Journal of Mathematics and Statistics 8 (1): 107-110, 2012
ISSN 1549-3644

Curve Fitting Using Conic by Evolutionary Computing

Zainor Ridzuan Yahya, Abd Rahni Mt Piah and Ahmad Abd Majid
School of Mathematical Sciences, University Sains Malaysia,
11800 USM, Pulau Pinang, Malaysia

Abstract: Problem statement: A direct method, such as least squares technique is usually used to solve problems involving matching a curve or a surface to a set of data points. The solution obtained by this direct method is precise or very good in approximation, but computationally not very efficient. Thus, in this study, we propose an indirect approach using Particle Swarm Optimization (PSO) technique as an alternative. **Approach:** As a case study, we use conic curve which satisfy C^0 continuity to be fitted to a given set of data points. PSO, a soft computing method is employed to optimize the control points and weights which are then used in conic equations. **Results:** Best fitted conic curve that represents all the given data points is then obtained. **Conclusion:** We use an indirect technique of soft computing methods, i.e., PSO to fit a curve to a given data set. We believe that other types of soft computing based heuristic procedures may also be used to solve related problems or to find its effectiveness.

Key words: Conics, curve fitting, particle swarm optimization method

INTRODUCTION

Curve fitting is the process of constructing a curve, or mathematical function that has the best fit to a series of data points, possibly subject to certain constraints. Curve fitting can involve either interpolation, where an exact fit to the data is required, or smoothing, in which a ’smooth’ function is constructed that approximately fits the data. A related topic in statistics is regression analysis, which focuses more on questions of statistical inference such as how much uncertainty is present in a curve that is fitted to observed data with certain random errors. Fitted curves can also be used as an aid for data visualization, to infer values of a function where no data are available and to summarize relationships among two or more variables. Extrapolation refers to the use of a fitted curve beyond the range of observed data and is subject to a greater degree of uncertainty which may also reflect the chosen method used to construct the curve as much as reflecting the error involved in the construction of fitted curve using observed data.

Particle Swarm Optimization (PSO) is an optimization technique proposed by Kennedy and Eberhart by means of particle swarm (Weise, 2007). PSO incorporates swarming behaviors observed in flocks of birds, school of fish, swarm of bees and even social behavior, from where the idea is originally emerged. Thus, PSO is a population based optimization tool which could be implemented and applied easily to solve various function optimization problems, or problems that can be transformed into function optimization forms. As an algorithm, the main strength of PSO is its fast convergence, which compares favorably with many global optimization algorithms like Genetic Algorithm (Eiben and Smith, 2007), Simulated Annealing (Kirkpatrick *et al*., 1983) and other global optimization algorithms. To apply PSO successfully, one of the key issues is finding how to map the problem solution into the PSO article, which directly affects its feasibility and performance.

Conic curve: A standard form of conics can be found in Yang (2004) and Farin (1989) Eq. 1:

$$r(t)=\frac{B_0^2(t)b_o+wB_1^2(t)b_1+B_0^2(t)b_2}{B_0^2(t)+wB_1^2(t)+B_2^2(t)} \tag{1}$$

Where:

$$B_i^2(t)=\binom{2}{i}t^i(1-t)^{2-i}$$

Corresponding Author: Zainor Ridzuan Yahya, School of Mathematical Sciences, University Sains Malaysia, 11800 USM, Pulau Pinang, Malaysia

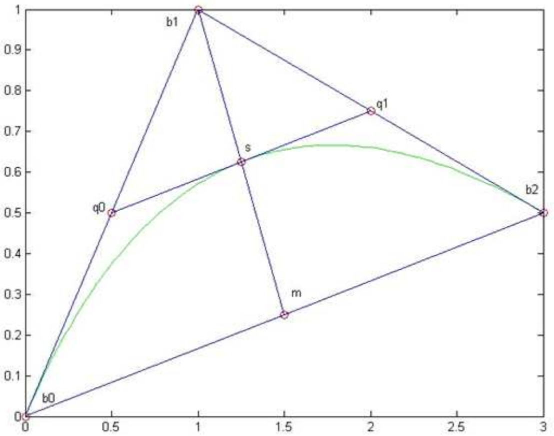


Fig. 1: The shoulder point: A rational quadratic with shoulder point **s** and shoulder tangent through q_0 and q1

and:

$$b_i\,(i = 0,1,2)$$

are control points of the Bezier curve and w is the middle weight. Here we list some useful properties of conics: For w < 1, we obtain an ellipse; for w = 1 a parabola; and for w > 1 a hyperbola:

- The straight line segments $[b_0, b_1]$ and $[b_1, b_2$ are tangents to r at $r(0) = b_0$ and $r(1) = b_2$, respectively
- For $w \geq 0$, the curve segment (1) lies in the convex hull of the control polygon
- The point s = r(1/2) of a conic segment in its standard form is called the shoulder point. It can be computed from:

$$s = \frac{1}{2}q_0 + \frac{1}{2}q_1$$

Where:

$$q_0 = \frac{b_0 + wb_1}{1+w}, q_1 = \frac{wb_1 + b_2}{1+w}$$

are called the characteristic points. The shoulder tangent is spanned by q_0 and q_1. Note that the shoulder tangent is parallel to $\langle b_0, b_2 \rangle$; see Fig. 1. As a consequence:

$$w = \frac{\|s - m\|}{\|b_1 - s\|}$$

where, m is the midpoint of b_0 and b_0.

Particle Swarm Optimization: PSO starts by having a population of particles initialized with random positions marked by vector $\overrightarrow{x_i}$ and random velocities $\overrightarrow{v_i}$ (Das *et al*., 2008). The population of such particles is called a 'swarm' S. Each particle P has two state variables viz., its current position $\overrightarrow{x_i}(t)$ and its current velocity $\overrightarrow{v_i}(t)$. It is also equipped with a small memory which comprises of its previous best position $\vec{p}(t)$, i.e., the personal best experience and the best $\vec{p}(t)$ of all P, $\vec{g}(t)$ i.e., the best position found so far in the neighborhood of the particle. The PSO scheme has the following algorithmic parameters:

- V_{max} or maximum velocity which restricts $\overrightarrow{V_i}(t)$ within the interval $[-V_{max}, V_{max}]$ an inertia weight factor ω
- TWO uniformly distributed random numbers, $0 \leq \varphi_1, \varphi_2 \leq 1$ on the velocity update formula
- two constant multiplier terms C_1 and C_2 known as 'self-confidence' and 'swarm confidence', respectively

Initially the settings for $\vec{p}(t)$ and $\vec{g}(t)$ are for all particles. Once the particles are all initialized, an iterative optimization process begins, where the positions and velocities of all the particles are altered by the following recursive equations.

The equations are presented for the d^th dimension of the position and velocity of the i^th particle Eq. 2 and 3:

$$\begin{aligned} V_{id}(t+1) &= \omega v_{id}(t) + C_1\varphi_1(p_{id}(t) - x_{id}(t)) \\ &+ C_2\varphi_2(g_{id}(t) - x_{id}(t)) \end{aligned} \tag{2}$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \tag{3}$$

The first term in the velocity updating formula represents the inertial velocity of the particle. Since the coefficient C_1 has a contribution towards the self-exploration (or experience) of a particle, we regard it as the particle's self-confidence. On the other hand, the coefficient C_2 has a contribution towards motion of the particles in global direction, which takes into account the motion of all the particles in the preceding program iterations, naturally its definition 'as swarm confidence' is apparent. After having calculated the velocities and position for the next time step t+1, the first iteration of the algorithm is completed. Typically, this process is iterated for a certain number of time steps, or until some acceptable solution has been found by the algorithm or until an upper limit of CPU usage has been reached. The algorithm is summarized in the following pseudo code:

The PSO Algorithm:

Input: Randomly initialized position and velocity of the particles: $\overrightarrow{X_i}(0)$ and $\overrightarrow{V_i}(0)$.

Output: Position of the approx. global optima $\overrightarrow{X*}$

```
Begin
While terminating condition is not reached
For i = 1 to number of particles
Evaluate the fitness:= $f(\overrightarrow{X_i})$
Update $\overrightarrow{p_i}$ and $\overrightarrow{g_i}$;
Adapt velocity of the particle using Eq. 3;
Update the position of the particle;
Increase i;
end while
end
```

Proposed technique: In our proposed method, we estimate the control points by finding the intersection point between the point connecting two adjacent points at both ends of the curve (Fig. 2). Then we choose the size of the search area to find the best value for the control point. For example, we choose the search space size = 1 where the control point is identified in the search space (Fig. 3). To find the best weight, an initial value for the weight will be assigned at random in the range of [0, 1].

Our proposed algorithm starts by having an initial swarm position with swarm size =25 Fig. 4. PSO is used to find the best control point and weight, to enable us to generate the best fitted curve to the given data. To obtain the best solution, we use the sum squares error method proposed by Yahya *et al*. (2006). We set number of iterations = 50, inertia $\omega = 1.0$ and $C_1 = C_2 = 2.0$. The pseudo code of our proposed technique is given below. Figure 5 displays the results.

Curve fitting algorithm:

```
Get data points;
Approximate intermediate control point;
Determine search space to find intermediate control points;
```

Initialization:

```
Initialize intermediate control point for each particle;
Initialize weight (randomly chosen from [0, 1]) for each particle;
Best Solution      = 1000;
Best Control Point = (0,0);
Best Weight        = 1;
For  i             =1  until last iteration
For  j             =1  until number of particle
Update the control point of the particle;
Update the weight of the particle;
Evaluate solution (new Solution);
If new Solution better than best Solution
```

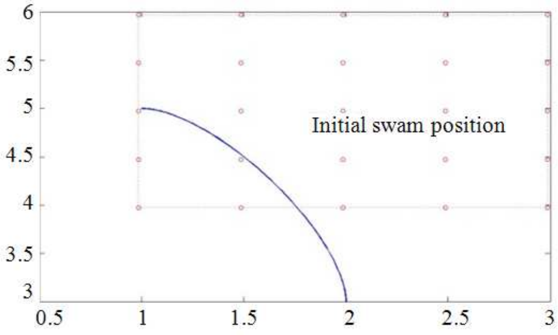


Fig. 2: Approximation of intermediate control point

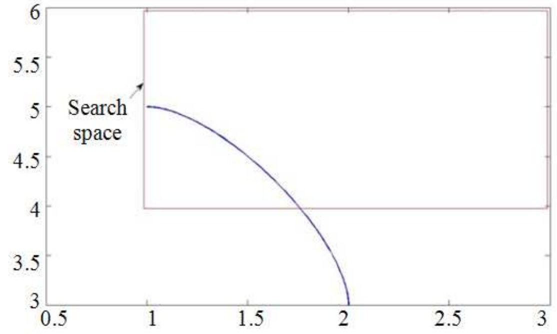


Fig. 3: Search space for finding intermediate control point

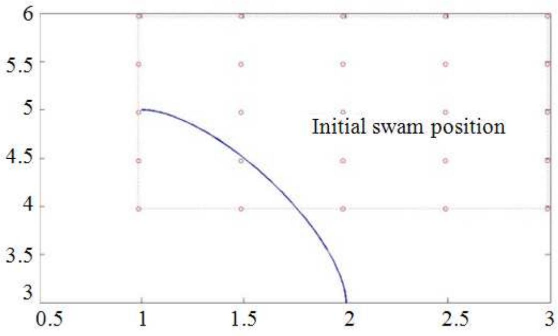
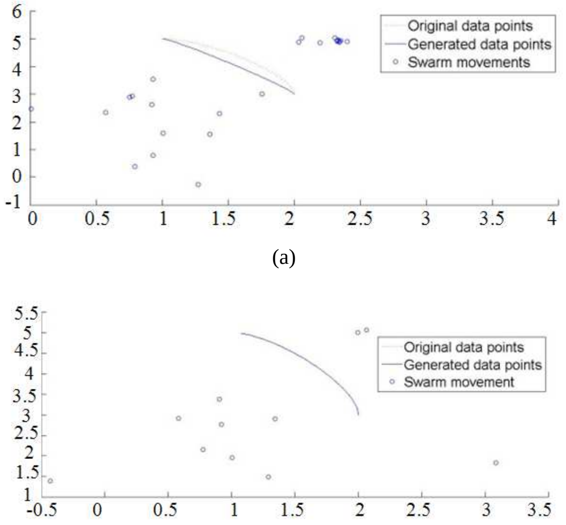


Fig. 4: Initial swarm position of intermediate control point



(a)

(b)

Fig. 5: The swarm movements to find the best intermediate control point (a) After 10th iteration (b) Final iteration

```
Best Solution = new Solution;
Best Control Point = new Control Point;
Best Weight = new Weight;
End If
Update velocity for control point and weight;
increase j;
end For
increase i;
end For;
```

CONCLUSION

Instead of using one of the usual direct techniques, we use an indirect technique of soft computing methods, i.e., PSO to fit a curve to a given data set. In order to find the best solution, we map the curve fitting problem to the best PSO scheme. In the future, we intend to use a similar algorithm in order to solve surface fitting problem. The effects of changed parameters found in the PSO scheme will also be studied.

ACKNOWLEDGEMENT

The researchers would like to thank University Sains Malaysia for supporting this research under its Research University Grant, 1001/PMATHS/817049.

REFERENCES

Das, S., A. Abraham and A. Konar. 2008. Particle swarm optimization and differential evolution algorithms: Technical analysis, applications and hybridization perspectives. Adv. Comput. Intell. Indus. Syst., 116: 1-38. DOI: 10.1007/978-3-540-78297-1_1

Eiben, A.E. and J.E. Smith, 2003. Introduction to Evolutionary Computing. 1st Edn., Springer, Berlin, ISBN: 3540401849, pp: 299.

Farin, G., 1989. Curvature continuity and offsets for piecewise conics. ACM Trans. Graph., 8: 89-99. DOI: 10.1145/62054.62056

Kirkpatrick, S., C.D.J. Gelatt and M.P. Vecchi. 1983. Optimization by simulated annealing. Science, 220: 671-680. DOI: 10.1126/science.220.4598.671

Weise, T., 2007. Global optimization algorithms-theory and application. University of Kassel.

Yahya, F., J.M. Ali, A.A. Majid and A. Ibrahim, 2006. An automatic generation of G^ 1 curve fitting of arabic characters. Proceedings of the International Conference on Computer Graphics, Imaging and Visualization, Jul. 26-28, IEEE Xplore Press, Sydney, pp: 542-547. DOI: 10.1109/CGIV.2006.18

Yang, X., 2004. Curve fitting and fairing using conic splines. Comput. Aided Design, 36: 461-472. DOI: 10.1016/S0010-4485(03)00119-2