ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022

Efficient Combination of Improved Fourier Periodic Expansion Function with Least Square Weight Method for Solving ODEs with Optimal Solution

Gudur Hemalatha [1], Gandham Vijay kumar [2]

Assistant Professor [1, 2]

Sree Dattha Engineering and Science Sheriguda, Ibarahimpatnam.

Abstract

Solving ordinary differential equations (ODEs) with traditional techniques is complicated and time taking task. Traditional techniques of ODE solvers are providing approximate solution. To overcome that drawback and to get exact solution many researchers implemented new techniques which reduces time complexity and gives better solution for given ODE with optimization. In this proposed technique we used generic method of solving ODEs using adaptive differential evolution. Obtained approximate solution is given to improved Fourier periodic expansion function with least square weight method for solving ODEs with optimal solution.

Keywords: Ordinary Differential Equation (ODE), generic method, optimization, Least Square Weight method, Fourier Periodic Expansion Function.

I. Introduction

Ordinary Differential Equations

The Ordinary Differential Equation (ODE) solvers in MATLAB® solve initial value problems with a variety of properties. The solvers can work on stiff or nonstiff problems, problems with a mass matrix, differential algebraic equations (DAEs), or fully implicit problems.

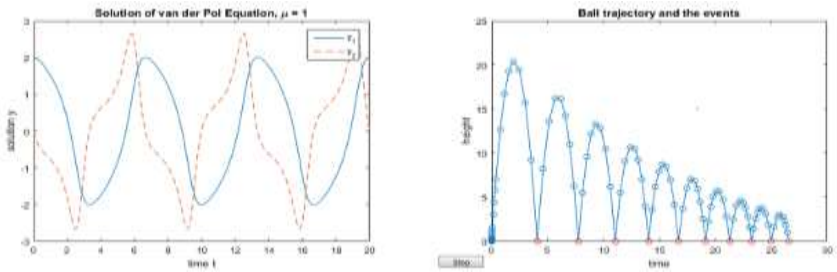

Fig1. Examples for which ODEs need to be solved

Choosing an ODE Solver

ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022

An ordinary differential equation (ODE) contains one or more derivatives of a dependent variable, y, with respect to a single independent variable, x, usually referred to as time. The notation used here for representing derivatives of y with respect to x is y' for a first derivative, y'' for a second derivative, and so on. The order of the ODE is equal to the highest-order derivative of y that appears in the equation.

For example, this is a second order ODE:

$$y'' = 9y$$

In an initial value problem, the ODE is solved by starting from an initial state. Using the initial condition, y_0, as well as a period of time over which the answer is to be obtained, (t_0, t_f), the solution is obtained iteratively. At each step the solver applies a particular algorithm to the results of previous steps. At the first such step, the initial condition provides the necessary information that allows the integration to proceed. The final result is that the ODE solver returns a vector of time steps $t = [t_0, t_1, t_2, ..., t_f]$ as well as the corresponding solution at each step $y = [y_0, y_1, y_2, ..., y_f]$.

Types of ODEs

The ODE solvers in MATLAB® solve these types of first-order ODEs:

- Explicit ODEs of the form $y' = f(x, y)$.
- Linearly implicit ODEs of the form $M(x, y)y' = f(x, y)$, where $M(x, y)$ is a nonsingular mass matrix. The mass matrix can be time- or state-dependent, or it can be a constant matrix. Linearly implicit ODEs involve linear combinations of the first derivative of y, which are encoded in the mass matrix.

  Linearly implicit ODEs can always be transformed to an explicit form, $y' = M^{-1}(x, y) f(x, y)$. However, specifying the mass matrix directly to the ODE solver avoids this transformation, which is inconvenient and can be computationally expensive.
- If some components of y' are missing, then the equations are called differential algebraic equations, or DAEs, and the system of DAEs contains some algebraic variables. Algebraic variables are dependent variables whose derivatives do not appear in the equations. A system of DAEs can be rewritten as an equivalent system of first-order ODEs by taking derivatives of the equations to eliminate the algebraic variables. The number of derivatives needed to rewrite a DAE as an ODE is called the differential index. The `ode15s` and `ode23t` solvers can solve index-1 DAEs.
- Fully implicit ODEs of the form $f(x, y, y') = 0$. Fully implicit ODEs cannot be rewritten in an explicit form, and might also contain some algebraic variables. The `ode15i` solver is designed for fully implicit problems, including index-1 DAEs.

You can supply additional information to the solver for some types of problems by using the `odeset` function to create an options structure.

Basic Solver Selection for ODEs in MATLAB

ODE45 performs well with most ODE problems and should generally be your first choice of solver. However, ODE23, ODE78, ODE89 and ODE113 can be more efficient than ode45 for problems with looser or tighter accuracy requirements. Some ODE problems exhibit stiffness, or difficulty in evaluation. Stiffness is a term that defies a precise definition, but in general, stiffness occurs when there is a difference in scaling somewhere in the problem.

For example, if an ODE has two solution components that vary on drastically different time scales, then the equation might be stiff. You can identify a problem as stiff if nonstiff solvers (such as ode45) are unable to solve the problem or are extremely slow. If you observe that a nonstiff solver is very slow, try using a stiff solver such as ODE15s instead. When using a stiff solver, you can improve reliability and efficiency by supplying the Jacobian matrix or its sparsity pattern.

Below table provides general guidelines on when to use each of the different solvers.

ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022

ode15s	Stiff	Low to Medium	Try ode15s when ode45 fails or is inefficient and you suspect that the problem is stiff. Also use ode15s when solving differential algebraic equations (DAEs).
ode23s		Low	ode23s can be more efficient than ode15s at problems with crude error tolerances. It can solve some stiff problems for which ode15s is not effective. ode23s computes the Jacobian in each step, so it is beneficial to provide the Jacobian via odeset to maximize efficiency and accuracy. If there is a mass matrix, it must be constant.
ode23t		Low	Use ode23t if the problem is only moderately stiff and you need a solution without numerical damping. ode23t can solve differential algebraic equations (DAEs).
ode23tb		Low	Like ode23s, the ode23tb solver might be more efficient than ode15s at problems with crude error tolerances.
ode15i	Fully implicit	Low	Use ode15i for fully implicit problems $f(t,y,y') = 0$ and for differential algebraic equations (DAEs) of index 1.

Solver	Problem Type	Accuracy	When to Use
ode45	Nonstiff	Medium	Most of the time, ode45 should be the first solver you try.
ode23		Low	ode23 can be more efficient than ode45 at problems with crude tolerances, or in the presence of moderate stiffness.
ode113		Low to High	ode113 can be more efficient than ode45 at problems with stringent error tolerances, or when the ODE function is expensive to evaluate.
ode78		High	ode78 can be more efficient than ode45 at problems with smooth solutions that have high accuracy requirements.
ode89		High	ode89 can be more efficient than ode78 on very smooth problems, when integrating over long time intervals, or when tolerances are especially tight.

This table contains a list of the available ODE and DAE example files as well as the solvers and options they use in MATLAB.

II. Literature Survey for ODEs Solving

Differential equations can describe nearly all systems undergoing change. They are ubiquitous is science and engineering as well as economics, social science, biology, business, health care, etc. Many researchers and mathematicians have studied the nature of Differential Equations and many complicated systems that can be described quite precisely with mathematical expressions.

There are basic 2 types of differential equations as ,

Ordinary differential equations		Partial differential equations
Class 1	Class 2	Class 3
$\frac{dy}{dx} = f(x)$	$\frac{dy}{dx} = f(x,y)$	$x\frac{\partial u}{\partial x} + y\frac{\partial u}{\partial y} = u$

Swiss mathematicians, brothers Jacob Bernoulli (1654-1705) and Johann Bernoulli (1667-1748), in Basel, Switzerland, were among the first interpreters of Leibniz' version of differential calculus. They were both critical of Newton's theories and maintained that Newton's theory of

ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022

fluxions was plagiarized from Leibniz' original theories, and went to great lengths, using differential calculus, to disprove Newton's Principia, on account that the brothers could not accept the theory, which Newton had proven, that the earth and the planets rotate around the sun in elliptical orbits. [3]

The first book on the subject of differential equations, supposedly, was Italian mathematician Gabriele Manfredi's 1707 On the Construction of First-degree Differential Equations, written between 1701 and 1704, published in Latin. [4] The book was largely based or themed on the views of the Leibniz and the Bernoulli brothers. Most of the publications on differential equations and partial differential equations, in the years to follow, in the 18th century, seemed to expand on the version developed by Leibniz, a methodology, employed by those as Leonhard Euler, Daniel Bernoulli, Joseph Lagrange, and Pierre Laplace.

In 1739, Swiss mathematician Leonhard Euler began using the integrating factor as an aid to derive differential equations that were integrable in finite form [5] The circa 1828 work of English physical mathematician George Green seems to have something to do with defining a test for an "integrable" or conservative field of force (or somehow has connection to thermodynamics via William Thomson); such as in terms of the later 1871 restylized "curl" notation (test of integrability) of James Maxwell (or possibly the earlier work of Peter Tait). [10]

III. Proposed Methodology

Generic Method:

1.1 Improved Fourier Periodic Expansion Function

This immediately gives any coefficient a_k of the trigonometrical series for $\varphi(y)$ for any function which has such an expansion. It works because if φ has such an expansion, then (under suitable convergence assumptions) the integral

$$a_k = \int_{-1}^{1} \varphi(y)\cos(2k+1)\frac{\pi y}{2}\,dy$$
$$= \int_{-1}^{1} \left(a\cos\frac{\pi y}{2}\cos(2k+1)\frac{\pi y}{2} + a'\cos 3\frac{\pi y}{2}\cos(2k+1)\frac{\pi y}{2} + \cdots\right) dy$$

1.2 Least Square Weight Method

The method of ordinary least squares assumes that there is constant variance in the errors (which is called homoscedasticity). The method of weighted least squares can be used when the ordinary least squares assumption of constant variance in the errors is violated (which is called heteroscedasticity).

Advantages of proposed Technique

ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022

Nearly exact solution

Less time Complexity

Applications of Proposed Technique

- science and engineering
- economics,
- social science,
- biology,
- business,
- health care, etc

IV. Result Analysis

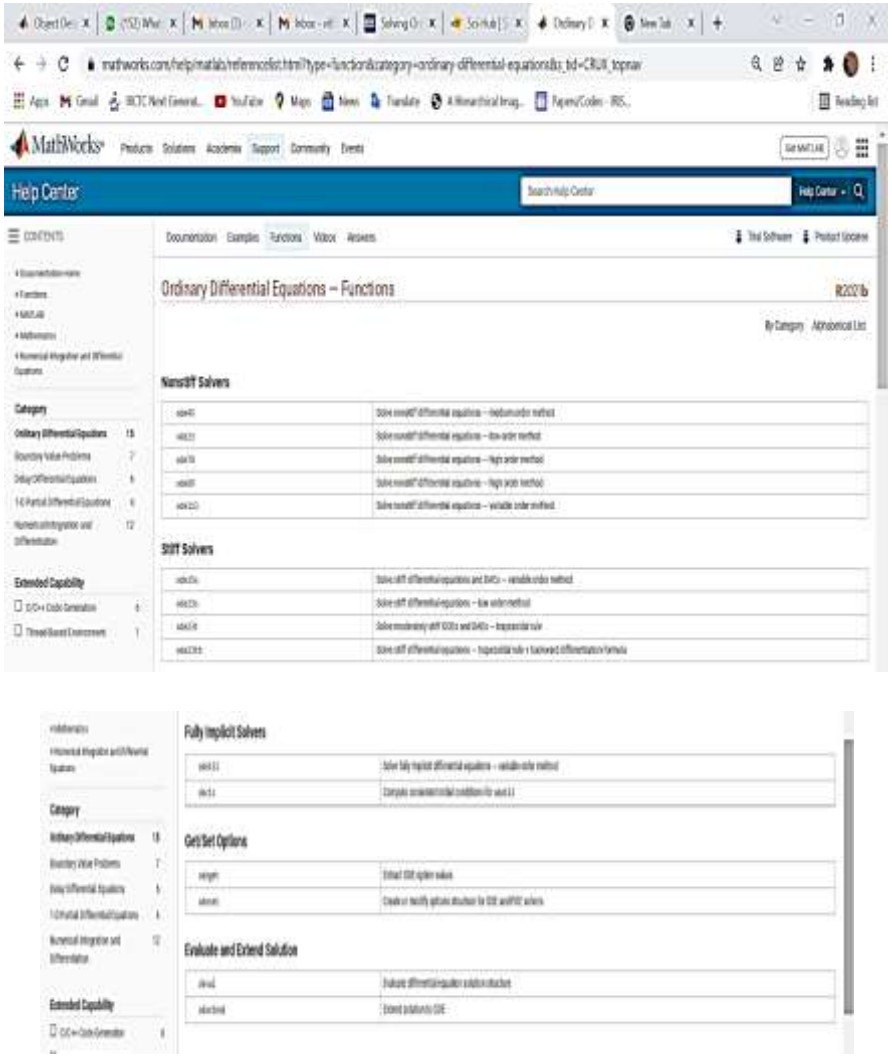

Fig 4.1 . Functions available in matlab for Solving Ordinary Differential Equation

For solving differential equation matlab needs 3 steps as,

ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022

Step1: initialising condition for Matlab ODE solver

Step2: Solving system equation using Matlab tools

Step3: Getting optimal solution and plotting

Example :

Output

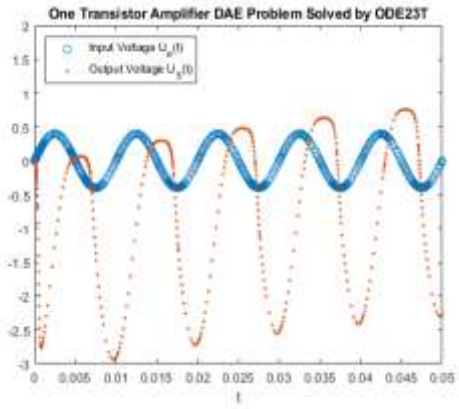

Fig. 4.2 Output of ODE23T DAE solution

V. Conclusion

With the help of proposed technique ODEs are solved with optimal solution. In proposed technique we used combination of two techniques as improved Fourier periodic expansion function and least square weight method.

REFERENCES

[1] I. M. Bomze, ‘‘Lotka-Volterra equation and replicator dynamics: New issues in classification,’’ Biol. Cybern., vol. 72, no. 5, pp. 447–453, Apr. 1995.

ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022

[2] J. Nam, M. Behr, and M. Pasquali, ‘‘Space-time least-squares finite element method for convection-reaction system with transformed variables,’’ Comput. Methods Appl. Mech. Eng., vol. 200, nos. 33–36, pp. 2562–2576, Aug. 2011.

[3] N. H. Sweilam, M. M. Khader, and A. M. Nagy, ‘‘Numerical solution of two-sided space-fractional wave equation using finite difference method,’’ J. Comput. Appl. Math., vol. 235, no. 8, pp. 2832–2841, Feb. 2011.

[4] S. B. Coşkun and M. T. Atay, ‘‘Fin efficiency analysis of convective straight fins with temperature dependent thermal conductivity using variational iteration method,’’ Appl. Thermal Eng., vol. 28, nos. 17–18, pp. 2345–2352, Dec. 2008.

[5] D. N. Butusov, A. I. Karimov, and A. V. Tutueva, ‘‘Symmetric extrapolation solvers for ordinary differential equations,’’ in Proc. IEEE NW Russia Young Res. Electr. Electron. Eng. Conf. (EIConRusNW), Feb. 2016, pp. 162–167.

[6] E. Oñate and S. Idelsohn, ‘‘A mesh-free finite point method for advectivediffusive transport and fluid flow problems,’’ Comput. Mech., vol. 21, nos. 4–5, pp. 283–292, May 1998.

[7] J. Park and I. W. Sandberg, ‘‘Approximation and radial-basis-function networks,’’ Neural Comput., vol. 5, no. 2, pp. 305–316, Mar. 1993.

[8] Y. Cai, X. Liu, and Z. Cai, ‘‘BS-Nets: An end-to-end framework for band selection of hyperspectral image,’’ IEEE Trans. Geosci. Remote Sens., vol. 58, no. 3, pp. 1969–1984, Mar. 2020.

[9] Y. Cai, X. Liu, Y. Zhang, and Z. Cai, ‘‘Hierarchical ensemble of extreme learning machine,’’ Pattern Recognit. Lett., vol. 116, pp. 101–106, Dec. 2018.

[10] K. M. Liew, X. Zhao, and A. J. M. Ferreira, ‘‘A review of meshless methods for laminated and functionally graded plates and shells,’’ Compos. Struct., vol. 93, no. 8, pp. 2031–2041, Jul. 2011.

[11] D. D. Ganji and A. Sadighi, ‘‘Application of homotopy-perturbation and variational iteration methods to nonlinear heat transfer and porous media equations,’’ J. Comput. Appl. Math., vol. 207, no. 1, pp. 24–34, Oct. 2007.

[12] P. Darania and A. Ebadian, ‘‘Development of the Taylor expansion approach for nonlinear integro-differential equations,’’ Int. J. Contemp. Math. Sci., vol. 1, no. 14, pp. 651–654, 2006.

[13] S. Zhang and S. Hong, ‘‘Variable separation method for a nonlinear time fractional partial differential equation with forcing term,’’ J. Comput. Appl. Math., vol. 339, pp. 297–305, Sep. 2018.

ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022

[14] J. Brest, M. S. Maucec, and B. Boskovic, ‘‘Single objective real-parameter optimization: Algorithm jSO,’’ in Proc. IEEE Congr. Evol. Comput. (CEC), Jun. 2017, pp. 1311–1318.

[15] A. W. Mohamed, A. A. Hadi, and A. K. Mohamed, ‘‘Gaining-sharing knowledge based algorithm for solving optimization problems: A novel nature-inspired algorithm,’’ Int. J. Mach. Learn. Cybern., vol. 11, no. 7, pp. 1501–1529, Jul. 2020.

[16] G. Mateescu, ‘‘On the application of genetic algorithms to differential equations,’’ J. Econ. Forecasting, vol. 3, no. 2, pp. 5–9, 2006.

[17] I. G. Tsoulos and I. E. Lagaris, ‘‘Solving differential equations with genetic programming,’’ Genet. Program. Evolvable Mach., vol. 7, no. 1, pp. 33–54, Mar. 2006.

[18] A. Sobester, P. B. Nair, and A. J. Keane, ‘‘Genetic programming approaches for solving elliptic partial differential equations,’’ IEEE Trans. Evol. Comput., vol. 12, no. 4, pp. 469–478, Aug. 2008.

[19] J. M. Chaquet and E. J. Carmona, ‘‘Using covariance matrix adaptation evolution strategies for solving different types of differential equations,’’ Soft Comput., vol. 23, no. 5, pp. 1643–1666, Mar. 2019.

[20] W. J. A. Lobao, D. M. Dias, and M. A. C. Pacheco, ‘‘Genetic programming and automatic differentiation algorithms applied to the solution of ordinary and partial differential equations,’’ in Proc. IEEE Congr. Evol. Comput. (CEC), Jul. 2016, pp. 5286–5292.

[21] M. Babaei, ‘‘A general approach to approximate solutions of nonlinear differential equations using particle swarm optimization,’’ Appl. Soft Comput., vol. 13, no. 7, pp. 3354–3365, Jul. 2013.

[22] A. Sadollah, H. Eskandar, D. G. Yoo, and J. H. Kim, ‘‘Approximate solving of nonlinear ordinary differential equations using least square weight function and metaheuristic algorithms,’’ Eng. Appl. Artif. Intell., vol. 40, pp. 117–132, Apr. 2015.

[23] A. Panda and S. Pani, ‘‘Determining approximate solutions of nonlinear ordinary differential equations using orthogonal colliding bodies optimization,’’ Neural Process. Lett., vol. 48, no. 1, pp. 219–243, Aug. 2018.

[24] J. M. Chaquet and E. J. Carmona, ‘‘Solving differential equations with Fourier series and evolution strategies,’’ Appl. Soft Comput., vol. 12, no. 9, pp. 3051–3062, Sep. 2012.

[25] C. A. J. Fletcher, Weighted Residual Methods. Berlin, Germany: Springer, 1988, pp. 98–162.