Multi-layer optimization algorithm

M Abdalla, J Yamin and E Al-Khawaldeh

Abstract
The optimization of experimental results has repeatedly posed major challenges for scientists and engineers. In this work, a systematic multi-layer optimization scheme is proposed in conjunction with particle swarm optimization algorithm to locate a global optimum that fits a cost function. The technique utilizes SP-lines to form three-dimensional patches surfaces from experimental data in multi-layer fashion and incorporates a multi-layer search using particle swarm optimization. The novel technique is illustrated and verified over two layers of experimental data to show its effectiveness.

Keywords
Data representation, multi-layer, optimization, surface generation

Received 17 August 2020; Revised received 9 July 2021; accepted 31 October 2021

Introduction
Researchers use experiments to answer questions by following the scientific method. Consequently, experiments design and results analysis are used to verify and validate the initial hypothesis or assertions. Typically, many trials of the experiments are carried out producing huge amounts of multivariable and multidimensional data. Even though there are many references on experimental design and statistical data analysis, but only a few addresses the data representation and search for an optima, especially for multivariate experiments, where the collected data from the repetition of the trials upsurge.

Currently, in literature, the optimization of experimental results has constantly posed major challenges for scientists and engineers. In such multivariate problems, the use of non-linear techniques like design of experiments (DoE) is used to explore the interaction effects of experiment variables. The DoE is considered the most effective and economical technique to evaluate the individual and combined effects of experiment variables on output responses.

Other multivariate optimizing techniques exist; such as the response surface methodology, which is a collection of statistical and mathematical methods, is well known as a useful method to analyze engineering problems based on both modeling and optimizing the response surface that is influenced by experiment variables. Neural Nets (NNs) and Genetic Algorithms (GAs) are used to solve multi-objective optimum problems. More techniques are found that use the knowledge-based learning extreme method with cuckoo search to determine the optimum points.

Historically, polynomials were used due to their attractive mathematical characteristics. Through the advent of computers, splines have gained high importance. They were first used as a replacement for polynomials in interpolation, then as a tool to construct smooth and flexible shapes in computer graphics.

The complexity of optimization strategies increases when dealing with experimental data. In this work, a novel Multi-layer Optimization (MLO) strategy that may be used to deal with experimental data is proposed. The techniques utilize the well-developed generated surfaces that are widely used for modeling in aerodynamically constrained shapes. Once the data is fitted by such surfaces and arranged in multi-layers then newly developed search techniques are used to hunt for optimum points based on a cost function.

The generation of data fitted surfaces is possible using many of the well-developed techniques. Representations for surfaces such as polynomial surface patches and
tensor product cubic splines or interpolation methods such as bilinear interpolation are very popular. Approximation of surfaces using variational methods and regression splines is another option. Segmentation of point measurements into surface patches are used by many researchers. Finally, registration of surfaces with point measurements is another field area of such surface generating techniques.

**Multi-layer graphing overview**

Piecewise linear approximations of experimental data require many pieces to express reasonable, realistic and smooth surfaces. On the other hand, the set of experimental data individual surface points would take large amounts of storage. Fitting a single formula for a surface if possible would be nice, but most of the time is not possible. Hence, a collection of surface patches that utilize piecewise simple formulae might work with a reasonable amount of storage. The disadvantage would be how to specify and render such surface patches.

Fitting surfaces to data point is of interest to many industries. Commonly use of splines for the development of modeling in the aircraft and automobile industries is popular and widely used.9–12 Surfaces may be captured and defined by Parametric or Implicit formulae. A parametric representation of the form:

$$ S(x, y, z) = (f(u, v), g(u, v), h(u, v)) $$

enable us to span the surface space using only two independent variables, which make it attractive for generating meshes and polygons but hard for morphing and intersections. On the other hand, implicit formulae of the form:

$$ S(x, y, z) = 0 $$

are excellent for morphing and intersections but cannot be used easily for iterative schemes to span the surface space. Systematic methods to generate known curved surfaces are possible with sweeping and rotation. But for more complex surfaces one may approximate desired surface spaces by patches. Parametric, Polynomial, Bilinear, Bézier, B-Spline patches may be used to generate surface patches.

In this work, cubic Splines are used to represent experimental data and create a mesh of the unknown surface space. Once the mesh is constructed the generated surface patches borders are used to encapsulate using a surface polynomial. Hence, a surface patch border is fitted using a cubic Spline with known border location that are used as a constraint. The following equations may generated for the surface patch at the designated locations.

$$ z = b_3 x^3 + b_2 x^2 + b_1 x + b_0, \quad y = y_0 $$

Substituting the known boundary points into equation (7) and comparing with the corresponding equation (3)–(6), then the resulted auxiliary equations maybe represented into the following matrix equation.

$$ \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & \gamma_0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & \gamma_0 & \gamma_0 & 0 & 0 \\
0 & 0 & 0 & \gamma_0^3 & \gamma_0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & \gamma_1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \gamma_1^3 & \gamma_1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & x_0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & x_0^3 & 0 & x_0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & x_1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & x_1^3 & x_1 & 0 & 0 \\
\end{bmatrix} \begin{bmatrix}
e_1 \\
e_2 \\
e_3 \\
e_4 \\
e_5 \\
e_6 \\
e_7 \\
e_8 \\
e_9 \\
e_{10} \\
e_{11} \\
e_{12} \\
e_{13} \\
e_{14} \\
e_{15} \\
\end{bmatrix} = \begin{bmatrix}
a_3 \\
a_2 \\
a_1 \\
a_0 \\
b_3 \\
b_2 \\
b_1 \\
b_0 \\
c_3 \\
c_2 \\
c_1 \\
c_0 \\
d_3 \\
d_2 \\
d_1 \\
d_0 \\
\end{bmatrix} $$

In a more compact form and using matrix notation, the following is adopted.

$$ Aq = w $$

where, $A \in \mathcal{R}^{nxm}$, $q \in \mathcal{R}^{nx1}$, $w \in \mathcal{R}^{nx1}$. One may use Moor Penrose generalized inverse13 to solve for the unknown surface polynomial coefficients, which may be computed using singular value decomposition.
(SVD). For the matrix $A \in \mathbb{C}^{m \times n}$ with rank $r$, the SVD is

$$A = UD^*V$$

(9)

where $U \in \mathbb{C}^{m \times m}$ and $V \in \mathbb{C}^{n \times n}$ are unitary matrices, and $D \in \mathbb{C}^{m \times n}$ is a diagonal matrix of the singular values, $s$, of the $A$ matrix. The singular values are the square roots of the Eigen-values of the square matrices $A^*A$ or $AA^*$ which are the same values, and the number of singular values is equal to the rank of $A$.

$$D = \begin{bmatrix} S & 0 \\ 0 & 0 \end{bmatrix}$$

(10)

where $S$ is the $r \times r$ of singular matrices. Rows $\{(n-r) \ldots n\}$, and columns $\{(m-r)\ldots m\}$, of the $D$ matrix are full of zeros.

$$A^+ = VD^*U^*$$

(11)

where $A^+ \in \mathbb{R}^{m \times n}$ and

$$D^+ = \begin{bmatrix} S^{-1} & 0 \\ 0 & 0 \end{bmatrix}^*$$

and $S^{-1} \in \mathbb{R}^{r \times r}$ matrix with $s_{i}^{-1} = (1/s_{i})$ along the diagonals.$^{14}$

Finally, the solution to the matrix equation (8) $q = A^+w$ is unique if $A^+A = I$ (i.e. from the uniqueness theorem).

From the previous derivations, the piece-wise defined patch surfaces could be attained. Hence, the overall surface constitutes the collection or the union of the patched surfaces that are represented by equation (7).

$$\mathcal{Z} = \cup_{i=1}^{n} \mathcal{Z}_{i}$$

(12)

Particle swarm optimization overview

The particle swarm optimization (PSO) has been shown to be an effective optima searching technique for difficult multidimensional none-convex problems.$^{15}$ Recently, the PSO has been successfully applied to solve large scale optimization problems in a diversity of fields. PSO was shown that for certain problems it actually outperformed GAs.$^{16}$ It is considered a stochastic evolutionary computational technique with minimal computational effort when compared with GA. Further analysis shows that the difference in computational effort between PSO and the GA is problem-dependent. It appears that PSO outperforms the GA with a larger differential in computational efficiency when used to solve unconstrained nonlinear problems with continuous design variables and less efficiency differential when applied to constrained nonlinear problems with continuous or discrete design variables.$^{17,18}$

Kennedy and Eberhart have discovered the Particle Swarm Optimizer while they were trying to model the social behavior of artificial life, such as bird flocking, fish schooling, ...etc.$^{15}$ A good metaphor of the PSO algorithm is to imagine a swarm of bees in a field searching for flowers. The objective of the swarms is to find the location of the most flowers. Basically, each bee will conduct a random search and it will memorize the densest location of flowers it encountered.

Now, under the assumption that all the individual swarms may share their best discoveries, then each individual swarm is guided by its own finding and it is influenced or swayed by the others reported best locations. Consequently, the swarms’ trajectories are dynamically changing and they will span the field to explore the densest flower locations. That is the swarms’ decisions are drawn from their own experience and others’ experiences and a blend of the two effects is created to create a flying trajectory.$^{19}$

The particles are moving and sweeping the solution space intelligently and it also summarizes the used pseudo-algorithm. Each particle in the swarm tracks its position by means of two vectors; one accelerates the particle in the direction of its own and another towards a global best for the whole swarm (best of the bests). The advancement of the particles in the solution space is controlled by a simple kinematic equation of the form:

$$p_{i}(t+1) = p_{i}(t) + \Delta t \cdot v_{i}(t)$$

(13)

where the $\Delta t$ is the advancement in time (increments) and $p_{i}$ is the current location of the particle (i.e. the solution). On the other hand, the particles’ velocities are updated during the search with the following formula:

$$v_{i}(t+1) = wv_{i}(t) + c_{1} \cdot rand() \cdot (p_{b_{i}}(t) - p_{i}(t))$$

$$+ c_{2} \cdot rand() \cdot (p_{Global} - p_{i}(t))$$

(14)

where $w$ is a scalar that generates some form of momentum for the particle taken from the previous iteration. The value that is used in this paper is 0.75, however, this value could be changed dynamically as well (in an adaptive fashion). The constants $c_{1}$ and $c_{2}$ represent the emphasis toward the particle’s best or toward the global swarm’s best weighted by random terms. Typical values for these weights are within the interval [0 4], but good reported values in the literature of the scalars are $c_{1} = c_{2} = 2$. The local best is the best value of the solution attained by the particles and the global best is the best value of the solution attained by the whole swarm. The stochastic behaviour of the PSO algorithm may be realized through the given two random sequences: $rand() \in [0, 1]$. Please note that the PSO algorithm is structured in such a way it will not get stuck into local optima.
PSO pseudo algorithm

**Step I**: Initialize position, velocity, local best, and global best of all the particle swarms
\[ p_i = \text{rand}(i = 1, 2, \ldots, n) \]
\[ v_i = \text{rand}(i = 1, 2, \ldots, n) \]
\[ p_{i, \text{Best}} = p_i \; \; (i = 1, 2, \ldots, n) \]

compute fitness function \( J_i \) for each \( p_{i, \text{Best}} \):
\[ p_{\text{Global}} = p_{i, \text{Best}} \; \; \text{that corresponds to } \; \min(J_i) \]

**Step II**: Repeat until fitness function of global position is acceptable or \( m \) iterations is reached

compute fitness \( J(p_i) \):
\[ J(p_i) < J(p_{i, \text{Best}}) \; \; \text{then} \; \; p_{i, \text{Best}} = p_i \]
\[ p_{\text{Global}} = p_{i, \text{Best}} \; \; \text{if} \; \; \min(J(p_{i, \text{Best}})) < p_{\text{Global}} \]
update particle velocity
\[ v_i = wv_i + c_1 \text{rand}(p_{i, \text{Best}} - p_i) + c_2 \text{rand}(p_{\text{Global}} - p_i) \]
update particle position
\[ p_i = p_i + \Delta v_i \]
loop step II

**Step III**: Optimal solution = the particles with best global fitness values

The MLO algorithm

A systematic MLO algorithm is summarized in this section. The MLO algorithm was mainly devised to optimize experimental data results with multi-variables.

MLO pseudo algorithm

**Step I**: Prepare and initialize the algorithm with the experimental data sets to be optimized

**Step II**: Implement SP-Lines fitting to form 3D-patches of ‘n’ surfaces based on the experimental data.

For \( i = 1 \) to \( n \)
\[ q_i = A_i^T w_i \]

Form \( z_i \); # Patched surfaces
end for \( i \)

**Step III**: Organize the generated patched surfaces, which is generated by step II, into a multi-layer fashion to form the problem solution space \( S \)
\[ S = \bigcup_{i=1}^{n} z_i \]

**Step IV**: Implement a stochastic PSO search algorithm to operate on the solution space \( S \)

Min \( J = \) Experimental Data Performance Index
Subject to: Exp. Data Multi-Layer Patched Surfaces

where \( J \) is measurable performance index functional, \( J: \mathbb{R}^n \to \mathbb{R} \) and a convex solution space \( S \subseteq \mathbb{R}^n \) we seek an optimum point \( p_{\text{Global}} \in S \) that minimizes \( J \) on \( S \).

The local convergence of the aforementioned algorithm depends on large on two conditions; namely, an algorithm condition and a convergence condition. The algorithm condition guarantees that new algorithm estimates cannot be worse than the current ones, while the convergence condition describes the ability of the algorithm to get closer to the optimality region in each iteration.

**Algorithm condition.** Let \( D \) be the mapping that combines the new position of one PSO particle \( p_i = p_i + \Delta v_i \) with the current best obtained PSO solution \( p_{i, \text{Best}} \). The optimality region may be defined as: \( R_\varepsilon = \{ p_i \in S : J(p_i(p_{i, \text{Best}})) < \varepsilon \} \), consequently the mapping \( D: S \times \mathbb{R}^n \to S \) should satisfy \( J_i(D(p_{i, \text{Initial}}, p_i(\Delta t))) \leq J_i(p_{i, \text{Best}}) \) and if \( p_i(\Delta t + 1) \notin S \) then \( J_i(D(p_{i, \text{Initial}}, p_i(\Delta t))) \leq J_i(p_i(\Delta t + 1)) \) where \( t \in \mathbb{N} \).

The algorithm condition verifies if the algorithm can sample a point in the optimality region, where one can think of this optimality region as the set of points around the optimum of the solution space in which the fitness of the PSO particle is lesser than the optimal solution value plus a strictly positive tolerance \( \varepsilon \). In other terms, at each iteration, if the new obtained fitness is better than the previous one, then the new one replaces the best fitness. Consequently, this algorithm condition, stipulates that the new solution suggested by \( D \) cannot be worse than the current solution.

**Convergence condition.** A local optimum is obtained by the following sufficient condition: \( \forall p_i \in S, \exists \delta \in \mathbb{R} > 0 \) and \( 0 < \lambda \in \mathbb{R} < 1 \) such that \( \mu_i(dist(p_i(\Delta t), R_\varepsilon)) = dist(p_i(\Delta t), R_\varepsilon) - \delta \) or \( p_i(\Delta t) \in \mathbb{R} \geq \lambda \), where \( \mu_i \) is the probability measure and \( p_i(\Delta t) \) is the \( i \)th PSO particle solution at iteration \( \Delta t \) and \( dist(p_i(\Delta t), R_\varepsilon) = inf_{r \in R_\varepsilon}(dist(p_i(\Delta t)) \cup r) \).

The second condition is the convergence condition that describes the ability of the algorithm to have a non-zero probability to get closer to the optimality region or sample a point within this region at each iteration. Showing that the algorithm satisfies the algorithm condition and convergence condition is sufficient to prove that the algorithm at least converges to a local minim, complete detailed proof of these conditions may be found in the following reference.

Consequently, the proposed algorithm satisfies the algorithm condition, while the convergence condition guarantees that the particles’ trajectory is convergent.
and the particles will keep moving closer to the global best particle. The effectiveness of the proposed algorithm and its convergence is demonstrated using a real engineering application.

**Numerical application**

In order to demonstrate and verify the effectiveness of the proposed optimization algorithm consider the popular internal combustion fuel mixing problem. A certain low grade of Gasoline is selected and prepared to be mixed with different concentrations of Methanol with the objective of achieving better engine output measured performance. In this application, consider nominal Gasoline with Octane Number of 89, which will be mixed with different percentages of Methanol. The desired volume percentages of Methanol that are to be added are 3%, 6%, 9%, 12% and 15%. The Methanol percentages for this study were selected such that the Methanol percentages were varied till the octane number equals to 100 (i.e. at 15% methanol) is achieved. The tests were carried out at the University of Jordan’s laboratory, to study the effect of these fuel blends on the engine’s performance. This was done using the TecQuipment TD43 engine that is depicted in Figure 1. This is a power test 4-Stroke, single cylinder, variable compression ratio engine having displacement volume of 582 cm$^3$ with an electrical dynamometer that is directly coupled to it. The experimental data of brake torque, brake power, air and fuel flow rate were directly obtained from the engine control panel, while the ambient conditions were obtained through the fitted instruments.

With the engine compression ratio fixed at 8, and the fuel flow control knob opened at its optimum level as recommended by the manufacturer, and at engine speed of 1500 RPM the ignition angles were varied to find the best torque angle. An optimum angle was found to be 15 degrees before top dead center, and it was fixed for the rest of the experiments. All measurements were performed under wide open throttle and variable speed conditions, over the range of 1000 to 2500 rev/min in steps of 250 rev/min. More details on the experimental work and data analysis maybe found on references.\textsuperscript{21–23}

**The optimization problem**

Where it is of practical interest to study the performance of internal combustion engines using multi-component fuels. This application presents an experimental investigation into the effect of Methanol over carbon monoxide (CO) and hydrocarbon (HC) emissions level. All the experiments were carried out under the same conditions (as much as possible) for uniformity and with variable speed conditions. To demonstrate the proposed technique let’s consider two sets of experimental data that require the simultaneous optimization of using the following cost function.

$$\text{Min } J = \alpha \text{ CO} + \beta \text{ HC}$$

Subject to \(\text{Exp. Data}\)  

(15)
where $\alpha$ and $\beta$ are weights that may be used to emphasize the importance of the CO or HC effects.

There are many engineering methods; such as Kent-Eisenberg and artificial neural network (ANN), that are widely used to correlate and model diversity of experimental data. Such techniques are used because direct mathematical modeling is either complex or in some cases not possible. Once the model is available different predictions and optimum parameter studies may be performed. In this work, an experimental driven data based mathematical model was build, tested and verified using the ANN technique and the optimum conditions for engine power, torque and specific fuel consumption were found. The model neural network’s diagram is shown in Figure 2.

The activation function for the hidden layers was chosen to be the Gaussian function, where the inputs to the network were the engine speed (RPM) and calorific values ($Q_{cv}$). Several number of hidden layers were tested and the best results were obtained with seven layers. The equations for the hidden layers for the CO predictions is given by equations $H_1$ through $H_7$ as shown in the equation below.

$$H_1 = e^{-0.5 (2.89570745865 - 0.000991903622948 \text{ RPM} - 0.0000663467800677 \text{ Q}_{cv})^2}$$

$$H_2 = e^{-0.5 (2.89570745865 - 0.000991903622948 \text{ RPM} - 0.0000663467800677 \text{ Q}_{cv})^2}$$

$$H_3 = e^{-0.5 (-25.1858695377 - 0.000269395230352 \text{ RPM} - 0.000603613366672 \text{ Q}_{cv})^2}$$

$$H_4 = e^{-0.5 (-7.53006475986 - 0.000791000013220 \text{ RPM} - 0.00166292211130 \text{ Q}_{cv})^2}$$

$$H_5 = e^{-0.5 (-37.4537823441 - 0.00087778841299 \text{ RPM} - 0.000091027102728 \text{ Q}_{cv})^2}$$

$$H_6 = e^{-0.5 (1.78169208247 - 0.000180631746009 \text{ RPM} - 0.000091027102728 \text{ Q}_{cv})^2}$$

$$H_7 = e^{-0.5 (41.3539098193 - 0.0000157856579561 \text{ RPM} - 0.00105637983882 \text{ Q}_{cv})^2}$$
The ANN identified the CO modeling equation for the measured engine data, which maybe used for the prediction of CO as follows:

\[
CO = 16.0271212731 - 5.48935065574788 H_1 \\
- 3.75714445655 H_2 + 0.330543236564 H_3 \\
- 4.86069317339 H_4 + 1.04168106001 H_5 \\
- 0.52343251169 H_6 + 3.20300403855 H_7 \\
\]

(17)

Finally, the equations for the activation function for the HC model are shown in equations \( H_1 \) through \( H_7 \).

\[
H_1 = e^{-0.5 (-67.340637877304 - 0.0036712068685159 RPM - 0.0015469043339997 Qcv)^2}
\]

\[
H_2 = e^{-0.5 (-64.526000576958 - 0.002546652673902 RPM - 0.0015284945010943 Qcv)^2}
\]

\[
H_3 = e^{-0.5 (-39.690335616333 - 0.001433544052539 RPM - 0.00084820593428518 Qcv)^2}
\]

\[
H_4 = e^{-0.5 (66.1326007189146 - 0.003454660619485 RPM - 0.0015264476855533 Qcv)^2}
\]

\[
H_5 = e^{-0.5 (39.7227564422573 - 0.001442845736732 RPM - 0.00084862488487913 Qcv)^2}
\]

\[
H_6 = e^{-0.5 (-2.5406869701733 - 0.000939220636382 RPM - 0.00001096222564003 Qcv)^2}
\]

\[
H_7 = e^{-0.5 (-14.04443363339 - 0.00416558912201866 RPM - 0.0000036286364065 Qcv)^2}
\]

(18)

**Figure 3.** Comparison between the experimental and model predicted values.
Figure 4. ANN optimization results by SAS-JMP© with weights of $\alpha = 30$ and $\beta = 1$.
ANN: artificial neural network.

Figure 5. Layer 1 data fitted wire mesh.
Consequently, the equation for the prediction for HC levels may be given by the following equation:

\[
HC = 370.47166484 - 3450.06956218 \, H_1 \\
- 903.827374055 \, H_2 - 19934.9798967 \, H_3 \\
+ 4280.49719774 \, H_4 + 19831.426155 \, H_5 \\
+ 16040.5406387 \, H_6 + 21824.232364 \, H_7
\]  

(19)

Furthermore, the accuracy of the ANN generated models fitting the experimental data was studied and accuracies of 97% for CO and 96% for HC were found to be adequate for these predictive models, and Figure 3 depicts the results of the validation process. This verification process increased the confidence in the ANN results and the generated predictive models were utilized to find the global optimum values for engine speed (RPM) and Q_{cv} for least emissions as prescribed by the optimization cost function.
The results of the ANN optimization process are as shown in Figure 4. The optimum values estimated by the ANN optimization were found to be \(<1119, 41,323>\), which results in an optimized operation with a minimum cost function value at arbitrarly chosen weights of \(\alpha = 30\) and \(\beta = 1\).

**MLO**

Using experimental data and equations (3)–(7) one can construct meshed patches of the two layers’ surfaces as depicted in Figures 5 and 6. While Figures 7 and 8 illustrate a high order single polynomial fitted surfaces using Matlab built-in functions. It is clear that the meshed patches have a better representation of the experimental data.

Once these patched surfaces are represented one can utilize the powerful search for optimum points using PSO method. The PSO will search the two layers’ spaces simultaneously for the minimum levels of CO and HC that satisfy the cost function depicted by equation (15). The PSO search is restricted to the ranges of the RPM and the Calorific value parameters. The search reveals that the point \(<1000, 41,463>\) results in an optimized operation with a cost function value of \(<33.1>\). Finally, while searching normalized values of CO and HC must be used or the cost function weights must be selected carefully. In this example, we have used \(\alpha = 30\) and \(\beta = 1\).

Figure 9 shows the PSO iterative search within the two layers’ surface spaces. A set of 100 Swarm Particles was used in the search. The speed of convergence is evident from the curve.

**Conclusions**

A technique that may be used to optimize experimental gathered data is presented, which could be used in resolving major challenges for scientists and engineers in finding optimum points. Systematic construction of the MLO scheme is derived and presented. Patched surfaces are used to capture the behavior of a collected experimental data. The overall surface is constructed from the patched pieces and a regional set of formulae are identified, which will be used for iterative evaluations of a cost function.

PSO algorithm was presented to locate a global optimum that fits a cost function within the constructed surfaces. The PSO algorithm searches the constructed multilayers
solution spaces simultaneously and optimize through a user predefined cost function.

The novel technique was successfully applied over two sets of experimental data. SP-Lines were used to form 3D patches surfaces from the experimental data in multi-layer fashion and incorporated a multi-layer search using PSO. The proposed technique was illustrated and verified and showed its effectiveness using ANN.

Finally, this proposed novel technique for constructing experimental data surfaces and performing a search for an optimum may be used for complex data structures that possess lots of layers of data (i.e. multi-variable and multi-dimensional).

Declaration of conflicting interests
The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding
The author(s) received no financial support for the research, authorship and/or publication of this article.

ORCID iD
M Abdalla https://orcid.org/0000-0003-1956-8897

References
1. Oehlert GW. A first course in design and analysis of experiments. Mountain View, CA: University of Minnesota, Creative Commons, 2010.
2. Balling RJ, Hedengren JD and Parkinson AR. Optimization methods for engineering design applications and theory. Provo, UT: Brigham Young University, 2013.
3. Seltman HJ. Experimental design and analysis. Pittsburgh, Pennsylvania: Carnegie Mellon University, 2018.
4. Najafi G, Ghobadian B, Yusaf T, et al. Optimization of performance and exhaust emission parameters of a SI (spark ignition) engine with gasoline–ethanol blended fuels using response surface methodology. Energy 2015; 90(P2): 1815–1829.
5. Karimi F, Rafiee S, Taheri-Garavand A, et al. Optimization of an air drying process for Artemisia absinthium leaves using response surface and artificial neural network models. J Taiwan Inst Chem Eng 2012; 29: 43.
6. Jabbr AI, Vaz WS, Khairallah HA, et al. Multi-objective optimization of operating parameters for hydrogen-fueled spark-ignition engines. Int J Hydrogen Energy 2016; 41: 18291–18299.
7. Wong PK, Wong KL, Vong CM, et al. Modeling and optimization of biodiesel engine performance using kernel-based extreme learning machine and cuckoo search. Renewable Energy 2015; 74: 640–647.
8. Jain R, Kasturi R and Schunck BG. Machine vision. New York City: McGraw-Hill, Inc., 1995.
9. Birkhoff G and Boor CD. Piecise polynomial interpolation and approximation. In: Proc. General Motors Symposium, New York, 1965.
10. Schoenberg IJ. Contributions to the problem of approximation of equidistant data by analytic functions: part A.—On the problem of smoothing or graduation. A first class of analytic approximation formulae. Q Appl Math 1946; 4: 45–99.
11. Schoenberg IJ. Contributions to the problem of approximation of equidistant data by analytic functions: part B.—On the problem of osculatory interpolation. A second class of analytic approximation formulae. Q Appl Math 1946; 4: 112–141.
12. Davis J. B-splines and geometric design. SIAM News 1997; 29.
13. Barata JCA and Hussein MS. The moore-penrose pseudoinverse. A tutorial review of the theory. Baltimore, MD: John Hopkins University Press, 2013.
14. MacAusland R. The moore-penrose inverse and least squares. Mountain View, CA: University of Puget Sound, Creative Commons, 2014.
15. Kennedy and Eberhart RC. Particle swarm optimization. In: IEEE Proceedings of International Conference on Neural Networks IV, Picataway 1995.
16. Eberhart RC and Shi Y. Evolving artificial neural networks. In: Proceedings of International Conference of Neural Networks and Brain, Beijing, China, 1998.
17. Hassan R, Cohanim B, Weck Od, et al. A copmarison of particle swarm optimization and the genetic algorithm. Am Inst Aeronaut Astronautics 2004; 1: 1–13.
18. Trojanowski K and Michalewicz Z. Searching for optima in Non-stationary environments. In: The Congress on Evolutionar, IEEE.
19. Robinsons J and Rahmat-Samii Y. Particle swarm optimization in electromagnetics. IEEE Trans Antennas Propag. 2004; 52(2): 397–407.
20. van den Bergh F and Engelbrecht A. A convergence proof for the particle swarm optimizer. Fundam Inform 2010; 105: 341–374.
21. Abu-Zaied M, Badran O and Yamin J. Effect of methanol addition on the performance of spark ignition engines. Energy & fuels. Energy Fuels 2004; 18: 312—3117.
22. Yamin J, Abu Zaied M and Badran O. Comparative performance of spark ignition engine using blends of various methanol percentages with low octane number gasoline. Int J Environ Pollut 2005; 23: 336–344.
23. Kapusuz M, Ozcan H and Yamin JA. Research of performance on a spark ignition engine fueled by alcohol–gasoline blends using artificial neural networks. Appl Therm Eng 2015; 91: 525–534.