Research Article

Quest for an intelligent convergence solution for the well-known David, Fletcher and Powell quadratic function using supervised models

Volume 2 Issue 1 - 2018

AA Ojugo,[1] IP Okobah[2]

[1]Department of Mathematics/Computer, Federal University of Petroleum Resources Effurun, Nigeria

[2]Department of Computer Science Education, Federal College of Education Technical, Nigeria

Correspondence: AA Ojugo, Department of Mathematics/ Computer, Federal University of Petroleum Resources Effurun, Nigeria; Tel +2348120800233, Email arnoldojugo@gmail.com

Received: December 23, 2017 | **Published:** February 01, 2018

Abstract

Optimization tasks aims to resolve complex events by exploiting historic data as well as explores task knowledge and symbolic reasoning to yield new heuristic and paradigm, whose outcomes is the underlying probability of all possible feats in an optimal solution. The outcome consist of input constraints/parameters to be satisfied; While, guaranteeing an explicit reasoning structure that conveys data about the task via an algorithm that assigns as output, a set of variables that satisfies these constraints in its bid to prune off a huge portion of the search space. Our study seeks to optimize the quadratic functions via David Fletcher Powell method (and its use as pre-processor to train selected supervised models) to find all converged points. As dataset, we use the trial solution of the quadratic equation written as sum of 2-parts: (a) satisfy initial condition for the unconstrained optimization using DFP and supervised models to resolve quadratic function; and (b) uses DFP as pre-processor with adjustable parameters to train the supervised models. Results indicate QDA outperforms KNN, LDA and DFP in all cases to yield an approximate solution of the quadratic function, and also show the models converge closer to an analytic solution. This is easily extended to solve a wide range of other problems.

Introduction

Optimization models tend to resolve complex task by exploiting historic data whose underlying probability feats yield optimal solution. It achieves this also, by exploring domain knowledge and symbolic reasoning expressed as mathematical model to yield a heuristic method that is both tolerant to noise, accepts partial truth, uncertainty and ambiguities at its input–while, exhibiting a great level of robustness, flexibility, and continuous adaptation (as model feats) in its bid to resolves a constraints within the system. Thus, mimics agents in search of optimal solution (food and survival) in a domain space, and is mostly inspired by behavioral pattern and natural laws of biological evolution. It yields output feats with uncontrolled parameters as input (not explicitly present from outset); But, modeled therein via boundary values in domain space confined to real parameters and derives via experience, ability to recognize behavioral feats from observed data, and can suggest optimal solution of high quality that is void of over-fitting, irrespective of modification made to the model via other approximations with multiple agents. These constantly affect the quality of the solution. Also, many of such heuristics are combined to create hybrid(s) that seeks to explore the structural differences in the statistical-method(s) used, and help resolve the implications of the conflict constrained on the model by such a multi-agent populated system–as agents can often create their own behavioral rules based on historic dataset.[1]

Chaotic and dynamic phenomena and events abound almost in every sphere of our daily endeavors. These must be resolved via the consequent creation of systems and models that mimic such behaviors. Thus, nonlinear systems are today-modeled as mathematical equations (models) to help resolve nonlinear, dynamic and complex feats in tasks such as in control systems and other applications. These systems are often modeled as quadratic functions, and there exists various methods in use to resolve quadratic cum differential equations. Optimal solution for such nonlinear, quadratic equations is still a challenge task.[2-4] Studies have shown that parallel processor computers in order to solve a first order differential equation will use unsupervised model (neural network model) as a factor of speed;[5] while[6] went further to represent a new method to solve first-order linear ordinary and partial equations using ANN, as was further buttressed by[6,7] provided a hybrid ANNPSO intelligence model to solve Wessinger equation (though the model could not satisfy the initial and boundary conditions). It was improved by;[5] while[6] furthered to satisfy the solution for initial and boundary condition problems. Search methods aim to maximize/ minimize constraint satisfaction problem objective functions and yield a feasible, optimal (closest to best) solution. CSPs are dynamic, nonlinear and complex-making the search for its solution cumbersome and inexplicable to resolve. The study presents comparative stochastic model for solving quadratic equation using DFP as preprocessor to seek optimality and convergence property.

Mathematical Optimization

Mathematical optimization is a selection of the best element from a set of available alternatives. It thus, aims either at a maximization or minimization task of real function that symmetrically choose inputs from an allowed set of variables, to compute the function's output by finding the best available values from a set of alternatives via the objective function in a given domain[8-10] and[11] notes:

Definition 1

A continuous optimization defined as a pair (S, f). S is set of possible solutions: $S = R^N$ and N is number of controllable parameters and R is the real number line. Thus, f is a multi-objective function ($f: S \rightarrow R$) to be optimized-where a solution vector X is as limited between lower and upper bounds ($x_{lb} \le x \le x_{ub}$). For maximization task (search for a solution greater than or equal to all other solutions), and minimization task (search for solution that is smaller than or equal to the all other solutions).The set of maximal and minimal solution $S_{max} \subseteq$ S of a function f: S → R defined as:

$$X_{max} \in S_{max} \Leftrightarrow \forall X \in S : f\left(X_{max}\right) \le f\left(X\right) \to (1)$$

$$X_{min} \in S_{min} \Leftrightarrow \forall X \in S : f\left(X_{min}\right) \le f\left(X\right) \to (2)$$

Materials and Methodology

The quadratic problem formulation for DFP

We aim to minimize

$$f\left(x_1 x_2\right) = 5x_1^2 + 5x_1^2, x_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \textit{ as start points} \text{ and}$$

Dai (2001) notes that a point x* ε A is a global minimum of f_o (x) if:

$$f\left(x^0\right) \le f\left(x\right) \textit{ for all } x \,\varepsilon\, A(3)$$

An ideal optimization allows its objective function to have a unique minimize. Thus, the function in Eq. 3 is a global minimize. Conversely, a point x* ε A is a local minimize if for any $\varepsilon > 0$, f(x*) ≤ f(x) and for any $x \varepsilon A$, $\left\|x - x^*\right\| \le \varepsilon$.

Line search method / algorithm

[12,13]notes that numeric method algorithm for unconstrained optimization (as grouped into line search and trust region) aims to minimize the nonlinear function f(x) as in Eq. 4. Thus, an iterative, continuous f(x) with initial point x_1, its kth iteration (new point x_{k+1}) is computed as thus:[14]

$$Y = \min_{x \,\varepsilon\, R^N} f\left(x\right) \to (4)$$

The goal of this convergence analysis is to study the sequence $\{x_k\}$ feats generated and compare its difference between to the convergence performances of other methods[15,16] such as DFP and with DFP as a pre-processor. The sequence x_k generated converges to a point x* if:

$$Y = \lim_{k \to \infty} \left\|x_k - x^*\right\| = 0 \to (5)$$

With the solution for x* as in Eq. 5 not available, a possible replacement is Eq. 6, which unfortunately, also does not guarantee the convergence of x_k will then yield Eq. 6 as given by[17,18] as thus:

$$Y = \lim_{k \to \infty} \left\|x_k - x_{k-1}\right\| = 0 \to (6)$$

But, global convergence for unconstrained optimization aims to prove that Eq. 7 ensures x_k is close to the set of stationery points where ∇(x) = 0, or Eq. 8, which ensures that a least subsequence of x_k is close to the set of the stationery points.[19,20]

$$\lim_{k \to \infty} \left\|g_k\right\| = 0 \to (7)$$

$$\lim_{k \to \infty} \operatorname{InF} \left\|g_k\right\| = 0 \to (8)$$

But $g_k = g\left(x_k\right) = \nabla f\left(x_k\right)$. Local convergence aims to address the convergence speed of the sequence generated by the algorithm. In studying this, we assume sequence x_k converges to a local minimize x*, so that the second order sufficient condition and the Newton method, can converge very slowly.[13-21]

Quasi-Newton method

[22,23]has Newton's method as basis for Quasi-Newton, and the most widely used for solving nonlinear equations and unconstrained optimization. If for a smooth function f(x) and a positive hessian, then second order Taylor's expansion yields thus:

$$f\left(x+p\right) \cong f\left(x\right) + p^T \nabla_f\left(x\right) + \frac{1}{2} p^T \nabla_f\left(x\right)$$

$$\textit{Differentiating yields } \frac{\partial f\left(x+p\right)}{\partial p} \cong \nabla_f\left(x\right) + \nabla^2{}_f\left(x\right) = 0$$

$$\textit{Thus}, \nabla^2{}_f\left(x\right) = -\nabla_f\left(x\right) \to (9)$$

As long as $\nabla^2_f(x)$ is a positive definite, $\nabla^2_f(x)$ p is Newton direction, and the next approximation is:

$$x^{k+1} = x^k - \frac{\nabla f\left(x\right)}{\nabla^2 f\left(x\right)} \to (10)$$

Newton direction has proven to be more expensive than Steepest Descent direction. We must compute Hessian matrix and invert it (not applicable with Quasi Newton). Its merits are: (a) convergent rate for Newton method is quadratic, and thus, there is a lot to gain in finding its direction, (b) they form a good starting point if $f''(x)$ is positive definite, and (c) they are simple and easy to implement. However, its demerits is: (a) they are not globally convergent for many tasks, (b) it may diverge if starting point approximation is far from the solution, (c) it fails if hessian matrix is not inverted, and (d) requires analytic second order derivatives of f. Also, the method is seen as an approximation of the Newton Raphson method.[23,24]

$$x^{k+1} = x^k - \frac{f\left(x\right)}{f'\left(x\right)} \to (11)$$

Consider the Quasi Newton method behaviour from Broyden's class of unconstrained optimization problem given by: *min* $\{f(x): x \varepsilon R^N\}$. The class consists of iterations of the form:

$$x_{k+1} \leftarrow x_k + \alpha_k \rho_k \textit{ where } \rho_k = -\beta_k^{-1} g_k \to 12$$

g_k is gradient of f at x_k, Hessian approximation β_k is updated by[25,26] and α_k is step size. So:

$$\beta_{k+1} = \beta_k - \frac{\beta_k s_k s_k^T \beta_k}{s_k^T \beta_k s_k} + \frac{y_k y_k^T}{y_k^T s_k} + \varnothing\left(s_k^T \beta_k s_k\right) v_k v_k^T \rightarrow (13)$$

Yields 2-update formulae: For $\varnothing = 0$, yields Broyden, Fletcher, Goldfarb and Shanno (BFGS) method; while, $\varnothing = 1$, yields the Davidson, Fletcher and Powell method.[27,28] Notes that with exact line search, all class members yield same iterates and their performance varies markedly. We assume step-size α_k is chosen by an inexact line search satisfying two conditions as thus:

$$f\left(x_k + \alpha_k \rho_k\right) \le f\left(x_k\right) + \sigma \alpha_k g_k^T \rho_k \rightarrow (14)$$

$$g\left(x_k + \alpha_k \rho_k\right)^T \rho_k \ge \beta g_k^T \rho_k \rightarrow (15)$$

Note several important results about this class of methods are that they do not require an exact line search. If all assumptions hold that: x* is the minimize of *f*, the hessian matrix H is positive definite, $\alpha_k = 1$ in DFP/BGFS as Eq. 16 is true and x_k converges to x* Q-super linearly, and its sum will be finite, if $\|x_1 - x^*\|$ *and* $\|\beta_1 - H\|$ are sufficiently small and step size $\alpha_k = 1$ will satisfy the line search conditions of Eq. 14 and Eq. 15.[28,29]

$$\sum_{k=0}^{\infty} \left\| x_{k+1} - x^* \right\| < \infty \rightarrow (16)$$

DFP as preprocessor

DFP optimizes (with/without an exact line search). Here, DFP computes solution of a given quadratic functions (as problem domain) via stochastic hybrid models, define convergence properties and state its influence on such convergence properties. The algorithm is:[1-3]

1. At iteration *i*, DO:
2. Choose $A_i \cong H^{-1}$ (inverse of hessian matrix at *i*).
3. If x^i is optimal, stop; Else obtain the search direction p*i* by solving: $\rho^i = -A i \nabla f i(x)$
4. Minimize f(x) in direction of p*i* via

$$\min_{\alpha} f\left(x^i + \alpha \rho^i\right) \textit{ to find } \alpha$$

5. Define: $\rho^i = x^{i+1} - x^i = \Delta x^i \left(\textit{change in } x\right)$

$$y^i = \nabla f_{i+1}(x) - \nabla f_i(x) = \nabla g_i \left(\textit{change in gradient}\right)$$

6. Compute:

$$\beta_i = \frac{\left(\rho^i\right)^T \left(\rho^i\right)}{\left(\rho^i\right)^T \left(y^i\right)}$$

$$C_i = \frac{-A_i y^i \left(A_i y^i\right)^T}{A_i y^i \left(y^i\right)^T}$$

7. Update the hessian matrix A as thus $A_{i+1} = Ai + Bi + Ci$

Set $i = i + 1$ and return to step 2

Intelligent proposed model

We seek to compare resolution of above quadratic function via the supervised models below using unsupervised model (HMM as benchmark) to measure comparative performances.

Linear Discriminate Analysis (LDA)

LDA is a simple and effective supervised classification method with wide range of applications. Its basic theory is to classify compounds (rules) dividing n-dimensional descriptor space into two regions separated by a hyper-plane that is defined by linear discriminate function. Discriminant analysis generally transforms classification tasks into functions that partitions data into classes; thus, reducing the problem to an identification of a function. The focus of discriminate analysis is to determine this functional form (assumed to be linear) and estimate its coefficients. LDA was first introduced in 1936 by Ronald Aylmer Fisher and his LDA function works by finding the mean of a set of attributes for each class, and using the mean of these means as boundary. The function achieves this by projecting attribute points onto the vector that maximally separates their class means and minimizes their within-class variance as expressed in Eq. 17 as follows:

$$LDA = X'S^{-1}\left(X_2 - X_1\right) - \frac{1}{2}\left(X_2 + X_1\right)' S^{-1}\left(X_2 - X_1\right) > c \rightarrow (17)$$

where X is vector of the observed values, X_i (i = 1, 2…) is the mean of values for each group, *S* is sample covariance matrix of all variables, and c is cost function. If the misclassification cost of each group is considered equal, then c = 0. A member is classified into one group if the result of the equation is greater than c (or = 0), and into the other if it less than c (or = 0). A result that equals c (set to 0) indicates such a sample cannot be classified into either class, based on the features used by the analysis. LDA function distinguishes between two classes-if a data set has more than two classes, the process must be broken down into multiple two-class problems. The LDA function is found for each class versus all samples that were not of that class (one-versus-all). Final class membership for each sample is determined by LDA function that produced the highest value and is optimal when variables are normally distributed with equal covariance matrices. In this case, the LDA function is in same direction as Bayes optimal classifier,[30-33] and it performs well on moderate sample sizes in comparison to more complex method. Its mathematical function is simple and requires nothing more complicated than matrix arithmetic. The assumption of linearity in the class boundary, however, limits the scope of application for linear discriminate analysis. Real-world data frequently cannot be separated by linear boundary. When boundaries are nonlinear, the performance of the linear discriminate may be inferior to other classification methods. Thus, to curb this-we adopt a decimal encoding of the data to give us a semblance of linear, continuous boundaries.

Quadratic Discriminant Analysis (QDA)

QDA is another distance-based classifier by, which is very similar to and more of an extension of LDA. Both discriminate functions assume that values of each attribute in each class are normally distributed, however, the discriminate score between each sample and each class is calculated using the sample variance-covariance matrix of each class separately rather than the overall pooled matrix and so is a method that takes into account the different variance of each class. While, LDA assumes that the covariance matrices of the groups are equal; QDA makes no assumption. When the covariance matrices are not equal, the boundary between the classes will be a hyper-conic and in theory, the use of quadratic discriminate analysis will result in better discrimination and classification rates. However, due to the increased number of additional parameters to be estimated, it is possible that the classification by QDA is worse than that of linear discriminate analysis. The QDA is found by evaluating the Eq. 18:

$$QDA = X'\left(S_1^{-1} - S_2^{-1}\right)X + 2\left(Y_2' S_1^{-1} - Y_1' S_2^{-1}\right)X - \left[Y_2' S_1^{-1} Y_2 - Y_1' S_2^{-1} Y_1 + Ln\left(\frac{|S_2|}{|S_1|}\right)\right] > c \rightarrow (18)$$

The same conditions apply to the nature of c as well as the classification, in the case that the result is equal to c or zero. As with LDA, the QDA distinguishes between two classes. For multiple class data sets, this was handled the same as for linear discriminate analysis.[34,35] Size of differences in variances determines how much better QDA performs better than LDA. For large variance differences, QDA excels when compared to LDA. Additionally, of the two, only QDA can be used when population means are equal. QDA is more broadly applicable than the LDA; But, less resilient in non-optimal conditions. The quadratic discriminate can behave worse than the linear discriminate for small sample sizes. Additionally, data that is not normally distributed results in a poorer performance by the quadratic discriminate, when compared to the linear discriminate. found the performance of the quadratic discriminate function to be more sensitive to the dimensions of the data than the linear discriminate, improving as the number of attributes increases to a certain optimal number, and then rapidly declining. Linear and nonlinear discriminate functions are the most widely used classification methods. This broad acceptance is due to their ease of use and the wide availability of tools. Both, however, assume the form of the class boundary is known and fits a specific shape. This shape is assumed to be smooth and described by a known function. These assumptions may fail in many cases. In order to perform classification for a wider range of real-world data, a method must be able to describe boundaries of unknown, and possibly discontinuous, shapes.

K-Nearest Neighborhood (KNN)

The *K*-nearest neighbor (KNN) model is a well-known supervised learning algorithm for pattern recognition that first introduced by, and is still one of the most popular nonparametric models for classification problems. *K*-nearest neighbor assumes that observations, which are close together, are likely to have the same classification. The probability that a point x belongs to a class can be estimated by the proportion of training points in a specified neighborhood of x that belong to that class. The point may either be classified by majority vote or by a similarity degree sum of the specified number (*k*) of nearest points. In majority voting, the number of points in the neighborhood belonging to each class is counted, and the class to which the highest proportion of points belongs is the most likely classification of *x*. The similarity degree sum calculates a similarity score for each class based on the *K*-nearest points and classifies *x* into the class with the highest similarity score. Its lower sensitivity to outliers allows majority voting to be commonly used other than the similarity degree sum. We use majority voting for the data sets to determine which points belongs to neighborhood so that distances from *x* to all points in the training set must be calculated. Any distance function that specifies which of two points is closer to the sample point could be employed. The most common distance metric used in *K*-nearest neighbor is the Euclidean distance. The Euclidean distance between each test point *ft* and training set point *fs*, each with *n* attributes, is calculated via Eq. 19:

$$d = \left[(f_{t1} - f_{s1})^2 + (f_{t2} - f_{s2})^2 \ldots + (f_{tn} - f_{sn})^2\right]^{\frac{1}{2}} \rightarrow (19)$$

In general the following steps are performed for the *K*-nearest neighbor model: (a) chosen of *k* value, (b) distance calculation, (c) distance sort in ascending order, (d) finding *k* class values, (e) finding dominant class.

A challenge in using *K*nn is to determine optimal size of *k*, which acts as smoothing parameter. A small *k* is not sufficient to accurately estimate the population proportions around the test point. A larger *k* will result in less variance in probability estimates (but for risk of introducing more bias). *K* should be large enough to minimize probability of a non-Bayes decision, and small enough that all points included, gives an accurate estimate of the true class[33] found optimal value of k to depend on sample size and covariance structures in each population and on the proportions for each population in the total sample. For cases where differences in covariance matrices and difference between sample proportions are both small or both large, it is found that optimal *k* is $N^{3/8}$ (*N* is number of samples in the training set). If and when there is a large difference between covariance matrices, and a small difference between sample proportions (or vice-versa), the optimal value *k* is determined by $N^{2/}$.[33] This model presents several merits in that: (a) its mathematical simplicity does not prevent it from achieving classification results as good as (or better than) other more complex pattern recognition techniques, (b) it is free of statistical assumptions, (c) its effectiveness does not depend on the space distribution of classes, and (d) when the boundaries between classes are not hyper-linear or hyper-conic, K-nearest neighbor performs better than LDA.

But, found LDA performs slightly better than K-nearest neighbor if the population covariance matrices are equal, a condition that suggests linear boundary. As differences in covariance matrices increases, *k*-nearest neighbor performs increasingly better than LDA and QDA function. However, despite these merits, the demerits of *k*-nearest neighbor model include that it does not work well if large differences are present in samples in each class. K-nearest neighbor provides poor data about the structure of its classes, and relative importance of variables in classification. Also, it does not allow graphical representation of the results, and in case of large number of samples, computation become excessively slowly. In addition, K-nearest model requires more memory and processing requirements than other methods. All prototypes in training set must be stored in memory and used to calculate Euclidean distance from every test sample. The computational complexity grows exponentially as the number of prototypes increases.

Result Findings and Discussion

Model performance and effectiveness measure

Performance[34,35] of the models, are evaluated via computed MSE, MRE, MAE and COE as: To measure their effectiveness and classification accuracy, we adopt the misclassification rate of each model as well as its corresponding improvement percentages of the proposed model in comparison with those of other classification models for the DFP data (in both training and test data) summarized in (Table 1). The equations for its improvement percentage when trained with DFP is as below:

$$Improvement\ Percentage = \frac{MR(A)-MR(B)}{MR(A)} x100 \quad (20)$$

Results show from tables 2 shows our supervised models in LDA, QDA and KNN showed an improvement rate of 45.8%, 64.2% and 46.1% respectively. Also, it is observed that though the *KNN* scores were sensitive to relative magnitude of different attributes, all attributes are scaled by their z-scores before using *KNN* model. This is in tandem with and results are supported by,[36-40] (Table 2). The number of data-points in quadratic function has great influence on model performance-such that, if the data points are too small, most supervised models may not achieve its accuracy. If it is too many, it can result also in overtraining as well as over-parameterization.[41-45]

Table 1 Model convergence performance evaluation

Model	MSE	MRE	MAE	COE	Converge accuracy %
DFP	0.56	0.43	0.49	0.49	30.02
LDA	0.87	0.69	0.65	0.581	23.54
KNN	0.67	0.55	0.56	0.481	28.6
QDA	0.46	0.37	0.46	0.818	39.2

Table 2 Improvement percentage with Dfp-preprocessor

Improvement %		
Model	**Training data**	**Testing data**
LDA	41.16	45.83
KNN	41.79	46.09

The rationale for the choice of models includes

a. Unsupervised Model has great capability in learning to approximate function-making them flexible and mostly, universal estimator cum approximator. Their adaptive feat is a huge merit in modeling dynamic, changing states. However, the problem is that of encoding, fitness function selection, parameterization and training/retraining model to suit the purpose for which is seeks to understudy the underlying probability feats. But, when these are taken care of-unsupervised models perform excellently well.[45-50]

b. Supervised Model such as LDA, QDA and SVM (support vector machines) have their demerits as in section II.

c. DFP: Many well-behaved continuous functions have been developed which rely on using data about the gradient of the function to guide the direction of search. Derivatives cannot be computed if function is discontinuous. Such methods are generally referred to as *hill-climbing*. They perform well on functions with only one peak (*unimodal* functions). But on functions with multi-objective and multi-peaks, (multimodal functions), they suffer from fact that the first peak found is climbed, and this may not be the highest peak. Having reached the top of a local maximum, no further progress can be made.[51-55]

Related study

The fuzzy trained neural network in evaluating the well-known Wessinger's quadratic function as a constraints satisfaction problem. Their results show the model converged closer to data points in the range of their analytical solution adopted the memetic algorithm and SA to evaluate quadratic function. His trained his hybrids using DFP as a pre-processor to yield approximate solutions to the quadratic function. A trial solution of the quadratic equation is also written as sum of two parts: (a) to satisfy initial condition for unconstrained optimization using DFP and hybrids as separate methods, and (b) apply DFP as a pre-processor with adjustable parameters of for ANN-TLRN hybrid.[56-61]

Conclusion

Study consists of 4-phases: (a) train models using quadratic function, (b) delete outliers in data by resolving the quadratic function and finding underlying probability on convergence ability, (c) calculate membership probability of output points in each class, and (d) assign output to appropriate class by largest probability. We adopt four (4) known intelligent and statistical classification models: LDA, QDA, *k*nn and PHMM. QDA outperforms LDA, KNN and DFP classic model. The unsupervised models do not assume the shape of the partition, unlike the linear and quadratic discriminate analysis. In contrast to *K*-nearest neighbor model, the proposed model does not require storage of training data. Once the model has been trained, it performs much faster than *K*-nearest neighbor does, because it does not need to iterate through individual training samples. The proposed model does not require experimentation and final selection of a kernel function and a penalty parameter as is required by the support vector machines. Our proposed model solely relies on a training process in order to identify the final classifier model. Finally, the unsupervised models does not need large amount of data in order to yield accurate results.

Acknowledgements

None.

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. Ojugo AA. *A comparative stochastic model solution on the convergence problem of quadratic functions*. Unpublished Thesis, Mathematics and Computer Science Department, Nigeria; 2015.
2. Fletcher R. *Practical methods of optimization*. 2nd edn. John Wiley & Sons, Chichester, UK; 1987. p. 456.
3. Fletcher R, Reeves C. Function minimization by conjugate gradients. *Computational Journal*. 1964;7(2):149–154.
4. Friedlander A, Martinez JM, Molina B, et al. Gradient method with retards and generalization. *SIAM J of Numerical analysis*. 1999;36:275–289.

5. Ghalambaz M, Noghrehabadi AR, Behrang MA, et al. Hybrid gravitational search neural network method to solve the well known Wessinger's equation. *World Academy of Science and Technology*. 2011;49:803–807.
6. Malek A, Beidokhti RS. Numerical solution for high order differential equations using hybrid neural network-Optimization method. *Applied Mathematics and Computation*. 2006;183:260–271.
7. Khan J, Zahoor R, Qureshi IR. Swarm intelligence for problem of non-linear ordinary differential equations and its application to well known Wessinger's equation. *European J Sci Research*. 2009;34(4):514–525.
8. Ojugo AA, Eboka AO, Okonta EO, et al. Genetic algorithm rule-based intrusion detection system. *Journal of Emerging Trends in Computing and Information Systems*. 2012;3(8):1182–1194.
9. Armijo S. Minimization function having Lipschitz continuous partial derivatives. *Pacific Journal of Mathematics*. 1966;16:1–3.
10. Ojugo AA. *An artificial neural networks gravitational search model for rainfall runoff simulation and modeling*. unpublished PhD, Computer Sci Department, Nigeria; 2012.
11. Allenotor D. A stochastic solution on convergence properties of quadratic functions, Computing Information Systems. *Development Informatics and Allied Research Journal*. 2016;7(2):9–20.
12. Dai YH, Yuan JY, Yuan Y. Modified two-point step size gradient methods for unconstrained optimization. *Computational Optimization and Application*. 2002;22:103–109.
13. Polyak BT. The conjugate gradient method in extreme problems. *USSR Computation Mathematics: Mathematic Physics*. 1969;9:94–112.
14. Barzilar J, Borwein JW. Two point step size gradient methods. *IMA Journal of Numerical Analysis*. 1988;37:141–148.
15. Powell MJ. On the convergence of the variable algorithm D, *Winston Mathematics Application*. 1971. p. 21–38.
16. Powell MJ. Some global convergence properties of a variable metric algorithm for minimization without exact line search. Inc: Cottle RW & Lemke CE (Eds.), Nonlinear programming. *SIAM Proceedings of AMS*. 1976;9: 3–72.
17. Gottlieb D, Orszag SA. Numerical analysis of spectral methods: theory and applications. *CBMS-NSF Regional Conference Series in Applied Mathematics*. 1977;172:26.
18. Daniel JW. Conjugate gradient method for linear/nonlinear operator equations. *SIAM J Numeric Analysis*. 1967;4(1):10–26.
19. Dai YH, Yuan Y. A nonlinear conjugate gradient method with strong global convergence property. *SIAM Journal of Optimization*. 1999;10:177–182.
20. Dai YH, Yuan Y. Alternate minimization gradient method. *IMA Journal of Numerical Analysis*. 2003;23(3):377–393.
21. Polak E, Ribiere G. Note sur la convergence de directions conjugees. *Rev Francaise Informat Researche Operationelle*. 1969;3(16):35–43.
22. Dennis JE, Moore JJ. A characterization of superliner convergence and its application to Quasi Newton methods. *Mathematical Computation*. 1974;28:549–560.
23. Dennis JE, Moore JJ. Quasi Newton method: motivation and theory. *SIAM Rev*. 1977;19(1):46–89.
24. Liu Y, Storey C. Efficient generalized conjugate gradient algorithms. *J of Optimization Theory Application*. 1991;69(1):129–137.
25. Broyden CG. Quasi Newton method and their application to function approximation. *Mathematical Computation*. 1967;21:368–381.
26. Cauchy A. Method general pour la resolution de systems d'equations simultanees. *Computer Rendition of Science Paris*. 1987;25:46–89.
27. Dixon LCW. Variable metric algorithms necessary and sufficient conditions for identical behaviour on non-quadratic functions. *J of Optimization Theory and Application*. 1972;10(1):34–40.
28. Ritter K. Local and superliner convergence of a class of variable method. *Computing*. 1979;23:287–297.
29. Griewank A, Toini PH. Local convergence analysis of partitioned Quasi Newton updates. *Numerical Mathematics*. 1982;39(3):429–448.
30. Billings S, Lee K. Nonlinear Fisher discriminate analysis using a minimum squared error cost function and the orthogonal least squares algorithm. *Neural Networks*. 2002;15:262–270.
31. Fix E Hodges J. Discriminatory analysis-nonparametric discrimination: Consistency properties, Project No. 21-49-004, Report No. 4, USAF School of Aviation, Randolph Field, Texas, USA; 1951.
32. Chaovalitwongse W. On the time series k-nearest neighbor classification of abnormal brain activity. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*. 2007;37(6):1–3.
33. Enas G, Choi S. Choice of the smoothing parameter and efficiency of k-nearest neighbor. *Computers and Mathematics with Applications*. 1986;12(2):235–244.
34. Ojugo A, Yoro R. Computational intelligence in stochastic solution for Toroidal Queen. *Progress in Intel Comp App*. 2013;2(1):46–56.
35. Ojugo AA, Emudianughe J, Yoro RE, et al. Hybrid artificial neural network gravitational search algorithm for rainfall runoff modeling in Hydrology. *Progress Intel Comp App*. 2013;2(1):22–33.
36. Perez M, Marwala T. Stochastic optimization approaches for solving Sudoku. *IEEE Transaction on Evol Comp*. 2011;1:256–279.
37. Mandic D, Chambers J. *Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability.* In:John Wiley, Sons, editors. USA; 2001. p. 308.
38. Meade AJ, Fernandez AA. The numerical solution of linear ordinary differential equations by feed forward neural networks. *Mathematical and Computer Modeling*. 1994;19(12):1–25.
39. Ojugo AA. A comparative study of intelligent models in the forecast of rainfall runoff: a case of the Benin-Owena River Basin in Nigeria. *Technical-Report Series for the Academic Staff Union of Polytechnics.* 2012;5(4):12–18.
40. Ojugo AA, Allenotor D, Oyemade DA, et al. Immunization Model for Ebola Virus in Rural Sierra-Leone. *African Journal of Computing & ICTs*. 2015;8(1):1–10.
41. Lagaris IE, Likas A, Fotiadis DI. Artificial neural networks for solving ordinary and partial differential equation. *IEEE Transactions Neural Network*. 1998;9(5):987–1000.
42. Berardi V, Zhang GP. The effect of misclassification costs on neural network classifiers. *Decision Sciences*. 1999;30(3):659–668.
43. Calisir D, Dogantekin E. An automatic diabetes diagnosis system based on LDA-Wavelet Support Vector Machine Classifier. *Expert Systems with Applications*. 2011;38:311–8315.
44. Chakraborty R. *Soft computing and fuzzy logic*. Lecture notes, Nigeria; 2010. p. 1–37.
45. Dai YH. Alternate step gradient method, Report AMSS-2001-041. *Academy of Mathematics and Systems Science*. 2001. p. 32–56.
46. Denton JW, Hung MS. A comparison of nonlinear optimization methods for supervised learning in multilayer feed forward networks. *European J of Operation Research.* 1996;93:358–368.
47. Forsythe GE. On asymptotic directions of the s-dimension optimum gradient method. *Numerische Mathematik*. 1986;11:57–76.

48. Giles D, Draeseke R. Economic Model Based on Pattern recognition via fuzzy C-Means clustering algorithm. *Economics Dept Working Paper, EWP0101*. Canada; 2001. p. 1–50.

49. Hager W, Zhang H. A new conjugate gradient method wit guaranteed descent and an efficient line search. *SIAM J of Optimization*. 2003. p. 305–333.

50. Inan G, Elif D. Adaptive neuro-fuzzy inference system for classification of EEG using wavelet coefficient. *J Neurosci Methods*. 2005;148(2):113–121.

51. Lee H, Kang IS. Neural algorithms for solving differential equations. *Journal of Computational Physics*. 1990;91(1):110–131.

52. Nash J, Sutcliffe J. River flow forecasting with conceptual models. *J of Hydro Sci*. 1970;10(3):282–290.

53. Ojugo AA, Allenotor D, Oyemade DA, et al. Comparative stochastic study for credit-card fraud detection models. *African Journal of Computing & ICTs*. 2015;8(1):15–24.

54. Pham D, Karaboga D. Training Elman and Jordan networks for system identification using genetic algorithms. *Artificial Intelligence in Engineering*. 1999;13:107–117.

55. Koc E, Ghanbarzadeh A, Otri S. Optimization of weights of multi-layered perceptrons using bees algorithm. *Proceedings of Intelligent Manufacturing Systems*. Sakarya University, Nigeria; 2006. p. 38–46.

56. Pham DT, Liu X. *Artificial Neural Networks for Identification, Prediction and Control*. Springer Verlag, UK; 1995.

57. Plumb AP, Rowe RC, York P, et al. Optimization of the predictive ability of artificial neural network models. *European J of Pharmaceutical Sciences*. 2005;25(4–5):395–405.

58. Raydan M. On Barzilai and Borwein choice of stepsize for the gradient method. *IMA Journal Numeric Analysis*. 1993;13:321–326.

59. Reynolds R. An introduction to cultural algorithms. *IEEE Transaction on Evolutionary Programming*. 1994. p. 131–139.

60. Stachurski A. Superliner convergence of a class of variable method, *Mathematical Programming*. 1981;14:178–205.

61. Ursem R, Krink T, Jensen M, et al. Analysis and modeling of controls in dynamic systems. *IEEE Transaction on Evolutionary Computing*. 2002;6(4):378–338.