Available online at www.sciencedirect.com

ScienceDirect

Procedia Engineering 10 (2011) 3734–3740

Engineering Procedia

ICM11

An Approximate Solution for a Simple Pendulum beyond the Small Angles Regimes Using Hybrid Artificial Neural Network and Particle Swarm Optimization Algorithm

Alireza Yekrangi[a] , Mehdi Ghalambaz[b], Aminreza Noghrehabadi[c]*, Yaghoub Tadi Beni[d], Mohamadreza Abadyan[a]*, Molood Noghreh Abadi[e], Mehdi Noghreh Abadi[f]

[a]*Mechanical Engineering Group, Ramsar Branch, Islamic Azad University, Ramsar, Iran*
[b]*Department of Mechanical Engineering, Dezful Branch, Islamic Azad University, Dezful, Iran*
[c]*Department of Mechanical Engineering, Ahvaz Branch, Islamic Azad University, Ahvaz, Iran*
[d]*Faculty of Engineering, University of Shahrekord, Shahrekord, Iran*
[e]*Department of Computer Engineering, Dezful Branch, Islamic Azad University, Dezful, Iran*
[f]*Khouzestan Science and Research Branch, Islamic Azad University, Ahvaz, Iran*

Abstract

Simple pendulum is the most popular example in mechanics. Study on the physics of simple pendulum is a key to understanding the nonlinear dynamics of many other systems. However, there is an exact analytical solution for this problem, but its exact solution is in the form of the Jacobi elliptic integral which it is hard for using in simple engineering manipulations. Hence, determining an accurate simple approximate solution is helpful. This study presents a new method by using hybrid neural networks and particle swarm optimization algorithm, in order to find a simple approximate solution for motion of a nonlinear pendulum beyond the small angles regime. The approximate solution is simple and powerful to converge to the exact solution. The results of the approximate solution are compared with exact solution and linear solution, using tables and graphs. Furthermore, the present method is expandable to solve complex pendulums.

* Corresponding author. Tel.: +98-916-312-8841; fax: +98-611-332-9199.
E-mail address: A.R.Noghrehbadi@scu.ac.ir.
* Corresponding author. Tel.: +98-913-314-7180; fax: +98-641-526-1054.
E-mail address: Abadyan@yahoo.com.

1877-7058 © 2011 Published by Elsevier Ltd. Open access under CC BY-NC-ND license.
Selection and peer-review under responsibility of ICM11
doi:10.1016/j.proeng.2011.04.611

Keywords: Simple pendulum, Differential equation, Neural networks, Particle Swarm Optimization (PSO)

1. Introduction

Simple pendulum is the most popular example in mechanics and is studied in introductory university as classical mechanics course. In many textbooks and engineering problems, the periodic motion exhibited by a simple pendulum is harmonic only for small angle oscillations [1]. Beyond this limit, the equation of motion is nonlinear. However, there is an exact analytical solution for this problem, but its exact solution is in the form of integrals. So determining an accurate simple approximate solution is very helpful. In order to find a simple approximate solution for the nonlinear pendulum, there are several methods, such as exp function [2], harmonic balance based methods [3], parameter expansion [4], perturbation techniques [5], decomposition [6], and variational approaches [7]. Most of these techniques have been used to obtain analytical approximate solutions for the nonlinear pendulum [8]. Excellent reviews on some asymptotic methods for strongly nonlinear equation can be found in details in [9]. In general, given the nature of the nonlinear phenomenon, the approximation methods can only be applied within certain ranges of the physical parameters and to certain classes of problems. In other hand, artificial neural networks and particle swarm optimization algorithm are existing form of artificial intelligence. The artificial neural networks mimic the learning process of the human brain in order to extract patterns from historical data. Artificial neural networks are simple in shapes and very powerful in function approximation [10]. The Particle Swarm Optimization algorithm was inspired by the natural flocking and swarming behavior of birds and insects [11]. It is a very powerful gradient free optimization algorithm, for finding the global optimum parameters of a function.
The aim of present paper is to use the ability and nature of neural networks in order to determine an accurate and simple expression for the solutions of a nonlinear pendulum. The approximate solution was obtained by means of a hybrid artificial neural network and particle swarm optimization algorithm.

2. Pendulum problem

An ideal simple pendulum consists of a particle of mass *m* suspended by a massless rigid rod of length *L* that is fixed at the upper end such that the particle moves in a vertical circle. This simple mechanical system oscillates with a symmetric restoring force due to gravity, as illustrated in fig. 1. The equation of motion is given by

$$\frac{d^2\theta}{dt^2}+\frac{g}{l}\sin\theta=0 \tag{1-a}$$

where θ is the angular displacement in radians ($\theta=0$ at the rest position), *t* is the time, *l* is the length of pendulum and, *g* is the local acceleration of gravity. In the situation that particle is released from angle θ_0 with zero velocity the initial condition is as follow

$$\theta(0)=\theta_0, \qquad \frac{d\theta}{dt}(0)=0 \tag{1-b}$$

For small angle oscillation, the approximation $\sin\theta\approx\theta$ is valid and eq. (1) becomes a linear differential equation [12] analogous to the one for the simple harmonic oscillator. Linearised form of eq (1) is written as follow

$$\frac{d^2\theta}{dt^2}+\frac{g}{l}\theta=0 \tag{2-a}$$

$$\theta(0)=\theta_0, \qquad \frac{d\theta}{dt}(0)=0 \tag{2-b}$$

where the exact solution of eq (2) is

$$\theta(t)=\theta_0\cos\left(\sqrt{\frac{g}{l}}t\right) \tag{3}$$

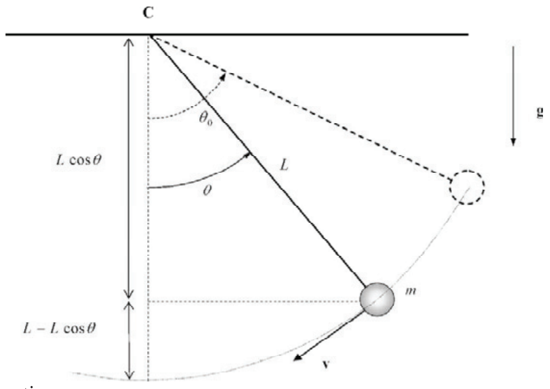


Fig. 1. The simple pendulum schematic

Beyond the small angles the eq (1) is remain in its nonlinear form. The exact solution of eq (1) is:

$$\theta(t)=2\arcsin\left\{\sin\frac{\theta_0}{2}sn\left[K\left(\sin^2\frac{\theta_0}{2}\right)-\frac{g}{l}t;\sin^2\frac{\theta_0}{2}\right]\right\} \tag{4}$$

where $K(m)=\int_0^1\frac{dz}{\sqrt{(1-z^2)(1-m\,z^2)}}$ and $sn(u;m)$ is the Jacobi elliptic function [13.].

In the following next sections, artificial neural network and particle swarm optimization algorithm will be introduced.

3. Artificial neural network

Artificial Neural Networks (ANNs), along with training and verification techniques are a very powerful method to describe relationship between independent and dependent variables, when the explicit form of mapping is not known. The common types of feed forward neural networks are Multi Layer Perceptrons (MLPs). MLP network consist of a large number of simple processing elements (neurons). The neurons are commonly organized in layers and process data in the feed forward direction, i.e. from the network input to its output. Neurons are connected by weighted links. The weighs will be adapted during network training. The role of each single neuron is to sum the incoming signals, transform them according to its transfer function[14], and either deliver the results to subsequent layers for further processing, or generate network output. Typical form of MLPs is three layer perceptron. The structure of a three layer perceptron is shown in fig 2.

The MLPs are universal approximators. From Kolmogorov existence theorem, any continuous function of n variable can be approximated using a three-layered perceptron with $n(2n+1)$ nodes [10,15-16]. So, the accuracy of the approximation dose not depends on the number of the hidden layers, but it fully depends on the number of neurons in the hidden layer [15].

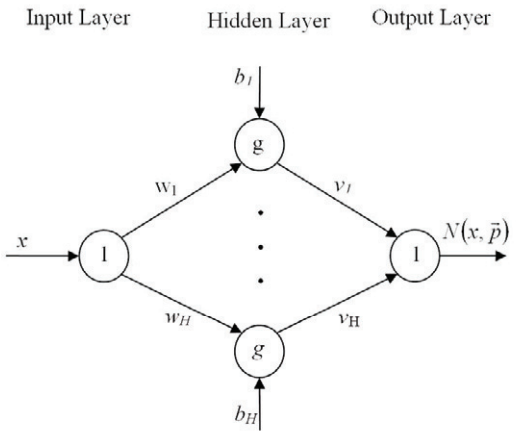


Fig. 2. A multi-layer perceptron

Fig.2 shows a three-layered perceptron with one input x, one hidden layer consisting of H neuron, and one output $N(x,p)$. This structure is used in the present method. For each entry x the network output is computed by following equation:

$$N(x,p)=\sum_{i=1}^{H}\left(v_i\, g\left(w_i\, x+b_i\right)\right) \tag{5}$$

where w_i is a weight parameter from input layer to *i*th neuron in hidden layer; v_i is a weight parameter from *i*th neuron in hidden layer to output layer, and b_i is a bias for ith neuron in hidden layer.

4. Particle Swarm Optimization

The Particle Swarm Optimization algorithm was first proposed by Eberhart and Kennedy [17] inspired by the natural flocking and swarming behavior of birds and insects. The concept of PSO gained in popularity due to its simplicity. Like other swarm-based techniques, PSO consists of a number of individuals refining their knowledge of the given search space. The individuals in a PSO have a position and a velocity and are denoted as particles. The PSO algorithm works by attracting the particles to search space positions of high fitness. Each particle has a memory function, and adjusts its trajectory according to two pieces of information, the best position that it has so far visited, and the global best position attained by the whole swarm. If the whole swarm is considered as a society, the first piece of information can be seen as resulting from the particle's memory of its past states, and the second piece of information can be seen as resulting from the collective experience of all members of the society. Like other optimization methods, PSO has a fitness evaluation function that takes each particle's position and assigns it a fitness value. The position of highest fitness value visited by the swarm is called the global best. Each particle remembers the global best, and the position of highest fitness value that has personally visited, which is called the local best. Particle size (n), inertia weight (ω) and maximum acceleration factor (a) are considered as important factors in PSO. The excellent reviews on PSO can be found in details in [17, 18].

5. Problem formulation

Let's assume $\theta_T(t,\vec{p})$ as approximate solution for eq (1), where $\vec{p}$ is a vector which contains adjustable parameters. These parameters (i.e. adjustable parameters) should be determined regarding to minimize the following sum of squared errors, subject to given initial conditions.

$$Error(\vec{p}) = \sum_{i=1}^{m} f\left(t_i, \theta_T\left(t_i,\vec{p}\right), \frac{d\theta_T\left(t_i,\vec{p}\right)}{dt}, \frac{d^2\theta_T\left(t_i,\vec{p}\right)^2}{dt^2} \right)^2 \tag{6}$$

In order to transform eq (6) to an unconstrained problem $\theta_T(t,\vec{P})$ is written as

$$\theta_T = \theta_0 + t^2 N(t,p) \tag{7}$$

where the first term (θ_0) satisfies the initial conditions and does not contain any adjustable parameters and the second term is constructed so as not to affect the initial conditions. $N(t,\vec{p})$ is a three-layer perceptron neural network which involves adjustable parameters (the weights and biases).
By using eq (5) and eq (6) general form of $\theta_T(t,\vec{p})$ is written as

$$\theta_T(t,\vec{P}) = \theta_0 + t^2 \sum_{i=1}^{H} \left(v_i\, g\left(w_i\, x + b_i\right)\right) \tag{8}$$

Although, there are many different choices for the neural network transfer functions, sigmoid function is used as transfer function in this study.
In order to calculate $Error(\vec{p})$ the derivatives of $\theta_T(t,\vec{p})$ respect to independent variable x is needed.
Derivation of $N(t,\vec{p})$with sigmoid transfer function $\left(\frac{1}{1+\exp(-t)}\right)$ is

$$\frac{d\theta_T}{dt} = \sum_{i=1}^{H} v_i\, w_i \left[-\left(\frac{1}{1+\exp\left(-\left(w_i\, t + b_i\right)\right)}\right)^2 + \left(\frac{1}{1+\exp\left(-\left(w_i\, t + b_i\right)\right)}\right)\right] \tag{9}$$

6. **Results**

The PSO algorithm was coded with MATLAB 2007. Combinations of user-specified parameters of PSO are used as follow: Inertia weight = 0.9, Population = 150 and Acceleration factors = 2.5.
In order to demonstrate the present method, eq (1) was solved for two different examples. The neural network results will be compared with exact solution and linear solution, in the form of tables and figures.

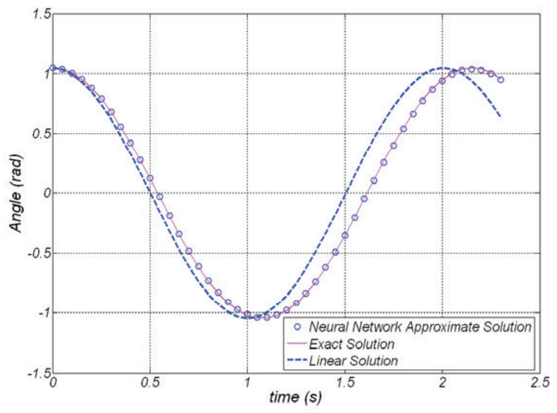
Example 1:
A simple pendulum with g/L =1that released from $\theta_0=\pi/3$.
The closed form of approximate solution for this example is obtained by minimizing the eq (8), using particle swarm optimization algorithm. The optimized parameters of eq (10) are given in table 1. A comparison between approximate solution and exact solution for this example is shown in figure (3-a) and in table 2.

Table 1: Optimal adaptive parameters of example 1 and example 2.

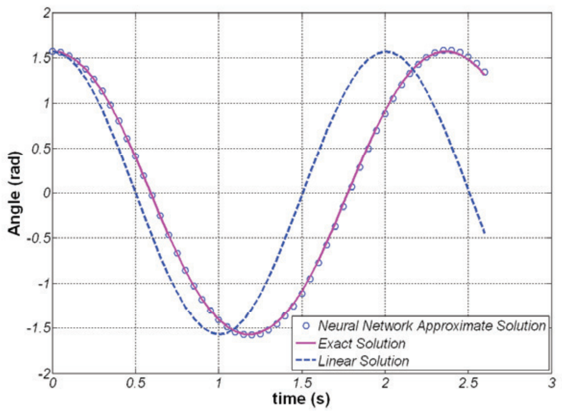
Example 1			Example 2		
Wi	bi	Vi	Wi	bi	Vi
-3.28993	1.457166	2.636525	3.182682	-4.88666	3.678795
2.662879	-5.69686	-1.68658	0.158035	-0.04514	-6.97696
-2.00643	1.923934	-7.47463	-0.8492	-0.07741	5.747183
4.783807	-3.14296	1.472062	-4.0008	3.377355	-4.26915

Table 2: A comparison between approximate solution and exact solution for example 1 and example 2

Example 1				Example 2			
$t\,(time)$	θ_{Exact}	θ_{ANN}	$\theta_{Exact}-\theta_{ANN}$	$t\,(time)$	θ_{Exact}	θ_{ANN}	$\theta_{Exact}-\theta_{ANN}$
0	1.047198	1.047198	0	0	1.570796	1.570796	0.00E+00
0.2	0.8802	0.880063	0.000138	0.2	1.37637	1.37637	0.001522
0.4	0.419638	0.421651	0.002013	0.4	0.804056	0.804056	0.002425
0.6	-0.1923	-0.18898	0.003325	0.6	-0.02926	-0.02926	0.006351
0.8	-0.73365	-0.72801	0.005643	0.8	-0.85765	-0.85765	0.002446
1	-1.02244	-1.01681	0.005629	1	-1.40801	-1.40801	0.002982
1.2	-0.98276	-0.97819	0.004569	1.2	-1.58025	-1.58025	0.010725
1.4	-0.62381	-0.61994	0.003871	1.4	-1.36688	-1.36688	0.0247
1.6	-0.04589	-0.05026	0.004373	1.6	-0.77225	-0.77225	0.030906
1.8	0.548954	0.53836	0.010594	1.8	0.069523	0.069523	0.037207
2	0.948899	0.939894	0.009005	2	0.885107	0.885107	0.031541
2.2	1.037773	1.027209	0.010564	2.2	1.428007	1.428007	0.004699



(a)



(b)

Fig. 3:A comparison between approximate solution and exact solution for (a): example 1 (b): example 2

Example 2:
A simple pendulum with $g/L=1$ that released from $\theta_0=\pi/2$.

The optimized parameters of approximate solution for this example are given in table 1. A comparison between approximate solution, exact solution and linear solution is shown in figure (3-b) and in table 2.

7. Conclusions

The physics of a simple pendulum for large angles is much more complicated than that for small angle because of the nonlinearity of the system. In the present paper an approximate solution for nonlinear pendulum using hybrid neural network and particle swarm optimization algorithm was introduced. The approximate solution is base on ability of neural network on function approximation and it is very close to the exact solution. The approximate solution can be use in engineering calculation and in laboratory instead of exact solution

References

[1] Serway AR, Beichner RJ., *Physics for Scientists and Engineers*. 5th ed. Harcourt Brace, Orlando, FL, 2000: 402–404.

[2] Xu F., A generalized soliton solution of the Konopelchenko-Dubrovsky equation using He's Exp-function method, *Naturforsch,* 2007;**62a**:685-8

[3] Hu H,.solution of a quadratic nonlinear oscillator by the method harmonic balance. *J. Sound Vib.* 2006;**293**:462-8.

[4] Zhang LN, Xu L., Determination of the limit cycle by He's parameter expansion for oscillators in a potential, *Naturforsch.* 2007;**62a**:396-8

[5] He JH., A new perturbation technique which is also valid for large parameters, *J. sound Vib.* 2000; **299**:1257-63.

[6] Ray SS., Exact solutions for time-fractional diffusion-wave equations decomposition method. *Phys Scr.* 2007;**75**:53-61.

[7] Assas, LM., Approximate solutions for the generaliz KdV-burgers equation by He's variational iteration method. *phys.Scr* . 2007;**76**:161-4.

[8] Amore P, Aranda A., Improved lindstedt-Poincare method for the solution nonlinear problems. *J. Sound Vib.* 2005;**283**:1115-36.

[9] He JH., Some asymptotic methods for strongly nonlinear equations. *J. Mod.Phys. B.* 2006;**20**:1141-99.

[10] Hornik, K, Stichcombe M, White H,. Multilayer feedforward networks are universal approximators. J. *Neural Networks* 1989;**2**:359.

[11] Kennedy J, Eberhart R., Particle swarm optimization", Proc neural networks. *Proceedings vol. IEEE international Conference on,* 1995;1942-1948.

[12] Ricardo H., *A Modern Introduction to Differential Equations*. 2th ed. Elsevier Academic Press, Canada, 2009. pp. 377-81.

[13] Milne-Thomson LM., in Abramowitz M, Ste-gun IA., (eds) *Handbook of Mathematical Functions* (Dover Publications, Inc., Nova Iorque, 1972).

[14] Cybenko G., Approximation by superposition of a sigmoidal function, Mathematics of Control, *J. Signals and Systems* 1989;**2:**303.

[15] Hornik K., Approximation capabilities of multilayer feedforward networks. *J. Neural Networks*. 1991;**4**:251.

[16] Lippmann RP., An introduction to computing with neural nets. *IEEE ASSP Magazine* 1987; 4–22.

[17] Shi Y, Eberhart R., Parameter selection in particle swarm optimization. *Proceedings of the Seventh Annual Conference on Evolutionary Programming*. New York, 1998;**b**:591-600**.**

[18] Yang IT., Performing complex project crashing analysis with aid of particle swarm optimization algorithm, Int. J. of Project Management, 2007;**25**:637-646.