Adaptive Nonconvex Optimization for AI and Machine Learning

(Patent Pending)

Randal Allen
Lone Star Analysis
Orlando, FL
rallen@lone-star.com

ABSTRACT

This paper discusses a novel approach to nonconvex optimization which has broad-reaching applications, including those prevalent in artificial intelligence, neural nets, supervised (regression and classification) and unsupervised (clustering) machine learning, and quantum computing.

A system of methods which reaps the benefits of both grid search and random search, without their corresponding limitations, is uniquely combined with an exact method of multipliers to produce a novel approach to solving general nonconvex objective functions. At its core, independent random variables adapt themselves to produce a finer search for an extremum, according to user-defined precision specification. Because the system is gradient-free, the architecture allows for logic gates with implications for machine learning and quantum computing. Furthermore, Monte Carlo methods increase confidence in locating the global extremum facilitating verification and validation of trustable artificial intelligence.

Finally, as an example, the regression form of supervised learning (replacing neural net back-propagation with this novel nonconvex optimization approach) is applied to determination of an aerodynamic rolling moment coefficient. The results of the new approach were favorably compared to that obtained by the traditional system identification method but were achieved using only 40% of the baseline training data and an order of magnitude less iterations.

ABOUT THE AUTHOR

Randal Allen is the Chief Scientist of Lone Star Analysis. He is responsible for applied research and technology development across a wide range of MS&A disciplines. He holds a CMSP with NTSA. He is co-author of the textbook, "Simulation of Dynamic Systems with MATLAB and Simulink." He is an Associate Fellow of AIAA. He holds a Ph.D. in Mechanical Engineering (University of Central Florida), an Engineer's Degree in Aeronautical and Astronautical Engineering (Stanford University), an M.S. in Applied Mathematics and a B.S. in Engineering Physics (University of Illinois, Urbana-Champaign). He serves as an Adjunct Professor in the MAE department at UCF.

Adaptive Nonconvex Optimization for AI and Machine Learning

Randal Allen
Lone Star Analysis
Orlando, FL
rallen@lone-star.com

MOTIVATION

There are several sources motivating this innovative technology. An MIT Technology Review on Intelligent Machines states, the U.S. Military wants its autonomous machines to explain themselves. The latest machine-learning techniques are essentially black boxes. DARPA is funding efforts to open them up (Knight, W., 2017). DARPA also announced its Artificial Intelligence Exploration (AIE) program, a key component of the agency's broader artificial intelligence (AI) investment strategy aimed at ensuring the United States maintains an advantage in this critical and rapidly accelerating technology area (DARPA, 2018). In February, President Trump enacted the "American AI Initiative," identifying AI as a priority for government research and development. The next day, the DoD released its own AI strategy, positioning its newly created Joint Artificial Intelligence Center (JAIC) at the forefront of its efforts (Lamberth, M., 2019).

BACKGROUND

Perhaps the seminal optimization problem is rooted in the method of least squares, credited to Gauss in the late 18th century and published by Legendre in the early 19th century. This problem seeks to minimize the sum of the squares of the residual error between a set of observed data and a line fitted to them.

In the 20th century, methods have been developed to solve problems applicable to operations research and management sciences including game theory, decision analysis, queueing theory, inventory theory, transportation, and networking. These problems are predominantly concerned with seeking the *unique* extremum of systems which exists due to what is known as convexity. Figure 1 shows an example of a convex function where there is a unique extremum, in this case a minimum, at the origin. Optimization of convex systems is not difficult when using modern computing equipment. In fact, linear optimization was used in the 1940's before general purpose digital computers existed. Today, digital systems optimize these relatively simple problems easily.

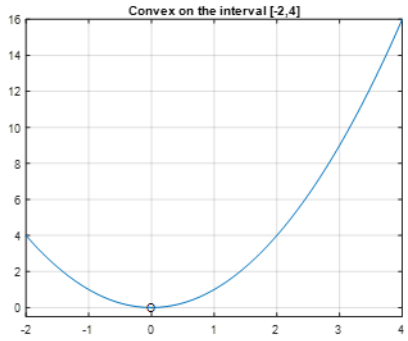

Figure 1: Convexity – a unique extremum [0, 0] exists on the interval [-2, 4]

However, most useful systems are not linear, not convex, or worse, may contain discontinuities. Figure 2 shows an example of a nonconvex function which has several extrema. In addition to multiple extrema, nonconvex functions pose other challenges including varying curvature, flat regions, deep wells, and saddle points when the curve becomes a higher dimensional surface. Theoretical solutions for these problems do not exist and weak guarantees are based on

application-dependent approximations. Furthermore, there are no recipes for parameter selection associated with any solution.

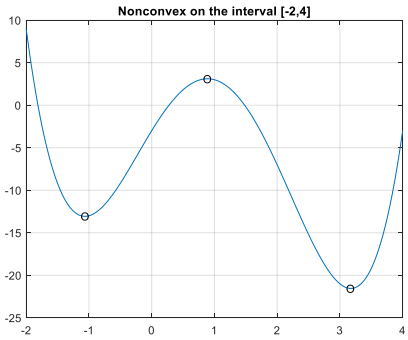

Figure 2: Nonconvexity – multiple extrema [-1, -13], [1, 3], and [3, -22] exist on the interval [-2, 4]

Thus, difficulty in optimizing nonconvex systems is an important topic for many practical applications, processes, and systems. Aircraft and spacecraft optimal control, milling machine optimal cutting speed, automated bidding systems, and complex artificial neural networks associated with machine/deep learning are examples of the diversity of modern nonlinear and nonconvex systems.

Disadvantages of Current Systems

As will be explained later, constrained optimization may be cast into unconstrained optimization by augmenting the problem. Without going into detail, Figure 5 shows some of the more popular multivariate numerical methods applied to constrained problems.

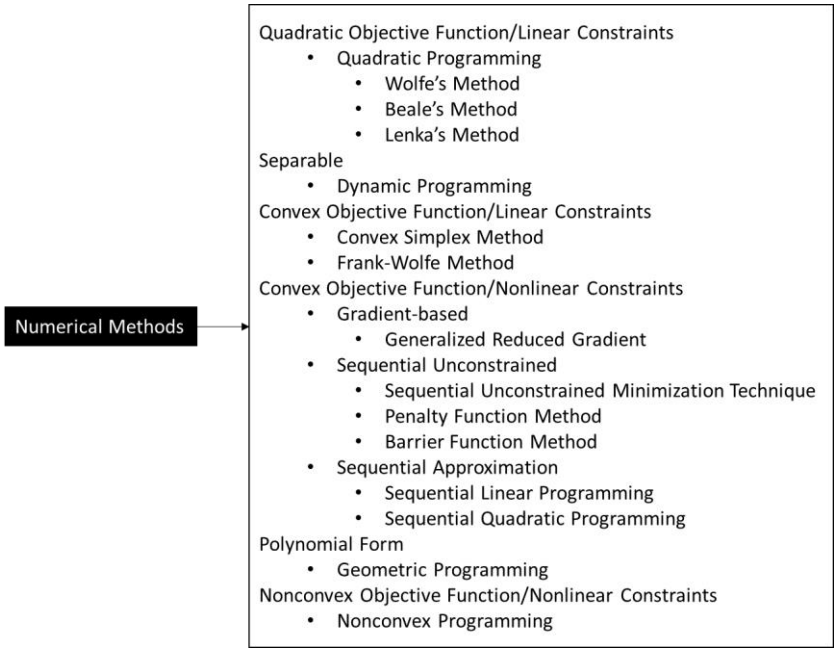

Figure 5: Constrained Optimization Problems and Multivariate Numerical Methods

Whether familiar with these methods or not, it's easy to recognize a sample of their shortcomings.

- *a priori* problem structure knowledge (e.g., quadratic, separable, convex, polynomial) is required for algorithm choice and parameter selection, including independent variables in the nonconvex case.
- Numerical issues associated with (partial) derivatives for *all* gradient-based methods.

- Keen insight needed for the selection of a starting point for function evaluation methods.
- Premature termination of nonconvex algorithms when local extrema are found.

Therefore, a system is needed to solve the most-general nonconvex optimization problem and must be gradient-free (no derivatives or partial derivatives), must not require a priori problem structure knowledge for selecting a starting point or other parameters, and must increase confidence of locating global extrema.

Because architectural foundation of complex problems of relevance contain inherent nonlinearities, nonconvexities, and possible unknown discontinuities, one of the many applications of this innovative technology is a substitution for back-propagation methods, associated with machine/deep learning. That is, instead of back-propagation (which is the backward propagation of errors), this innovative technology achieves minimization of those errors directly.

TECHNICAL APPROACH

Due to inherent nonlinearities, nonconvexities, and discontinuities of many important systems, such as the complex artificial neural networks of machine/deep learning, a system is needed which addresses the disadvantages of current methods.

To avoid numerical issues associated with gradient-based methods, a *gradient-free* method is desired. Furthermore, since artificial decision-making network topologies may include non-differentiable logic gates, gradient-based methods will not suffice for many applications. This also has implications for quantum computation and quantum information by implementation of quantum circuits. These circuits form the basis for implementing various computations. These circuits (quantum gates) are also discontinuous, so gradient-based methods are not suited for these applications.

The proposed system leverages the benefits of both grid search and random search where their combination alleviates their corresponding limitations. As such, the system no longer requires an explicit starting point, but implicitly deploys a plurality of randomly-selected starting points. Because the system is constructed to solve the most general nonconvex optimization problem, this breakthrough approach does not require *a priori* knowledge of the problem structure nor does it require tuning of parameters.

If the problem happens to be well-behaved, meaning the nonconvexity contains relatively few extrema, the likelihood of locating the global extremum is high. However, in the case if high nonconvexity, with many extrema, the system may employ Monte Carlo methods to provide additional outcomes. By collecting the results over many iterations, the outcomes may be sorted to observe the extremum among other extrema. One immediate benefit is the system provides all the results which may be evaluated for cost/benefit analysis. For example, a suboptimal result may be selected due to budgetary or other restrictions. Another immediate benefit is the system provides the probability associated with the aggregate outcomes. Knowing the probability of occurrence associated with local extrema and the global extremum supports trust in artificial intelligence application and sheds light on any potential *black swans*.

Overview

Referring to the flowchart of Figure 6, the system is started by computing the overall number of random variables. Generating random variables leads to an extremum computation and its corresponding coordinates. At this point, the inner loop tolerance is checked.

- If the inner loop tolerance condition is not met, the range of random variables is updated, making sure they lie within their original bounds, and new random variables are generated over this narrowed range, thus restarting the generation process.
- If the inner loop tolerance condition is met, constraints are updated, and the outer loop tolerance is checked.
    - If the outer loop tolerance condition is not met, penalty parameters are increased, and the generation process is restarted.
    - If the outer loop tolerance condition is met, the process ends.

Lastly, the system may employ Monte Carlo methods to collect information for further analysis.

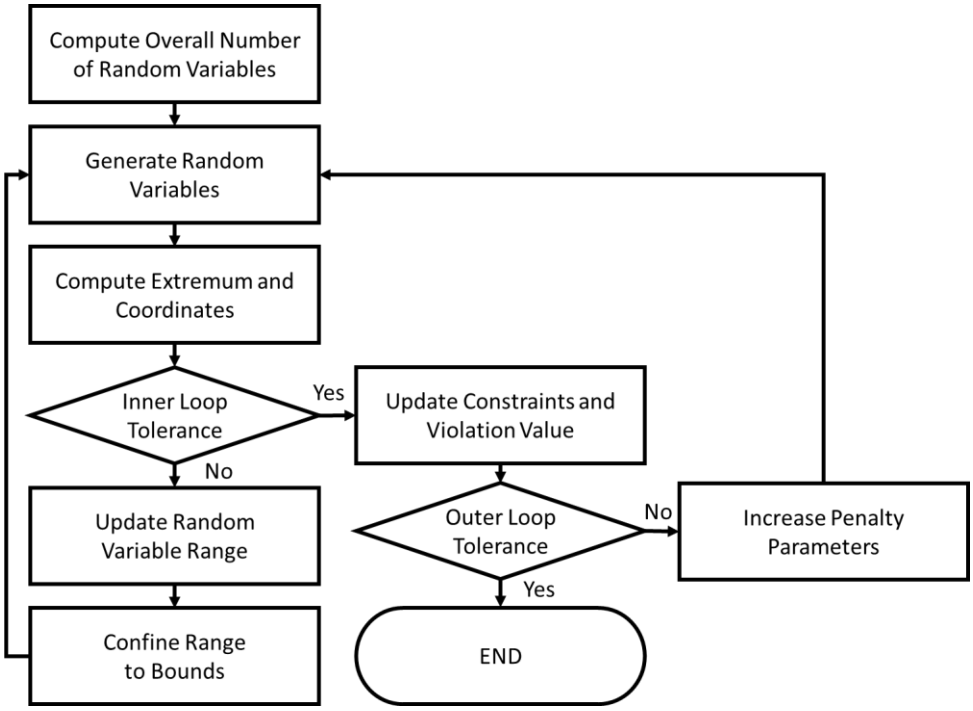

Figure 6: System Flowchart

Details

Before embarking on the details, it is helpful to understand optimization problems are framed within the context of a minimization problem, recognizing a maximization problem is simply the negative of a minimization problem. Additionally, all equality constraints are set equal to zero and all inequality constraints will be set less than zero.

The mathematical models of optimization include independent variables, constraints, and an objective function. Independent variables describe an allocation of resources represented by the model. For example, the number of hours to operate a machine. The system operates to find optimal values of these unknowns. Constraints impose limits on the values the independent variables can assume. For example, the output of a machine cannot be negative. The objective function is a measurement based on the independent variables. The system determines the independent variables such that the objective function is optimized. For example, the cost of operating the machine may be minimized or the output of the machine may be maximized.

The system requires identification of the objective function (f), the inequality (g) and/or equality (h) constraints, and the lower and upper bounds for each independent variable (x). Furthermore, the system requires the convergence tolerance which impacts the precision of the result and defines the resolution of the random grid. Specifying the tolerance determines the number of decimal places required for convergence. For example, if the problem is related to the financial world, a tolerance of 0.01 is specified, meaning the solution should converge to the nearest cent. The sensitivity of each independent variable on the objective function plays a role in the selection of the resolution. While the resolution defines the number of independent variables to be placed across the hyperparameter grid, the independent variables are placed on this grid in a randomly-spaced fashion, as shown in Figure 7.

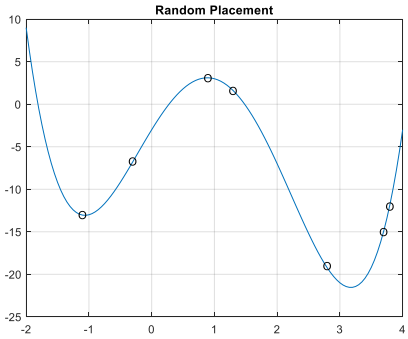

Figure 7: Random Placement of Independent Variables Across the Grid

Referring to the flowchart of Figure 6, the overall number of random variables is computed by subtracting the lower bound from the upper bound (for each independent variable) and dividing the difference by the resolution. The maximum from this set of results is the overall number of random variables used by the system.

Next, random variables are generated for each independent variable based on their lower bound (LB) and upper bound (UB). The inequality constraints (g) and equality constraints (h) are computed, appended to the objective function, and evaluated. This is done for each of the overall number of random variables.

There are several system implementations which append the constraints to the objective function. One such method is through the exact multiplier penalty function. [As an aside, the method of multipliers is used with (i) Newton's method if the Hessian (a matrix of partial derivatives) is available, (ii) a quasi-Newton method if only gradients (derivatives) are available, or (iii) the conjugate gradient method. This system uniquely combines the method of multipliers with the benefits of the grid search and random search avoiding the numerical difficulties associated with derivatives.] The initial optimal Lagrange multiplier (U) is updated from its negligibly small, but nonzero initial value, $U = U + \max(2\mu g, -U)$, where μ is a constraint penalty parameter. Then, four terms are computed which will be subsequently appended to the objective function. The first term is $\mu(\max(g + U/2\mu, 0))^2$. The second term is $U^2/4\mu$. The third term is νh, where ν is another penalty parameter. The fourth term is μh^2. The terms are appended to the objective function as follows, calling special attention to the minus sign on the second term.

$$f + \mu(\max(g + U/2\mu, 0))^2 - \frac{U^2}{4\mu} + \nu h + \mu h^2$$

Note: μ is updated through multiplication, therefore select its initial value to be 1; whereas ν is updated through addition, so its initial value is set to 0.

Returning to the set of repeated evaluations, the minimum and its corresponding set of independent variables are saved. The current minimum value is compared with the previous minimum value. For the first comparison, the initial minimum value is large by design. Since the difference between the initially large minimum value and the current minimum value is large, the small tolerance is certainly exceeded. Therefore, a second iteration is needed. This is the "Inner Loop Tolerance" decision block in shown in the flowchart of Figure 6 (above).

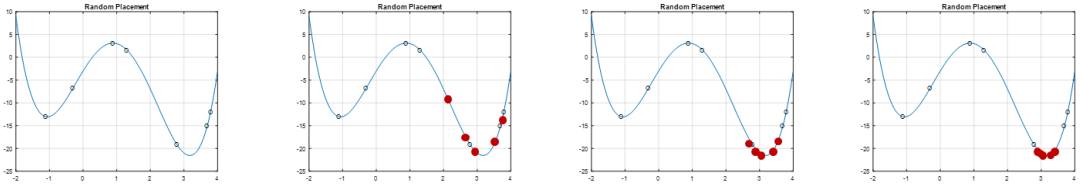

Figure 8: Adaptive Placement of Random Independent Variables

If the inner loop tolerance decision is *not* satisfied, a new range of random variables is centered on the current minimum, based on current coordinates and resolution, $LB = x - resolution$ and $UB = x + resolution$. Since this new range is centered on the location of the current minimum, and since the current minimum could lie near the lower bound or upper bound, the range is checked to be sure it lies above the initial lower or bound or below the initial upper

bound. If it does not, the new bound (lower or upper) is reset to the initial bound value (lower or upper). This iterative approach adaptively produces a finer search for a minimum within tolerance. Figure 8 shows an example of how this iterative approach adapts to produce a finer search. At this point, a set of new random variables are generated for each independent variable, based on their new lower and upper bounds, and the process is restarted.

If, however, the inner loop tolerance decision *is* satisfied, the constraints are updated with the coordinates corresponding to the current minimum. Furthermore, the constraint violation value is updated, which is simply the maximum of the absolute value of all equality constraints, $\max(\text{abs}(h))$. The constraint violation value is compared with the outer loop tolerance. This is the "Outer Loop Tolerance" decision block in shown in the flowchart of Figure 6 (above).

If the outer loop tolerance decision is *not* satisfied, either (1) increase the penalty parameter ($\mu = 10\mu$) or (2) increase the penalty parameter ($v = v + 2\mu h$) and increase the optimal Lagrange multiplier ($U = U + \max(2\mu g, -U)$). The either/or choice is determined by comparing the constraint violation value with an initial large comparison value. If the constraint violation value exceeds the initial large comparison value, choose option (1), otherwise, select option (2). If option (2) is selected, the initial large comparison value is updated by setting it to the current constraint violation value for the next iteration. After selection of (1) or (2), a set of new random variables are generated for each independent variable, and the process is restarted.

If, however, the outer loop tolerance decision *is* satisfied, the process ends, thus obtaining one solution from the system.

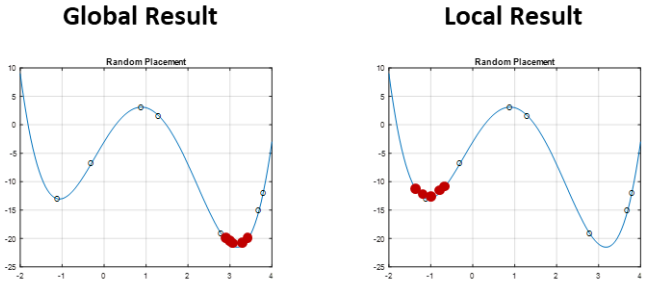

Figure 9: Outcomes from Monte Carlo Methods

Finally, the system may employ Monte Carlo methods which may provide additional outcomes. By collecting the results over many iterations, possible outcomes may be sorted to observe the extremum among other extrema. Figure 9 shows an example of two possibilities: one outcome is a local minimum and the other is the global minimum. Given the random placement of random independent variables across the grid, it is unlikely that the system would terminate on the local minimum. Although unlikely, the probability of the system terminating at the local minimum is not zero. Therefore, the data may be examined to inspect the statistics associated with the possible outcomes. Figure 10 shows a small probability associated with the local minimum outcome compared to the large probability associated with the global minimum outcome. Thus, the system is stochastic in both the adaptive placement of random independent variables over iterations as well as in the application of Monte Carlo methods.

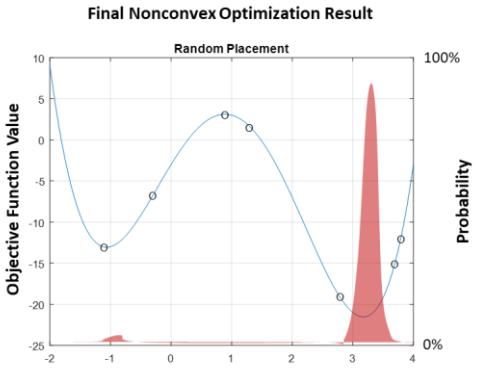

Figure 10: Distribution of Outcomes from Monte Carlo Methods

Testing

The system has been successfully tested with a suite of standard benchmarking problems (Sedlaczek and Eberhard, 2005) including the Rosenbrock and Griewank functions. (The Rosenbrock function contains a global minimum deep within a long narrow parabolic valley. The Griewank function is a sinusoidal function with a parabolic envelope.) Practical nonlinear operations research testing included game theory, decision analysis, queueing theory, and inventory applications. Additional successful testing was performed on naval aviation skill attainment (NAWCTSD) and naval manning models (CNATRA).

USE CASE

As a practical example, consider the process of aerospace vehicle system identification as applied to the estimation of the rolling moment aerodynamic parameter, C_l. One artificial neural net approach uses 5 independent variables to determine 3 dependent variables. After a lengthy and taxing exercise in numerical investigation (input/output scaling, initial network weights, number of hidden nodes, learning rate, momentum parameter, and slope factors of the sigmoidal activation functions) convergence is achieved after 2000 iterations. The result is a complex, opaque, uninterpretable, unexplainable relationship between the inputs and outputs. Also, if there are any changes to the inputs or outputs, the model must be retrained.

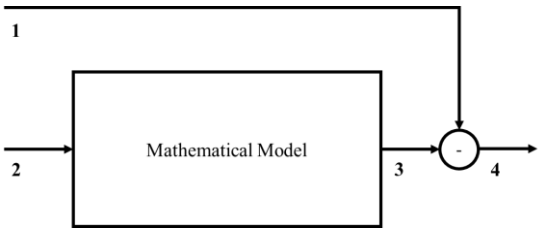

Figure 11: System Architecture

Figure 11 refers to an architecture for system identification problems using the proposed invention. Let the aileron deflection be the known input (signal 2). Let the roll moment aerodynamic parameter be the known output (signal 1). One skilled in aerodynamics will recognize the direct relationship between aileron deflection and rolling moment aerodynamics. One skilled in machine/deep learning will recognize this as a regression form of supervised learning where known input/output data is used to train a model (in this case, determine coefficients of the mathematical model) for a response to new input data. The difference between the computed output (signal 3) and the known output (signal 1) is the error (signal 4). Optimizing (minimizing) the error (signal 4) determines the coefficients of the mathematical functions. Assuming the relationship between input and output is unknown, a generic mathematical model is used. The reason for a generic mathematical model is to demonstrate it's possible to describe the relationship between aileron deflection and roll maneuver without having *a priori* knowledge regarding the aerodynamics. Of course, one skilled in aerodynamics may implement a model describing the mathematics from aileron input to roll maneuver output. The coefficients describe the model and are used to explain the relationship between the input (aileron deflection) and output (roll moment aerodynamic parameter). Rather than using a ratio of 5/3 (output/input) as in the

artificial neural net case, only 1 is used for both input and output with the proposed invention, i.e., 40% less data is required to determine the relationship between input and output. Furthermore, the results are achieved in 200 iterations – an order of magnitude *less* than required by the artificial neural net approach.

While the model is still relatively complex, it is transparent, interpretable, and explainable. Because of these attributes, this innovative technology is much more reliable, especially for flight safety certification. Finally, the mathematical model can be subsequently exercised to explore extreme cases, e.g., letting variables go to zero and letting variables approach infinity. Hence increasing confidence model deployment.

Other use cases include applications to machine/deep learning, reinforcement learning, cryptography, and self-organized complex systems.

SUMMARY/CONCLUSION

In short, the advantages are

- The system is gradient-free, numerically stable, and allows for the inclusion of logic/quantum gates.
- There is no need for *a priori* information to start the system with a known set of coordinates.
- The system configuration can be established to have control over the precision of the search by selecting the resolution.
- There is no need to follow any prescribed architectural philosophy.
- By design, the system solves the most-general optimization problem.

The system may be applied to prevalent problems of operations research and management sciences including game theory, decision analysis, queueing theory, inventory theory, transportation, and networking. The system may also be applied to supervised (regression and classification) and unsupervised (clustering) machine learning architectures. The ability to handle probabilistic inputs, coupled with the inclusion of logic gate architecture, facilitates another application to quantum computing, e.g., quantum Fourier series and quantum cryptography.

The system may be configured for real-time applications offering many advantages. Of course, real-time requirements for guidance, navigation, and control processes are different than real-time requirements for e-commerce transactions. In either case, the system may be augmented such that known constraints (if any) could be built into the objective function. Also, by applying an appropriate resolution, the system may be configured to execute in a deterministic time frame. This is something worth exploring.

REFERENCES

1. Bazaraa, M., et al., (2006), Nonlinear Programming: Theory and Algorithms, Wiley.
2. Chen, R., et al., (2019), Neural Ordinary Differential Equations, arXiv:1806.07366.
3. DARPA, (2018), https://www.darpa.mil/news-events/2018-07-20a.
4. Fouskakis, D., and Draper, D., (2001), Stochastic Optimization: A Review, International Statistical Review.
5. Hillier, F.S. and Hillier, M.S., (2003), Introduction to Management Science, McGraw Hill.
6. Hillier, F.S. and Lieberman, G.J., (2005), Introduction to Operations Research, McGraw Hill.
7. Jain, P. and Kar, P., (2017), Non-Convex Optimization for Machine Learning, arXiv:1712.07897.
8. Kitaev, A., et al., (2002), Classical and Quantum Computation, American Mathematical Society.
9. Knight, W., (2017), The U.S. Military Wants Its Autonomous Machines to Explain Themselves, https://www.technologyreview.com/s/603795/the-us-military-wants-its-autonomous-machines-to-explain-themselves.
10. Kolman, B., and Beck, R.E., (1980), Elementary Linear Programming with Applications, Academic Press.
11. Lamberth, M., (2019), The White House and Defense Department unveiled AI strategies. Now what?, https://www.c4isrnet.com/opinion/2019/02/27/the-white-house-and-defense-department-unveiled-ai-strategies-now-what.
12. Mermin, N.D., (2007), Quantum Computer Science, An Introduction, Cambridge University Press.
13. Nielsen, M., znd Chuang, I., (2016), Quantum Computation and Quantum Information, Cambridge University Press.

14. Sedlaczek, K., and Eberhard, P., (2005), Constrained Particle Swarm Optimization of Mechanical Systems, 6th World Congresses of Structural and Multidisciplinary Optimization.