta),\\
v(x,t) &= \frac{2\epsilon[4\delta\gamma c_1^3 + (\delta-\beta)c_2]}{Ac_1} - \frac{24\delta\epsilon\gamma c_1^2}{A}\mathrm{sech}^2(\xi),
\end{aligned}
\tag{97}
$$

where $\xi = c_1x + c_2t + \Delta$, $A = 4\alpha\delta^2 + \beta\epsilon^2$, with c_1, c_2, Δ and α through ϵ arbitrary. For $\epsilon = 0$, (96) reduces to Kawamoto's system; for $\epsilon = 0$, $\delta = -2$ to Ito's system. Neither of these systems has polynomial solutions in sech or tanh.

7.4. The Fisher and FitzHugh–Nagumo equations

For the Fisher equation (Malfliet, 1992),

$$
u_t - u_{xx} - u(1-u) = 0, \tag{98}
$$

with `PDESpecialSolutions` (Tanh option) we found the (real) solution

$$u(x,t) = \tfrac{1}{4} \pm \tfrac{1}{2}\tanh\xi + \tfrac{1}{4}\tanh^2\xi, \tag{99}$$

with $\xi = \pm(1/2\sqrt{6})x \pm (5/12)t + \Delta$. In addition, there are four complex solutions.

Obviously, `PDESpecialSolutions` handles ODEs also. For example, we can put the FitzHugh–Nagumo (FHN) equation (Hereman, 1990),

$$u_t - u_{xx} + u(1-u)(\alpha - u) = 0, \tag{100}$$

where $-1 \le \alpha < 1$, into a travelling frame,

$$\beta v_z + \sqrt{2}v_{zz} - \sqrt{2}v(1-\sqrt{2}v)(\alpha - \sqrt{2}v) = 0, \tag{101}$$

with $v(z) = v(x - (\beta/\sqrt{2})t) = \sqrt{2}u(x,t)$. Ignoring the inversion symmetry $z \to -z$, $\beta \to -\beta$ of (101), we find with `PDESpecialSolutions` (Tanh option)

$$v(z) = \frac{1}{2\sqrt{2}}\left[\beta + (\beta-2)\tanh\left[\frac{\sqrt{2}}{4}(2-\beta)z + \Delta\right]\right], \tag{102}$$

if $\alpha = \beta - 1$;

$$v(z) = \frac{(\beta+2)}{2\sqrt{2}}\left[1 - \tanh\left[\frac{\sqrt{2}}{4}(\beta+2)z + \Delta\right]\right], \tag{103}$$

if $\alpha = \beta + 2$; and

$$v(z) = \frac{1}{2\sqrt{2}}\left[1 + \tanh\left[\frac{\sqrt{2}}{4}z + \Delta\right]\right], \tag{104}$$

if $\alpha = (1/2)(\beta + 1)$. In these solutions (see e.g. Hereman, 1990) β and Δ are arbitrary.

7.5. A degenerate Hamiltonian system

Gao and Tian (2001) considered the following degenerate Hamiltonian system,

$$\begin{aligned} &u_t - u_x - 2v = 0, \\ &v_t - 2\epsilon uv = 0, \qquad \epsilon = \pm 1, \end{aligned} \tag{105}$$

which was shown to be completely integrable by admitting infinitely many conserved densities. Our code does not find sech-solutions. With the SechTanh option, `PDESpecialSolutions` returns the solutions:

$$\begin{aligned} &u(x,t) = -\epsilon c_2 \tanh\xi, \\ &v(x,t) = \tfrac{1}{2}\epsilon c_2(c_1 - c_2)\mathrm{sech}^2\,\xi, \end{aligned} \tag{106}$$

which could have been obtained with the tanh-method in Section 2; and

$$\begin{aligned} &u(x,t) = \tfrac{1}{2}i c_2\epsilon(\mathrm{sech}\,\xi + i\tanh\xi), \\ &v(x,t) = \tfrac{1}{4}c_2(c_1 - c_2)\epsilon\,\mathrm{sech}\,\xi(\mathrm{sech}\,\xi + i\tanh\xi), \end{aligned} \tag{107}$$

plus their two complex conjugates. There are no constraints on c_1, c_2, and ϵ, and $\xi = c_1x + c_2t + \Delta$. The above solutions were reported in Gao and Tian (2001).

7.6. The combined KdV–mKdV equation

The combined KdV–mKdV equation (see Gao and Tian, 2001)

$$u_t + 6\alpha u u_x + 6\beta u^2 u_x + \gamma u_{xxx} = 0, \tag{108}$$

describes a variety of wave phenomena in plasma, solid state, and quantum physics. We chose this example to show that ODEs of type (27), which are free of $\sqrt{1-S^2}$, can admit mixed tanh–sech solutions.

First, `PDESpecialSolutions` with the Tanh option, produces

$$u(x,t) = -\frac{\alpha}{2\beta} \pm i\sqrt{\frac{\gamma}{\beta}}c_1 \tanh\left(c_1 x + \frac{c_1}{2\beta}(3\alpha^2 + 4\beta\gamma c_1^2)t + \Delta\right). \tag{109}$$

Next, with the Sech option, `PDESpecialSolutions` computes

$$u(x,t) = -\frac{\alpha}{2\beta} \pm \sqrt{\frac{\gamma}{\beta}}c_1 \mathrm{sech}\left[c_1 x + \frac{c_1}{2\beta}(3\alpha^2 - 2\beta\gamma c_1^2)t + \Delta\right]. \tag{110}$$

Third, with the SechTanh option, `PDESpecialSolutions` finds

$$u(x,t) = -\frac{\alpha}{2\beta} + \frac{1}{2}\sqrt{\frac{\gamma}{\beta}}c_1(\mathrm{sech}\,\xi \pm i\tanh\xi), \tag{111}$$

and

$$u(x,t) = -\frac{\alpha}{2\beta} - \frac{1}{2}\sqrt{\frac{\gamma}{\beta}}c_1(\mathrm{sech}\,\xi \mp i\tanh\xi), \tag{112}$$

where $\xi = c_1 x + (1/2)(c_1/\beta)(3\alpha^2 + \beta\gamma c_1^2)t + \Delta$. In all solutions c_1, Δ, α, β and γ are arbitrary. The solutions were reported in Gao and Tian (2001), although there were minor misprints.

7.7. The Duffing equation

Duffing's equation (Lawden, 1989),

$$u'' + u + \alpha u^3 = 0, \tag{113}$$

models a nonlinear spring problem. Its cn and sn solutions

$$\begin{aligned} u(x) &= \pm\sqrt{\frac{2m}{(1-2m)\alpha}}\,\mathrm{cn}\left(\frac{\epsilon x}{\sqrt{1-2m}} + \Delta; m\right), \qquad \epsilon = \pm 1, \\ u(x) &= \pm\sqrt{\frac{-2m}{(1+m)\alpha}}\,\mathrm{sn}\left(\frac{\epsilon x}{\sqrt{1+m}} + \Delta; m\right), \qquad \epsilon = \pm 1, \end{aligned} \tag{114}$$

are computed by `PDESpecialSolutions` with the JacobiCN and JacobiSN options. There are four sign combinations in (114). Since $0 \le m \le 1$, the cn solution is real when $\alpha > 0$ and $m < 1/2$. The sn solution is real for $\alpha < 0$. Such conditions are not automatically generated. During simplifications the code assumes $\alpha > 0$ (see Section 6.2 for details).

Initial conditions fix the modulus in (114). For example, $u(0) = a$ and $\dot{u}(0) = 0$ lead to $u(x) = a\,\mathrm{cn}(\sqrt{1+\alpha a^2}x;\,(\alpha a^2)/(2+2\alpha a^2))$.

7.8. A class of fifth-order PDEs with three parameters

To illustrate the limitations of `PDESpecialSolutions` consider the family of fifth-order KdV equations (Göktaş and Hereman, 1997),

$$u_t + \alpha u^2 u_x + \beta u_x u_{xx} + \gamma u u_{xxx} + u_{xxxxx} = 0, \tag{115}$$

where α, β, and γ are nonzero parameters.

An investigation of the scaling properties of (115) reveals that only the ratios α/γ^2 and β/γ are important, but let us proceed with (115).

7.8.1. Special cases

Several special cases of (115) are well known (for references see Göktaş and Hereman, 1997). Indeed, for $\alpha = 30$, $\beta = 20$, $\gamma = 10$, Eq. (115) reduces to

$$u_t + 30u^2 u_x + 20u_x u_{xx} + 10u u_{xxx} + u_{xxxxx} = 0, \tag{116}$$

which belongs to the completely integrable hierarchy of higher-order KdV equations constructed by Lax. Eq. (116) has two tanh-solutions:

$$u(x,t) = 4c_1^2 - 6c_1^2 \tanh^2(c_1 x - 56c_1^5 t + \Delta), \tag{117}$$

and

$$u(x,t) = a_{10} - 2c_1^2 \tanh^2[c_1 x - 2(15a_{10}^2 c_1 - 40a_{10}c_1^3 + 28c_1^5)t + \Delta], \tag{118}$$

where a_{10}, c_1, Δ are arbitrary.

For $\alpha = \beta = \gamma = 5$, one obtains the equation,

$$u_t + 5u^2 u_x + 5u_x u_{xx} + 5u u_{xxx} + u_{xxxxx} = 0, \tag{119}$$

due to Sawada and Kotera (SK) and Dodd and Gibbon, which has tanh-solutions

$$u(x,t) = 8c_1^2 - 12c_1^2 \tanh^2(c_1 x - 16c_1^5 t + \Delta), \tag{120}$$

and

$$u(x,t) = a_{10} - 6c_1^2 \tanh^2[c_1 x - (5a_{10}^2 c_1 - 40a_{10}c_1^3 + 76c_1^5)t + \Delta], \tag{121}$$

where a_{10}, c_1, Δ are arbitrary.

The KK equation due to Kaup and Kupershmidt,

$$u_t + 20u^2 u_x + 25u_x u_{xx} + 10u u_{xxx} + u_{xxxxx} = 0, \tag{122}$$

corresponding to $\alpha = 20$, $\beta = 25$, $\gamma = 10$, and again admits two tanh-solutions:

$$u(x,t) = c_1^2 - \tfrac{3}{2}c_1^2 \tanh^2(c_1 x - c_1^5 t + \Delta), \tag{123}$$

and

$$u(x,t) = 8c_1^2 - 12c_1^2 \tanh^2(c_1 x - 176c_1^5 t + \Delta), \tag{124}$$

with c_1, Δ arbitrary, but no additional arbitrary coefficients.

The equation

$$u_t + 2u^2u_x + 6u_xu_{xx} + 3uu_{xxx} + u_{xxxxx} = 0, \tag{125}$$

for $\alpha = 2$, $\beta = 6$, $\gamma = 3$, was studied by Ito. It has one tanh solution:

$$u(x,t) = 20c_1^2 - 30c_1^2\tanh^2(c_1x - 96c_1^5t + \Delta), \tag{126}$$

again with c_1 and Δ arbitrary. `PDESpecialSolutions` (Tanh option) produces all these solutions.

7.8.2. General case

Eq. (115) is hard to analyze by hand or using a computer. After a considerable amount of time, `PDESpecialSolutions` (Tanh option) produced the solutions given below (but not in as nice a form). Our write-up of the solutions is the result of additional interactive work with *Mathematica*.

The coefficients a_{10}, a_{11}, and a_{12} in

$$u(x,t) = a_{10} + a_{11}\tanh(\xi) + a_{12}\tanh^2(\xi), \tag{127}$$

with $\xi = c_1x + c_2 + \Delta$, must satisfy the following nonlinear algebraic system with parameters c_1, c_2, α, β, and γ:

$$\begin{aligned}
&\alpha a_{12}^2 + 6\beta a_{12}c_1^2 + 12\gamma a_{12}c_1^2 + 360c_1^4 = 0,\\
&a_{11}(\alpha a_{12}^2 + 2\beta a_{12}c_1^2 + 6\gamma a_{12}c_1^2 + 24c_1^4) = 0,\\
&a_{11}(\alpha a_{10}^2c_1 - 2\gamma a_{10}c_1^3 + 2\beta a_{12}c_1^3 + 16c_1^5 + c_2) = 0,\\
&a_{11}(\alpha a_{11}^2 + 6\alpha a_{10}a_{12} + 6\gamma a_{10}c_1^2 - 12\beta a_{12}c_1^2 - 18\gamma a_{12}c_1^2 - 120c_1^4) = 0,\\
&2\alpha a_{11}^2a_{12} + 2\alpha a_{10}a_{12}^2 + \beta a_{11}^2c_1^2 + 3\gamma a_{11}^2c_1^2 + 12\gamma a_{10}a_{12}c_1^2\\
&\qquad - 8\beta a_{12}^2c_1^2 - 8\gamma a_{12}^2c_1^2 - 480a_{12}c_1^4 = 0,\\
&\alpha a_{10}a_{11}^2c_1 + \alpha a_{10}^2a_{12}c_1 - \beta a_{11}^2c_1^3 - \gamma a_{11}^2c_1^3 - 8\gamma a_{10}a_{12}c_1^3 + 2\beta a_{12}^2c_1^3\\
&\qquad + 136a_{12}c_1^5 + a_{12}c_2 = 0.
\end{aligned} \tag{128}$$

Assuming nonzero a_{12}, c_1, c_2, α, β, and γ, two cases must be distinguished:

Case 1. $a_{11} = 0$. In turn, this case splits into two sub-cases:

Case 1a.

$$a_{11} = 0, \qquad a_{12} = -\tfrac{3}{2}a_{10}, \qquad c_2 = c_1^3(24c_1^2 - \beta a_{10}), \tag{129}$$

where a_{10} must be one of the roots of

$$\alpha a_{10}^2 - 4\beta a_{10}c_1^2 - 8\gamma a_{10}c_1^2 + 160c_1^4 = 0. \tag{130}$$

Case 1b.

$$\begin{aligned}
&a_{11} = 0, \qquad a_{12} = -\frac{6\gamma}{\alpha}c_1^2,\\
&c_2 = -\frac{1}{\alpha}(\alpha^2a_{10}^2c_1 - 8\alpha\gamma a_{10}c_1^3 + 16\alpha c_1^5 + 12\gamma^2c_1^5),
\end{aligned} \tag{131}$$

provided that

$$\beta = \frac{1}{\gamma}(10\alpha - \gamma^2). \tag{132}$$

Case 2. $a_{11} \neq 0$. Then

$$\alpha = \frac{1}{392}(8\beta^2 + 38\beta\gamma + 39\gamma^2), \qquad a_{12} = -\frac{168}{2\beta + 3\gamma}c_1^2, \tag{133}$$

provided β is one of the roots of

$$(104\beta^2 + 886\beta\gamma + 1487\gamma^2)(520\beta^3 + 2158\beta^2\gamma - 1103\beta\gamma^2 - 8871\gamma^3) = 0. \tag{134}$$

Thus, case 2 also splits into two sub-cases:
Case 2a. If $\beta^2 = -(1/104)(886\beta\gamma + 1487\gamma^2)$, then

$$\begin{aligned} &\alpha = -\frac{1}{26}(2\beta + 5\gamma)\gamma, \qquad a_{10} = -\frac{52(4378\beta + 9983\gamma)}{7\gamma(958\beta + 2213\gamma)}c_1^2, \\ &a_{11} = \pm\frac{336}{2\beta + 3\gamma}c_1^2, \qquad a_{12} = -\frac{168}{2\beta + 3\gamma}c_1^2, \\ &c_2 = -\frac{364(1634\beta + 3851\gamma)}{2946\beta + 6715\gamma}c_1^5 \end{aligned} \tag{135}$$

where β is any root of $104\beta^2 + 886\beta\gamma + 1487\gamma^2 = 0$.
Case 2b. If $\beta^3 = (1/520)(1103\beta\gamma^2 + 8871\gamma^3 - 2158\beta^2\gamma)$, then

$$\begin{aligned} &\alpha = \frac{1}{392}(8\beta^2 + 38\beta\gamma + 39\gamma^2), \\ &a_{10} = \frac{28(1066\beta^2 + 5529\beta\gamma + 6483\gamma^2)}{(2\beta + 3\gamma)(6\beta + 23\gamma)(26\beta + 81\gamma)}c_1^2, \\ &a_{11}^2 = \frac{28\,224(26\beta - 17\gamma)(4\beta - \gamma)}{(2\beta + 3\gamma)^2(6\beta + 23\gamma)(26\beta + 81\gamma)}c_1^4, \qquad a_{12} = -\frac{168}{2\beta + 3\gamma}c_1^2, \\ &c_2 = -\frac{8(188\,900\,114\beta^2 + 1161\,063\,881\beta\gamma + 1792\,261\,977\gamma^2)}{105\,176\,786\beta^2 + 632\,954\,969\beta\gamma + 959\,833\,473\gamma^2}c_1^5, \end{aligned} \tag{136}$$

where β is any root of $520\beta^3 + 2158\beta^2\gamma - 1103\beta\gamma^2 - 8871\gamma^3 = 0$.

8. Other algorithms and related software

8.1. Other perspectives and potential generalizations

The algorithms presented in this article can be extended in several ways. For instance, one could modify the chain rule in Step T1 (S1, T1, or CN1) to compute other *types* of solutions or consider more complicated polynomials than those used in Step T2 (S2, T2, or CN2). Both options could be used together.

With respect to the first option, it suffices to know the underlying first-order differential equation of the desired fundamental function in the polynomial solution.

Table 2
Functions with corresponding ODEs and chain rules. $\mathcal{P}(x; g_2, g_3)$ is the Weierstrass function with invariants g_2 and g_3

Function	Symbol	ODE ($y' = \mathrm{d}y/\mathrm{d}\xi$)	Chain rule
$\tanh(\xi)$	T	$y' = 1 - y^2$	$\frac{\partial \bullet}{\partial x_j} = c_j(1 - \mathrm{T}^2)\frac{\mathrm{d}\bullet}{\mathrm{dT}}$
$\mathrm{sech}(\xi)$	S	$y' = -y\sqrt{1 - y^2}$	$\frac{\partial \bullet}{\partial x_j} = -c_j \mathrm{S}\sqrt{1 - \mathrm{S}^2}\frac{\mathrm{d}\bullet}{\mathrm{dS}}$
$\tan(\xi)$	TAN	$y = 1 + y^2$	$\frac{\partial \bullet}{\partial x_j} = c_j(1 + \mathrm{TAN}^2)\frac{\mathrm{d}\bullet}{\mathrm{dTAN}}$
$\exp(\xi)$	E	$y' = y$	$\frac{\partial \bullet}{\partial x_j} = c_j \mathrm{E}\frac{\mathrm{d}\bullet}{\mathrm{dE}}$
$\mathrm{cn}(\xi; m)$	CN	$y' = -\sqrt{(1 - y^2)(1 - m + my^2)}$	$\frac{\partial \bullet}{\partial x_j} = -c_j\sqrt{(1 - \mathrm{CN}^2)(1 - m + m\mathrm{CN}^2)}\frac{\mathrm{d}\bullet}{\mathrm{dCN}}$
$\mathrm{sn}(\xi; m)$	SN	$y' = \sqrt{(1 - y^2)(1 - my^2)}$	$\frac{\partial \bullet}{\partial x_j} = c_j\sqrt{(1 - \mathrm{SN}^2)(1 - m\mathrm{SN}^2)}\frac{\mathrm{d}\bullet}{\mathrm{dSN}}$
$\mathcal{P}(\xi; g_2, g_3)$	P	$y' = \pm\sqrt{4y^3 - g_2 y - g_3}$	$\frac{\partial \bullet}{\partial x_j} = \pm c_j\sqrt{4y^3 - g_2 y - g_3}\frac{\mathrm{d}\bullet}{\mathrm{dP}}$

Table 2 summarizes some of the more obvious choices. Several researchers, including Fan (2002a,b,c) and Gao and Tian (2001), seek solutions of the form

$$u_i(x, t) = U_i(\xi) = \sum_{j=1}^{M_i} a_{ij} w(\xi)^j, \qquad \xi = c_1 x + c_2 t + \Delta, \tag{137}$$

where $w(\xi)$ is constrained by a Riccati equation,

$$w'(\xi) = b + \epsilon w^2(\xi), \qquad \epsilon = \pm 1, \qquad b \text{ real constant.} \tag{138}$$

Ignoring rational solutions, (138) has the following solutions:

$$\begin{aligned} w(\xi) &= a\tanh(a\xi + c), && \text{if } \epsilon = -1,\ b = a^2, \\ w(\xi) &= a\coth(a\xi + c), && \text{if } \epsilon = -1,\ b = a^2, \\ w(\xi) &= a\tan(a\xi + c), && \text{if } \epsilon = 1,\ b = a^2, \\ w(\xi) &= a\cot(a\xi + c), && \text{if } \epsilon = -1,\ b = -a^2. \end{aligned} \tag{139}$$

So, (137) is polynomial in $\tanh\xi$, $\tan\xi$, $\coth\xi$, or $\cot\xi$. The integration constant c gets absorbed in Δ, and the constant a (or b) is an extra parameter in the nonlinear algebraic system for the a_{ij}. For single PDEs, Yao and Li (2002a,b) consider solutions of the form

$$u(x, t) = U(\xi) = \sum_{j=0}^{M} a_j w(\xi)^j + \sum_{j=0}^{M} b_j z(\xi) w(\xi)^{j-1}, \tag{140}$$

where $w(\xi)$ and $z(\xi)$ satisfy the Riccati equations

$$w'(\xi) = -w(\xi)z(\xi), \qquad z'(\xi) = 1 - z^2(\xi). \tag{141}$$

Since $w(\xi) = \mathrm{sech}(\xi)$, $z(\xi) = \tanh(\xi)$ this approach is similar to the sech–tanh method given in Section 4.

Generalizing further, Fan (2002b, 2003a,b,c), Fan and Hon (2002, 2003a) and Hon and Fan (2004b) take

$$y'(\xi) = \sqrt{b_0 + b_1 y + b_2 y^2 + b_3 y^3 + b_4 y^4}, \qquad b_i \text{ constant}, \tag{142}$$

which covers the functions sech, sec, tanh, tan, cn, sn, and $\mathcal{P}$. The parameters b_i are added to the nonlinear algebraic system, which makes such systems hard to solve without human intervention. Most often, such complicated nonlinear algebraic systems are solved interactively with the aid of *Mathematica* or *Maple*. To avoid unmanageable systems, $M_i(\leq 2)$ is often fixed in (137). Chen and Zhang (2003a, submitted for publication), Fan and Dai (2003) and Sirendaoreji (2003, 2004) use variants of (142) to compute polynomial and rational solutions in terms of tanh, sech, tan, Jacobi's elliptic functions, etc.

Zheng et al. (2002) introduce a clever method to compute mixed tanh–sech solutions for the combined KdV–Burgers equations. They seek formal solutions,

$$u(x,t) = U(\xi) = a_0 + \sum_{j=1}^{M} b_j \sin^j w(\xi) + \sum_{j=1}^{M} a_j \cos w(\xi) \sin w(\xi)^{j-1}, \tag{143}$$

subject to $\mathrm{d}w/\mathrm{d}\xi = \sin w(\xi)$ which, upon integration, gives $\sin w(\xi) = \mathrm{sech}(\xi)$ and $\cos w(\xi) = \pm\tanh(\xi)$. Alternatively, one can use $\mathrm{d}w/\mathrm{d}\xi = \cos w(\xi)$, which leads to $\cos w(\xi) = -\mathrm{sech}(\xi)$ and $\sin w(\xi) = \pm\tanh(\xi)$.

Liu and Li (2002a) seek solutions of the forms

$$\begin{aligned} U(\xi) &= \sum_{j=0}^{M} a_j \mathrm{sn}(\xi)^j, \qquad U(\xi) = \sum_{j=0}^{M} a_j \mathrm{sn}(\xi)^j + \sum_{j=0}^{M} b_j \mathrm{cn}(\xi)\mathrm{sn}(\xi)^{j-1}, \\ U(\xi) &= \sum_{j=0}^{M} a_j \mathrm{sn}(\xi)^j + \sum_{j=0}^{M} A_j \mathrm{cn}(\xi)\mathrm{sn}(\xi)^{j-1} + \sum_{j=0}^{M} b_j \mathrm{dn}(\xi)\mathrm{sn}(\xi)^{j-1} \\ &\quad + \sum_{j=0}^{M} B_j \mathrm{cn}(\xi)\mathrm{dn}(\xi)\mathrm{sn}(\xi)^{j-2}, \end{aligned} \tag{144}$$

which generalize the Jacobi elliptic function method in Section 4.

With respect to the second option, Gao and Tian (2001) consider

$$\begin{aligned} u_i(x,t) &= \sum_{j=0}^{M_i} a_{ij}(x,t) \tanh^j \Psi(x,t) \\ &\quad + \sum_{j=0}^{N_i} b_{ij}(x,t) \mathrm{sech}\, \Psi(x,t) \tanh^j \Psi(x,t), \end{aligned} \tag{145}$$

where $\Psi(x,t)$ is not necessarily linear in x and/or t. Of course, (145) arises from recasting the terms in (39) in a slightly different way than (40). Restricted to travelling waves, $\Psi(x,t) = c_1 x + c_2 t + \Delta$, both forms are equivalent.

Our algorithms could be generalized in many ways. With considerable effort, solutions involving complex exponentials multiplied by tanh or sech functions could be attempted.

A solution to the nonlinear Schrödinger equation is of this form. Fan and Hon (2003b), Hon and Fan (2004a) and Fan (2003b,c) give examples of complex as well as transcendental equations solved with the tanh-method.

8.2. Review of symbolic algorithms and software

There is a variety of methods to find solitary wave solutions and soliton solutions of special nonlinear PDEs. See e.g. Hereman and Takaoka (1990), Estévez and Gordoa (1995, 1998) and Helal (2002). Some of these methods are straightforward to implement in computer algebra systems (CAS).

The most comprehensive methods of finding exact solutions for ODEs and PDEs are based on similarity reductions via Lie point symmetry methods. These methods are hard to fully automate (for publications and software see e.g. Cantwell, 2002, Hereman, 1996 and Hydon, 2000). Most CAS have tools to solve a subset of linear and nonlinear PDEs. For example, *Mathematica's* `DSolve` can find general solutions for linear and weakly nonlinear PDEs. Available within *MuPAD*, the code `pdesolve` uses the method of characteristics to solve quasi-linear first-order PDEs. *Maple* offers the packages ODEtools (for solving ODEs using classification, integrating factor and symmetry methods) and PDEtools, which contains the function *pdesolve* to find exact solutions of some classes of PDEs. For information consult Cheb-Terrab and von Bülow (1995) and Cheb-Terrab (2001).

The methods presented in this paper are different from these efforts. Our algorithms and software only compute specific solutions of nonlinear PDEs which model travelling waves in terms of the tanh, sech, sn and cn functions. Our code can handle systems of ODEs and PDEs with undetermined parameters.

To our knowledge, only four software packages are similar to ours. The first package is ATFM by Parkes and Duffy (1996), who automated to some degree the tanh-method using *Mathematica*. In contrast to ATFM, our software performs the computations automatically from start to finish without human intervention. In our code, the number of independent variables x_i is not limited to one space variable x and time t; our code handles any number of dependent variables.

The second package is RATH by Li and Liu (2002), which automates the tanh-method. In contrast to our code, RATH only works for single PDEs. Extensions to cover systems of PDEs and sech solutions are under development. Surpassing our code, RATH can solve PDEs with an unspecified degree of nonlinearity and deal with negative and fractional exponents.

Table 3 compares the performance of `PDESpecialSolutions.m` and RATH. The solution times are comparable, yet occasionally there is a mismatch in the number of solutions computed. This is due in part to the representation of solutions. Occasionally special solutions are generated although—after inspection by hand—they are included in more general solutions.

Liu and Li (2002a) present the *Maple* code AJFM to automate the Jacobi elliptic function method for single PDEs. This package seeks solutions of the form (144).

The codes RATH and AJFM use the Ritt–Wu characteristic sets method, implemented by Wang (2001a,b). The CharSets package, available in Maple (Wang, 2002), is more versatile and powerful than our algorithm in Section 6.2.

Table 3
Comparison between codes PDESpecialSolutions.m and RATH. Test runs performed on a Dell Dimension 8200 PC, with 2.40 GHz Pentium 4 processor, 512 MB of RAM, with Mathematica v. 4.1 and Maple v. 7.0. The first 8 equations appear in Li and Liu (2002); the last 10 equations are listed in this paper

	PDESpecialSolutions.m		RATH		
Name of equation	CPU time (s)	# Sols.	CPU time (s)	# Sols.	Ref.
KdV–Burgers	0.125	2	0.328	1	(2.3)
KdV–Burgers–Kuramoto	0.390	8	25.641	7	(4.1)
7th-order dispersive	–	0	6.265	2	(4.7)
5th-order mKdV (Ito)	0.438	4	1.000	4	(4.11)
7th-order mKdV (Ito)	10.469	4	5.531	4	(4.13)
Generalized Fisher	0.406	4	0.469	2	(5.1)
Nonlinear heat conduction	–	0	0.485	2	(5.3)
Gen. combined KdV–mKdV	–	0	2.062	2	(5.5)
Boussinesq	0.218	1	0.142	1	(4)
KdV	0.125	1	0.126	1	(48)
KdV–Zakharov–Kuznetsov	0.469	1	0.142	1	(78)
mKdV–Zakharov–Kuznetsov	0.282	2	0.642	4	(81)
3D-mKdV	0.078	2	1.874	2	(87)
Gen. Kuramoto–Sivashinsky	0.734	16	1.453	8	(89)
Fisher	0.234	8	0.343	4	(99)
FitzHugh–Nagumo	0.719	12	–	0	(101)
Combined KdV–mKdV	0.204	2	0.251	2	(109)
Duffing	0.094	4	–	0	(114)

Finally, Abbott et al. (2002) designed a *Mathematica* notebook with key functions for the computation of polynomial solutions in sn and cn.

There are several symbolic tools for reducing and solving parameterized nonlinear algebraic systems. Some are part of codes to simplify overdetermined ODE and PDE systems. For example, the Maple package `Rif` by Wittkopf and Reid (2003) allows for the computation of solution branches of nonlinear algebraic systems. The most powerful algebraic solvers use some flavor of the Gröbner basis algorithm. For up-to-date information on developments in this area we refer to Grabmeier et al. (2003).

9. Discussion and conclusions

We presented several straightforward algorithms to compute special solutions of nonlinear PDEs, without using explicit integration. We designed the symbolic package `PDESpecialSolutions.m` to find solitary wave solutions of nonlinear PDEs involving tanh, sech, cn and sn functions.

While the software reproduces the known (and also a few presumably new) solutions for many equations, there is no guarantee that the code will compute the complete solution set of all polynomial solutions involving the tanh and/or sech functions, especially when the PDEs have parameters. This is due to restrictions on the form of the solutions and the limitations of the algebraic solver.

There is so much freedom in mixed tanh–sech solutions that the current code is limited to quadratic solutions.

Furthermore, the nonlinear constraints which arise in solving the nonlinear algebraic system may be quintic or of higher degree, and therefore unsolvable in analytic form. Also, since our software package is fully automated, it may not return the solutions in the simplest form.

The example in Section 7.8 illustrates this situation. By not solving quadratic or cubic equations explicitly the solutions (computed interactively with *Mathematica*) can be presented in a more compact and readable form.

In an attempt to avoid the explicit use of *Mathematica*'s `Solve` and `Reduce` functions, we considered various alternatives. For example, we used (i) variants of Gröbner bases on the complete system, and (ii) combinatorics on the coefficients in the polynomial solutions (setting $a_{ij} = 0$ or $a_{ij} \neq 0$, for the admissible i and j). None of these alternatives paid off for systems with parameters.

Often, the nonlinear solver returns constraints on the wave parameters c_j and the external parameters. In principle, one should verify whether or not such constraints affect the results of the previous steps in the algorithm. In particular, one should verify the consistency with the results from step 2 of the algorithms. We have not yet implemented this type of sophistication.

Acknowledgements

This material is based upon work supported by the National Science Foundation (NSF) under Grants Nos. DMS-9732069, DMS-9912293 and CCR-9901929. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.

WH thankfully acknowledges the hospitality and support of the Department of Applied Mathematics of the University of Stellenbosch, South Africa, during his sabbatical visit in Spring 2001. Part of the work was done at Wolfram Research, Inc., while WH was supported by a Visiting Scholar Grant in Fall 2000.

M. Hickman is thanked for his help with *Maple*. E. Parkes and B. Duffy are thanked for sharing their code ATFM. P. Abbott is thanked for sharing his *Mathematica* notebooks and testing earlier versions of our code. Z.-B. Li and Y.-P. Liu are thanked for providing us with the codes RATH and AJFM.

The authors are grateful to B. Deconinck, B. Herbst, P.G.L. Leach, W. Malfliet, J. Sanders, and F. Verheest for valuable comments. Last but not least, students S. Nicodemus, P. Blanchard, J. Blevins, S. Formaneck, J. Heath, B. Kowalski, A. Menz, J. Milwid, and M. Porter-Peden are thanked for their help with the project.

Appendix. Using the software package

We illustrate the use of the package `PDESpecialSolutions.m` on a PC. Users should have access to *Mathematica* v. 3.0 or higher.

Put the package in a directory, say myDirectory, on drive C. Start a *Mathematica* notebook session and execute the commands:

```
In[1] = SetDirectory["c:\\myDirectory"];  (* specify directory *)

In[2] = <<PDESpecialSolutions.m           (* read in package   *)

In[3] = PDESpecialSolutions[
        {D[u[x,t],t]-alpha*(6*u[x,t]*D[u[x,t],x]+D[u[x,t],{x,3}])+
         2*beta*v[x,t]*D[v[x,t],x] == 0,
         D[v[x,t],t]+3*u[x,t]*D[v[x,t],x]+D[v[x,t],{x,3}] == 0},
         {u[x,t],v[x,t]}, {x,t}, {alpha, beta}, Form -> Sech,
         Verbose -> True, InputForm -> False, NumericTest -> True,
         SymbolicTest -> True, SolveAlgebraicSystem -> True
         (*, DegreeOfThePolynomial -> {m[1] -> 2, m[2] -> 1} *)];
```

The package will compute the sech solutions (37) and (38) of the coupled KdV equation (22).

If the `DegreeOfThePolynomial` $\rightarrow \{m[1] \rightarrow 2, m[2] \rightarrow 1\}$ were specified, the code would continue with this case only and compute (37).

If `SolveAlgebraicSystem` $\rightarrow$ `False`, the algebraic system will be generated but not automatically solved.

The format of `PDESpecialSolutions` is similar to the *Mathematica* function DSolve. The output is a list of -lists with solutions and constraints. The Backus–Naur form of the function is

⟨*Main Function*⟩	→	`PDESpecialSolutions[`⟨*Equations*⟩, ⟨*Functions*⟩, ⟨*Variables*⟩, ⟨*Parameters*⟩, ⟨*Options*⟩`]`
⟨*Options*⟩	→	`Form`→ ⟨*Form*⟩ | `Verbose`→ ⟨*Bool*⟩ | `InputForm`→ ⟨*Bool*⟩ | `DegreeOfThePolynomial` → ⟨*List of Rules*⟩ | `SolveAlgebraicSystem`→ ⟨*Bool*⟩ | `NumericTest` → ⟨*Bool*⟩ | `SymbolicTest` → ⟨*Bool*⟩
⟨*Form*⟩	→	`Tanh` | `Sech` | `SechTanh` | `JacobiCN` | `JacobiSN`
⟨*Bool*⟩	→	`True` | `False`
⟨*List of Rules*⟩	→	`{m[1] →Integer,m[2] → Integer,...}`

The default value of `Form` is `Tanh`. The package `PDESpecialSolutions.m` has been tested on both UNIX work stations and PCs with *Mathematica* versions 3.0, 4.0 and 4.1. A test set of over 50 PDEs and half a dozen ODEs was used.

References

Abbott, P.C., Parkes, E.J., Duffy, B.R., 2002. The Jacobi elliptic-function method for finding periodic-wave solutions to nonlinear evolution equations. Notebook with *Mathematica* implementation of the Jacobi elliptic function method. Available at http://physics.uwa.edu.au/pub/Mathematica/Solitons/.

Ablowitz, M.J., Clarkson, P.A., 1991. Solitons, Nonlinear Evolution Equations and Inverse Scattering. Cambridge University Press, Cambridge, UK.

Baldwin, D., Göktaş, Ü, Hereman, W., Hong, L., Martino, R.S., Miller, J.C., 2001. PDESpecialSolutions.m: A Mathematica program for the symbolic computation of exact solutions expressible in hyperbolic and elliptic functions for nonlinear PDEs. Available at http://www.mines.edu/fs_home/whereman/.

Cantwell, B.J., 2002. Introduction to Symmetry Analysis. Cambridge Texts in Applied Mathematics. Cambridge University Press, Cambridge, UK.

Cheb-Terrab, E.S., 2001. PDETools Project. Center for Experimental and Constructive Mathematics. Simon Fraser University, Vancouver, British Columbia, Canada. Available at http://lie.uwaterloo.ca/pdetools.m.

Cheb-Terrab, E.S., von Bülow, K., 1995. A computational approach for the analytical solving of systems of partial differential equations. Comput. Phys. Comm. 90, 102–116.

Chen, H.T., Zhang, H.Q., 2003a. Improved Jacobin elliptic function method and its applications. Chaos Solitons Fractals 15, 585–591.

Chen, H.T., Zhang, H.Q., 2003b. An algebraic method for finding soliton and periodic solutions of nonlinear evolution equations. Comput. Phys. Comm. (submitted for publication) (preprint).

Das, G.C., Sarma, J., 1999. Response to Comment on A new mathematical approach for finding the solitary waves in dusty plasmas. Phys. Plasmas 6, 4393–4397.

Das, K.P., Verheest, F., 1989. Ion-acoustic solitons in magnetized multi-component plasmas including negative ions. J. Plasma Phys. 41, 139–155.

Drazin, P.G., Johnson, R.S., 1989. Solitons: An Introduction. Cambridge Texts in Appl. Maths. Cambridge University Press, Cambridge, UK.

Estévez, P.G., Gordoa, P.R., 1995. Non-classical symmetries and the singular manifold method: theory and six examples. Stud. Appl. Math. 95, 73–113.

Estévez, P.G., Gordoa, P.R., 1998. Non-classical symmetries and the singular manifold method: a further two examples. J. Phys. A 31, 7511–7519.

Fan, E., 2002a. Travelling wave solutions for nonlinear evolution equations using symbolic computation. Comput. Math. Appl. 43, 671–680.

Fan, E., 2002b. Multiple travelling wave solutions of nonlinear evolution equations using a unified algebraic method. J. Phys. A 35, 6853–6872.

Fan, E., 2002c. A new algebraic method for finding series of travelling wave solutions to a coupled Ito system. J. Phys. Soc. Japan 71, 2663–2668.

Fan, E., 2003a. A new algebraic method for finding the line soliton solutions and doubly periodic wave solution to a two-dimensional perturbed KdV equation. Chaos Solitons Fractals 15, 567–574.

Fan, E., 2003b. An algebraic method for finding a series of exact solutions to integrable and nonintegrable nonlinear evolution equations. J. Phys. A 36, 7009–7026.

Fan, E., 2003c. Uniformly constructing a series of explicit exact solutions to nonlinear equations in mathematical physics. Chaos Solitons Fractals 16, 819–839.

Fan, E., Dai, H.H., 2003. A direct approach with computerized symbolic computation for finding a series of traveling waves to nonlinear equations. Comput. Phys. Comm. 153, 17–30.

Fan, E., Hon, Y.C., 2002. An algebraic method with computerized symbolic computation and new families of travelling wave solutions for the Hirota–Satsuma coupled KdV equation. Nuovo Cimento B 117, 659–669.

Fan, E., Hon, Y.C., 2003a. A series of travelling wave solutions for two variant Boussinesq equations in shallow water waves. Chaos Solitons Fractals 15, 559–566.

Fan, E., Hon, Y.C., 2003b. Applications of extended tanh method to 'special' types of nonlinear equations. Appl. Math. Comput. 141, 351–358.

Fan, E., Zhang, H., 2002. Applications of the Jacobi elliptic function method to special-type nonlinear equations. Phys. Lett. A 305, 383–392.

Fu, Z., Liu, S., Liu, S., Zhao, Q., 2001. New Jacobi elliptic function expansion and new periodic solutions of nonlinear wave equations. Phys. Lett. A 290, 72–76.

Gao, Y.T., Tian, B., 2001. Generalized hyperbolic-function method with computerized symbolic computation to construct the solitonic solutions to nonlinear equations of mathematical physics. Comput. Phys. Comm. 133, 158–164.

Göktaş, Ü, Hereman, W., 1997. Symbolic computation of conserved densities for systems of nonlinear evolution equations. J. Symbolic Comput. 24, 591–621.

Grabmeier, J., Kaltofen, E., Weispfenning, V. (Eds.), 2003. Computer Algebra Handbook: Foundations, Applications, Systems. Springer Verlag, Berlin.

Guha-Roy, C., 1987. Solitary wave solutions of a system of coupled nonlinear equations. J. Math. Phys. 28, 2087–2088.

Helal, M.A., 2002. Soliton solution of some nonlinear partial differential equations and its applications in fluid mechanics. Chaos Solitons Fractals 13, 1917–1929.

Hereman, W., 1990. Application of a Macsyma program for the Painlevé test to the Fitzhugh–Nagumo equation. In: Conte, R., Boccara, N. (Eds.), Proc. Winter School Theoretical Physics, Partially Integrable Evolution Equations in Physics. Kluwer Academic Publishers, Dordrecht, pp. 585–586.

Hereman, W., 1991. Exact solitary wave solutions of coupled nonlinear evolution equations using MACSYMA. Comput. Phys. Comm. 65, 143–150.

Hereman, W., 1996. Symbolic software for lie symmetry analysis. In: Ibragimov, N.H. (Ed.), CRC Handbook of Lie Group Analysis of Differential Equations. New Trends in Theoretical Developments and Computational Methods, vol. 3. CRC Press, Boca Raton, FL, pp. 367–413. (Chapter 13).

Hereman, W., Takaoka, M., 1990. Solitary wave solutions of nonlinear evolution and wave equations using Macsyma. J. Phys. A 23, 4805–4822.

Hon, Y.C., Fan, E., 2004a. A series of new exact solutions for a complex coupled KdV system. Chaos Solitons Fractals 19, 515–525.

Hon, Y.C., Fan, E., 2004b. Solitary wave and doubly periodic wave solutions for the Kersten–Krasil' shchik coupled KdV–mKdV system. Chaos Solitons Fractals 19, 1141–1146.

Hydon, P.E., 2000. Symmetry Methods for Differential Equations. Cambridge Texts in Applied Mathematics. Cambridge University Press, Cambridge, UK.

Kichenassamy, S., Olver, P., 1993. Existence and non-existence of solitary wave solutions to higher order model evolution equations. SIAM J. Math. Anal. 23, 1141–1166.

Lawden, D.F., 1989. Elliptic Functions and Applications. Appl. Math. Sciences, vol. 80. Springer Verlag, New York.

Li, Z.-B., Liu, Y.-P., 2002. RATH: A Maple package for finding travelling solitary wave solutions to nonlinear evolution equations. Comput. Phys. Comm. 148, 256–266.

Liu, Y.-P., Li, Z.-B., 2002a. An automated Jacobi elliptic function method for finding periodic wave solutions to nonlinear evolution equations. Chinese Phys. Lett. 19, 1228–1230.

Liu, Y.-P., Li, Z.-B., 2002b. The link between the hyperbolic tangent function method and Jacobi elliptic function method. Phys. Lett. A (submitted for publication) (preprint).

Malfliet, W., 1992. Solitary wave solutions of nonlinear wave equations. Amer. J. Phys. 60, 650–654.

Malfliet, W., 2004. The tanh method: a tool for solving certain classes of nonlinear evolution and wave equations. J. Comp. Appl. Math. 164–165, 529–541.

Malfliet, W., Hereman, W., 1996. The tanh method: I. Exact solutions of nonlinear evolution and wave equations. Phys. Scripta 54, 563–568.

Parkes, E.J., Duffy, B.R., 1996. An automated tanh-function method for finding solitary wave solutions to non-linear evolution equations. Comput. Phys. Comm. 98, 288–300.

Parkes, E.J., Duffy, B.R., Abbott, P.C., 2002. The Jacobi elliptic-function method for finding periodic-wave solutions to nonlinear evolution equations. Phys. Lett. A 295, 280–286.

Parkes, E.J, Zhu, Z., Duffy, B.R., Huang, H.C., 1998. Sech-polynomial travelling solitary-wave solutions of odd-order generalized KdV equations. Phys. Lett. A 248, 219–224.

Sirendaoreji, S.J., 2003. Auxiliary equation method for solving nonlinear partial differential equations. Phys. Lett. A 309, 387–396.

Sirendaoreji, S.J., 2004. New exact travelling wave solutions for the Kawahara and modified Kawahara equations. Chaos Solitons Fractals 19, 147–150.

Wang, D., 2001a. Elimination Methods. Springer Verlag, New York.

Wang, D., 2001b. A generalized algorithm for computing characteristic sets. In: Shirayanagi, K., Yokoyama, K. (Eds.), ASCM 2001, Proc. 5th Asian Symp. Computer Mathematics. World Scientific, Singapore, pp. 165–174.

Wang, D., 2002. CharSets 2.0—A package for decomposing (differential) polynomial systems. Available at http://www.calfor.lip6.fr/~wang/.

Wittkopf, A., Reid, G., 2003. Introduction to the Rif Package Version 1.1. Available at http://www.cecm.sfu.ca/~wittkopf/Rif.html.

Yan, Z., 2003. The extended Jacobian elliptic function expansion method and its applications in the generalized Hirota–Satsuma coupled KdV system. Chaos Solitons Fractals 15, 575–583.

Yang, Z.J., 1994. Traveling-wave solutions to nonlinear evolution and wave equations. J. Phys. A 27, 2837–2856.

Yao, R.-X., Li, Z.-B., 2002a. New exact solutions for three nonlinear evolution equations. Phys. Lett. A 297, 196–204.

Yao, R.-X., Li, Z.-B., 2002b. New solitary wave solutions for nonlinear evolution equations. Chinese Phys. Lett. 11, 864–867.

Zheng, X.-D., Xia, T.-C., Zhang, H.-Q., 2002. New exact traveling wave solutions for compound KdV–Burgers equations in mathematical physics. Appl. Maths. E-Notes 2, 45–50.