FITTING PIECEWISE LINEAR FUNCTIONS USING PARTICLE SWARM OPTIMIZATION

Pavee Siriruk*

Received: February 28, 2013; Revised: March 12, 2013; Accepted: June 03, 2013

Abstract

The problem of determining a piecewise linear model for 2-dimensional data is commonly encountered by researchers in countless fields of scientific study. Examples of the problem or challenge are that of 2-dimensional digital curves, reliability, and applied mathematics. Nevertheless, in solving the problem, researchers are typically constrained by the lack of prior knowledge of the shape of the curve. Therefore, fitting a piecewise linear curve into a given set of data points is a useful technique. Moreover, any 2-dimensional continuous curve can be approximated arbitrarily by a piecewise linear function. In fitting a piecewise linear model, the number of segments and knot locations may be unknown. Several techniques can be employed, such as genetic algorithms as well as the least square error function, but not all techniques can guarantee convergence to a near optimal. In this paper, a method which employs the particle swarm optimization as the primary model fitting tool is introduced. The aim is to approximate a curve by optimizing the number of segments as well as their knot locations with fixed initial and final points. The experimental results which reveal the performance of the proposed algorithm are also presented.

Keywords: Optimization, particle swarm optimization, piecewise linear approximation

Introduction

One problem that invariably arises in determining the functional relationship between certain input and output variables is fitting a piecewise linear curve into a given set of data points. Reportedly, fitting a piecewise linear curve into a given set of data points is of great use when prior knowledge of the shape of the curve to be fitted does not exist (Kundu and Ubahya, 2001). As a result, the problem has received considerable attention from, and been studied by, researchers of various application fields, including engineering, chemometrics, and material science (Dunham, 1986; Pittman and Murthy, 2000). Alternatively, the same problem can be viewed as approximating any 2-dimensional continuous curve with a piecewise linear curve. Examples of study areas in which the

Department of Industrial Engineering, Suranaree University of Technology, Nakhon Ratchasima, Thailand.
E-mail: pavee@g.sut.ac.th
* *Corresponding author*

aforementioned problem is encountered are that of shape analysis (Dunham, 1986) and an approximation of the expected cost function during generator outages (Siriruk and Valenzuela, 2011).

A large number of techniques have been proposed to address the problem involving the creation of the piecewise linear curve. However, not all the techniques can guarantee convergence to a near optimal. Cantoni (1971) proposed an algorithm which determines the slopes and breakpoints that minimize the integral of the weighted squared error over the approximation interval. Later, Pittman and Murthy (2000) introduced a method by which genetic algorithms are employed to optimize the number and location of the pieces. Kundu and Ubhaya (2001) proposed an algorithm to optimize a piecewise linear continuous fit to a given set of data points by minimizing a weighted least square error.

Particle swarm optimization (PSO), which was first introduced in 1995, is a simple algorithm that seems to be effective for optimizing a wide range of functions (Kennedy and Eberhart, 1995). In this work, a new method which employs the PSO as the primary model fitting tool is introduced. In the case of piecewise linear functions, it is intended to optimize the number of segments as well as their knot locations in the model.

The Problem Statement

A set of data pointsis assumed:

$$D = \{x_i,y_i):1 \le i \le N\}, (x_i,y_i) \in \Re^2, 1 \le N \le \infty$$

where the values (x_i, y_i) have a relationship with an unknown function f such that $y_i = f(x_i)$. It is also assumed that $x_1 < x_2 <... < x_N$. The objective is to fit a model $\hat{f}(x)$ into a given set of data points. The following additional assumptions apply to the model $\hat{f}(x)$.

- $\hat{f}(x)$ is an h^*-piecewise linear function, where h^* is a positive integer representing the number of segments.
- Let $\{L_1, L_2, \ldots, L_{h^*}\}$ denote approximated slopes of the curve $\hat{f}(x)$, where the successive line segments $L_1, L_2, \ldots, L_{h^*}$ of the curve are listed from left to right.
- Let a_j and a_{j+1} be the left and right end points of L_j (knot locations), where $a_j = (u_j, v_j)$, $0 \le j \le h^* + 1$, with $u_0 = x_1$ and $u_{h^*+1} = x_N$.

The objective of this paper is to employ PSO to fit the function with the unknown shape of the curve by minimizing the sum of squared errors (SSE) between the original data points and the data points generated by $\hat{f}(x)$.

Algorithm Description

This section describes how PSO can be used to fit optimal piecewise linear function. One iteration of an algorithm consists of choosing knot locations using PSO, constructing model $\hat{f}(x)$, and calculating the evaluation function. The process continues until the stopping conditions are met.

If a starting point (x_1,y_1) is not the origin (0, 0), it is necessary to shift the starting point to the origin and thereby shift other data points in accordance with the shifted starting point. The following pseudo-code describes an algorithm to move the starting point:

FOR $i = 1$ *to* N
$x_i = x_i - x_1$
$y_i = y_i - f(x_1)$
END FOR

Notice that the new data range for x and y is $[0, x_N - x_1]$ and $[0, f(x_N) - f(x_1)]$, respectively. Once the starting point is at the origin, then a set of knot locations $(a_1, a_2, \ldots, a_{h^*+1})$ is selected in order to calculate the approximated slope for each segment. However, it is assumed that $a_1 = x_1$ and $a_{h^*+1} = x_N$. Thus, using PSO described below, there are simply chosen knot locations from a_2 to a_{h^*}, where $a_1 < a_2 <... < a_{h^*+1}$.

Particle Swarm Optimization

PSO is an iterative process which evaluates the solutions represented by the particle locations and adjusts the particle velocities based on prior knowledge. Each particle i maintains its current position, $A_i = (a_{i,1}, a_{i,2}, \ldots, a_{i,h^*+1})$, which is a set of knot locations. Each particle also maintains the velocity, $V_i = (v_{i,1}, v_{i,2}, \ldots, v_{i,h^*+1})$, and the best position that each particle has found so far, $P_i = (p_{i,1}, p_{i,2}, \ldots, p_{i,h^*+1})$. A population size of μ particles is initialized by randomly generating the vectors $\boldsymbol{A}$ and $\boldsymbol{V}$. The evaluation function is then used to evaluate the fitness of each particle. When the particle is moved, the new location and velocity are calculated based on A_i, V_i, P_i, and the best $\boldsymbol{P}$ vector from the neighborhood which is represented by $P_g = (P_{g,1}, P_{g,2}, \ldots, P_{g,h^*+1})$. The new velocity vectors are calculated using (1):

$$\begin{aligned} v_{i,d} &= K[v_{i,d} + \varphi_1 \bullet rnd() \bullet (p_{i,d} - a_{i,d}) \\ &\quad + \varphi_2 \bullet rnd() \bullet [p_{i,d} - a_{i,d}] \bullet \qquad (1) \\ \text{for } d &= 1,2,\ldots,h^*+1 \end{aligned}$$

where

$$K = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|}, \varphi = \varphi_1 + \varphi_2, \text{and } \varphi > 4.$$

Note that the values of φ_1 and φ_2 are set equally to 2.05, which are generally accepted values (Carlisle and Dozier, 2001). The new A_i vector is then calculated as below:

$$a_{i,d} = a_{i,d} + v_{i,d}. \qquad (2)$$

After computation, the new A_i vector must be sorted in the ascending order $(a_{i1} < a_{i2} < \ldots < a_{ih^*+1})$.

Constructing $\hat{f}(x)$

Once the knot locations are selected, the process of fitting piecewise linear curve into a set of data points can begin. The main steps of constructing $\hat{f}(x)$ are described below.

1) The first breakpoint is set at the first knot location (a_1).
2) A new breakpoint is obtained by moving to the next knot location.
3) All data points between the current breakpoint and the previous breakpoint are used to calculate the approximated slope using linear regression.
4) Steps 2) and 3) are repeated starting from the last breakpoint found so far.
5) The new model $\hat{f}(x)$ is obtained once the last knot location has been reached.

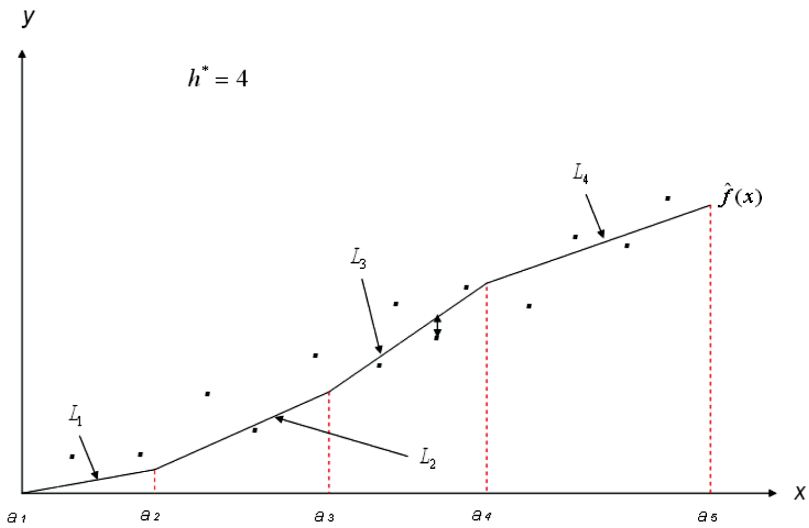

Figure 1. An outline of the evaluation function

All slopes in the new model connect to each other as depicted in Figure 1. A set of approximated slopes, $\{L_1,L_2,\ldots,L_{h^*}\}$, is obtained after applying the above procedure. The new model $\hat{f}(x)$ can be written as

$$\hat{f}(x_i) = \hat{f}(x_{i-1})+L_k \bullet (x_i-x_{i-1})$$
$$for\ \ a_n \le x_i \le a_n + 1 \text{ and } i \ne 1 \qquad (3)$$

where $n = 1,\ldots, h^*\text{-}1$ and $\hat{f}(x_1) = f(x_1)$. Note that the value of x_i in (3) is not the value that is shifted to the origin but the initial x_i value (i.e., prior to shifting).

The Evaluation Function

As mentioned earlier, the objective is to find the new model $\hat{f}(x)$ that can minimize the SSE and can be computed as below:

$$SSE = \sum_{i=1}^{N} [f(x_i)-\hat{f}(x_i)^2]. \qquad (4)$$

Therefore, the algorithm input is a set of knot locations $(a_1,a_2,\ldots,a_{h^*+1})$; and after computing for slope and implementing $\hat{f}(x)$, the algorithm output is the SSE computed by (4).

The process of choosing knot locations, constructing model $\hat{f}(x)$, and calculating evaluation function continues until the stopping conditions are met. Then the ***P*** vector with the lowest fitness value is chosen.

Numerical Examples

Methodology

In an effort to standardize the results of each dataset, a standard set of parameters is used. The population size can be described as:

$$\mu = \{2,4,8,16,32,64,128\}$$

All of the datasets use this parameter, but there is variation for each dataset. The algorithm continues until it exceeds the number of function evaluations as shown in Tables 1 and 2.

Table 1. Experimental dataset 1

Dataset 1	
Generating function	**Parameters**
$f(x)\begin{cases} 3.88x + 10.44 & -3.0 \le x \le -1.29 \\ -1.74x + 3.14 & -1.29 \le x \le 3.70 \\ 3.77x - 17.25 & 3.70 \le x \le 4.90 \end{cases}$	*N* : 791 *h** : 3 *# Function Evaluations:* 20,000 *Noise : Nor* (0,1)

Table 2. Experimental dataset 2

Dataset 2	
Generating function	**Parameters**
For $j = 1,2,\ldots,7$, $f(x) = \begin{cases} r_j c_j x+q_j f_{j+1}(x) \text{ for } 0 \le x \le P_j^{Max} \\ r_j\{c_j P_j^{Max}+f_{j+1}(x - P_j^{Max})\}+q_j f_{j+1}(x) \\ \qquad \text{for } x > P_j^{Max} \end{cases}$ For $j = 8$, $f_8(x) = xc_8$	*N* : 321 *h** : {4,5,6} *#Function Evaluations:* 4,000 *Noise:* -

Data

The proposed algorithm is applied to a couple of datasets in order to test its efficiency.

Dataset 1

Dataset 1 is borrowed from Pittman and Murthy (2000) in which it is generated from piecewise linear functions with unequally spaced knots of which the middle piece is considerably longer than either of the end pieces. This dataset uses normally distributed noise (ε). In other words, the function of dataset 1 can be expressed as $y = f(x) + \varepsilon$. Thus, the shape of this curve cannot be predicted. The parameters and generating function of dataset 1 are described in Table 1.

Dataset 2

Dataset 2 is borrowed from Siriruk and Valenzuela (2011). It is a recursive function which acts as a piecewise linear function with a large number of slopes. However, dataset 2 does not consider random noise. Unlike dataset 1, dataset 2 yields an exact set of data points. Therefore, the PSO algorithm can be tested to approximate the curve that does not include variation but does have a large number of slopes. The parameters and the generating function of dataset 2 are described in Table 2, the latter of which uses the data in Table 3.

Results

For dataset 1, the algorithm made 20000 function evaluations on each run. Therefore, there were 10000 iterations with a population size of 2. For a swarm size of 4, there were only 5000 iterations. The PSO algorithm found the best solution when $\mu = 16$. This population size returned 445.6074 SSE with 20000 function evaluations. The picture of the fitting curve is depicted in Figure 2.

For dataset 2, the overall best solution was found when the population size was $\mu = 4$ and the number of segments $h^* = 6$ (Table 4). With 4000 function evaluations, the SSE was found to be 78.33. At the swarm size $\mu = 128$ with the number of segments $h^* = 6$, the algorithm could not find better solutions. This indicated that the population size had an effect on the results. With a smaller population size, the algorithm will have more iterations than a larger population size. This indicates that each particle will have more time to adapt and search for better solutions.

When the number of segments h^* was was set to be 4 and 5, higher values of the SSE are obtained. This means that the function in dataset 2 did not fit well with these settings. The execution time of different values of $h^* = 4,5,6$ and $\mu = \{2,4,8,16,32,64,128\}$ was indistinguishable. All of the settings took less than 2 seconds to complete.

From Figure 3, it is clear that PSO provided a reasonable curve when the number of slopes (h^*) was 6. The number of slopes can efficiently be reduced to 6 with a small SSE remaining. In addition, the curve given by PSO is not much different from the original curve.

Conclusions

In this paper, particle swarm optimization has been employed to fit a function with the unknown shape of the curve by minimizing

Table 3. Data for generating function in dataset 22

j	1	2	3	4	5	6	7	8
r_j	0.99	0.99	0.9	0.98	0.96	0.98	0.98	-
q_j	0.01	0.01	0.1	0.02	0.04	0.02	0.02	-
c_j	1	1	8	14	14.8	14.8	14.8	100
P_j^{Max}	50	50	20	76	100	12	12	∞

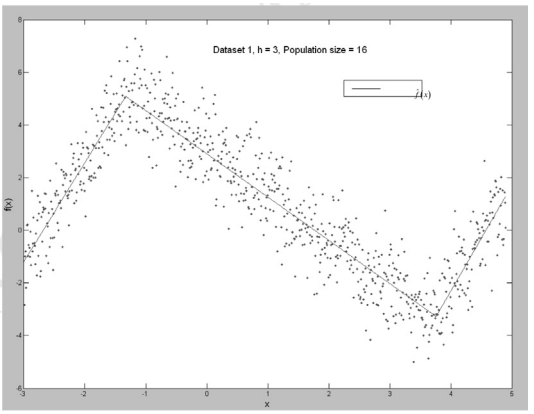

Figure 2. Results of dataset 1

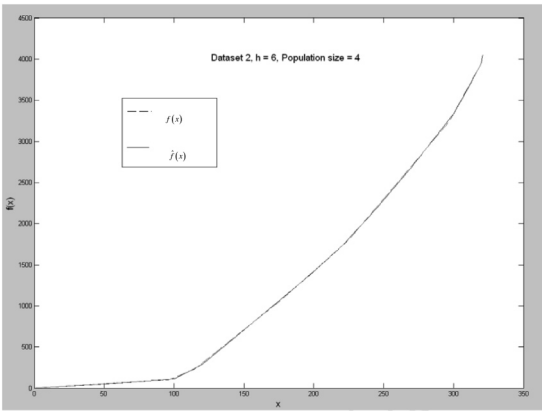

Figure 3. Results of dataset 2

the sum of squared errors. The proposed method has shown a good fit on a function for which there was no prior knowledge of the shape of the curve. Moreover, this algorithm performs well on approximating any 2-dimensional continuous curve with a piecewise linear curve with accuracy mostly intact.

References

Cantoni, A. (1971). Optimal curve fitting with piecewise linear functions. IEEE Trans. Comput., C-20(1):9-59.

Table 4. Results of dataset 2

h^*	μ	SSE
4	128	479
5	16	178.35
6	4	78.33

Carlisle, A. and Dozier, G. (2001). An off-the-shelf PSO. Proceedings of the 2001 Workshop on Particle Swarm Optimization; April 2001; Indianapolis, IN, USA, p. 1–6.

Dunham, J.G. (1986). Optimum uniform piecewise linear approximation of planar curves. IEEE Trans. Pattern Anal. Mach. Intell., PAMI-8(1):9-67.

Kennedy, R. and Eberhart, J. (1995). Particle swarm optimization. Proceedings of the IEEE International Conference on Neural Networks; November 27–December 1, 1995; Washington, DC, USA, p. 1942-1948.

Kundu, S. and Ubhaya, V.A. (2001). Fitting a least squares piecewise linear continuous curve in two dimensions. Comput. Matha. Appl., 41:9-1033.

Pittman, J. and Murthy, C.A. (2000). Fitting optimal piecewise linear functions using genetic algorithms. IEEE Trans. Pattern Anal. Mach. Intell., 22(7):17-701.

Siriruk, P. and Valenzuela, J. (2011). Cournot equilibrium considering unit outages and fuel cost uncertainty. IEEE Trans. Power Syst., 26(2):8-747.