THE PHILIPPINE-AMERICAN Academy of Science & Engineering

# Development and Evaluation of a Modified Luus-Jaakola Adaptive Random Search Algorithm

Sed Anderson K. Holaysan, Luis F. Razon[a], and Raymond R. Tan[a*]

[a] Department of Chemical Engineering
De La Salle University, 2401 Taft Avenue, Manila, Philippines

[*]E-mail: raymond.tan@dlsu.edu.ph

## ABSTRACT

Process systems engineering (PSE) approaches are useful for facilitating the optimal design and operation of industrial plants. However, in practice, the development of high-fidelity models results in non-linearity, which in turn results in computational difficulties. This study develops a modified metaheuristic algorithm for optimization of nonlinear (NLP) and mixed-integer nonlinear programming (MINLP) models. It is based on the Luus-Jaakola adaptive random search (LJ-ARS), but has been modified by incorporating some features from the line-up competition algorithm (LCA). Multiple points are used to explore the search space and are ranked according to the objective function. Each point moves towards a better-ranked point to improve its position. Each point's search space is also influenced by its rank, so as to provide a balance between exploration and exploitation. A lower limit for the space reduction factor is specified to prevent premature convergence. A probabilistic rounding-off procedure is used for integer variables, while the penalty function approach is used for constraint resolution. Eleven benchmark chemical engineering test problems are used to evaluate the performance of the modified LJ-ARS in comparison to the original LJ-ARS. Criteria for comparison are accuracy and consistency. Results show that the modified algorithm demonstrates better or comparable performance compared to the original algorithm for majority of the test problems.

**KEYWORDS:** Metaheuristic; Optimization; Nonlinear Programming; Mixed-Integer Nonlinear Programming; Process Systems Engineering

## 1 INTRODUCTION

Process systems engineering (PSE) techniques can be used to model industrial plants and determine their optimal design. If a process system is represented as a linear model, location of the optimal solution can be guaranteed using standard methods such as the simplex algorithm; however, the presence of nonlinearities results in significant computational difficulties (Martelli and Amaldi, 2013). Various algorithms for optimization of nonlinear programs are currently in use, and metaheuristic or stochastic search techniques are one such class of algorithms. One of these is direct search or adaptive random search by Luus and Jaakola (1973). Other metaheuristics developed based on analogies to real life behavior include simulated annealing by Kirkpatrick et al. (1983), genetic algorithm by Holland (1975), differential evolution by Price and Storn (1997), line-up competition by Yan and Ma (2001), and particle swarm by Kennedy and Eberhart (1995).

Further improvements to the original Luus-Jaakola method have been proposed through adjustment of algorithm parameters. Poplewski et al. (2011) modified this algorithm by using multiple data search region contraction parameters rather than a fixed one for all variables. Litinetski and Abramzon (1998)

proposed the use of multiple starting points, but it was determined that using a smaller number of points over multiple passes is more effective than using a large number of points over a single pass (Luus, 2003). Alternative methods for constraint resolution within the LJ-ARS framework have been suggested. Luus (1996) demonstrated the efficiency of a quadratic penalty function coupled with a Lagrange multiplier for the solution of models containing equality constraints. Other alternative methods which incorporate numerical methods for the solution of equality constraints were proposed by Luus and Wyrwicz (1996), and Luus (2000).

The line-up competition algorithm (LCA) of Yan and Ma (2001) used ranking to adjust search space and used penalty functions to resolve constraints. It was shown to take lesser computation time than other evolutionary algorithms when used to optimize NLP models (Yan and Ma, 2001) and MINLP models representing multi-product chemical batch processes (Yan et al., 2004). Hybrid algorithms have been formulated to combine the strengths of multiple metaheuristics. Adaptive random search may be used as the primary search pattern, or as a subroutine to a different search procedure. Gao et al. (2004) used direct search to strengthen the local search of particle swarm optimization of machine parameters, and showed that the modification exhibited better speed and accuracy than basic genetic algorithms and simulated annealing procedures. Chen (2008) used a local random search particle swarm algorithm, which yielded better results than the classic particle swarm algorithm and an evolutionary programming method. A hybrid of direct search, particle swarm, generating set search and complex was developed by Martelli and Amaldi (2013), and was determined to be robust for non-smooth problems with unrelaxable constraints or evaluation failures.

Sacco et al. (2008) showed that the direct search algorithm yields results at least on par with those from genetic algorithms, particle swarm, and simulated annealing. Jeżowski et al. (2005) showed that direct search works better than genetic algorithms for problems with discrete variables, while Liao and Luus (2005) demonstrated that the Luus-Jaakola algorithm is generally faster, more reliable, and easier to use than genetic algorithms. The ease of programming the Luus-Jaakola adaptive random search (LJ-ARS) procedure has led to its frequent use, but there remains room for improvement.

The Luus-Jaakola algorithm adapted for optimization of discrete variables (Luus, 1975) relies on an initially established global pseudo-solution which assumes all variables to be continuous. This results in a search limited to the neighborhood of the pseudo-solution, while the global discrete optimum may be in the vicinity of a sub-optimal continuous pseudo-solution. The solution may be affected by the position of the initial point, the search interval, and the search space reduction factor (Salcedo et al., 1990). However, finding good values for these parameters is often dependent on problem class and characteristics (Liao and Luus, 2005). Furthermore, it has been found that metaheuristics do not always succeed in finding the global optimum quickly or consistently for certain models, such as when phase equilibria and thermodynamics are involved, and it may be necessary to formulate specific algorithms geared towards certain types of problem structures (Fernandez-Vargas et al., 2013).

The rest of this paper is organized as described in this paragraph. The formal problem statement is given in the next section. The original LJ-ARS procedure and line-up competition algorithm (LCA) are presented. The development of the modified LJ-ARS is outlined, and some of its features are highlighted. The modified algorithm is compared to the original algorithm using eleven chemical engineering benchmark problems from related literature, and the results are discussed. Conclusions about the performance of the modified algorithm are drawn, and recommendations for future work are suggested.

## 2 PROBLEM STATEMENT

The general formulation of a nonlinear model to be optimized is:

$$Optimize\ f = f(\boldsymbol{x}) = f(x_1, x_2 \dots x_n) \tag{1}$$

$$g_i(x_1, x_2 \dots x_n) \leq 0 \tag{2}$$

$$h_i(x_1, x_2 \dots x_n) = 0 \tag{3}$$

where Equation (1) is the objective function, Equation (2) represents inequality constraints, and Equation (3) represents equality constraints (Boyd and Vandenberghe, 2004). The model is considered nonlinear if at least one nonlinear term is present, either in the objective function or among the constraints. A modified Luus-Jaakola adaptive random search (LJ-ARS) algorithm is formulated for the optimization of such nonlinear models.

## 3 DEVELOPMENT OF MODIFIED ALGORITHM

The modified algorithm is initialized by generating a number of random starting points and evaluating them according to the objective function. These points are ranked from worst to best. A movement step is performed by having a point of lower rank (denoted as point *j*) move towards the point ranked just above it (point (*j*-1)). The magnitude of this movement is proportional to the difference of their objective function evaluations. This step is performed for $j = n, (n-1), (n-2), \dots, 2$. For any given iteration, this movement is mathematically written as shown in Equation 4.

$$x_{jd} = x_{jd} + k_d(f_{(j-1)} - f_j) \tag{4}$$

where *x* is the location of a point with respect to a given dimension, subscript j refers to a certain point (according to rank), subscript *d* refers to a given dimension, *f* is the objective function value located by the given point, and *k* is a scaling factor that adjusts for the search space allowed for each point. The scaling factor *k* is specific for each dimension, and is based on the upper and lower bounds in that given dimension.

The search space $s_j$ of each point varies with its rank, in a manner similar to the scheme of Yan and Ma (2001). However, instead of assigning search space multipliers uniformly between 0 to 1, a minimum value $srf_{min}$ is used to prevent premature convergence. The effect of ranking on search space, for a given iteration and a given dimension, can be written as shown in Equation 5:

$$s_j = \left[srf_{min} + \left(\frac{j-1}{n-1}\right)(1 - srf_{min})\right] s_j \tag{5}$$

This expression causes the best-ranked point's search space to be multiplied by the factor $srf_{min}$, while the worst-ranked point's search space remains unchanged. The search spaces of the other points are multiplied by factors evenly distributed from $srf_{min}$ to 1.

While the generation of new points assumes that all variables are continuous, the floor and ceiling functions are used to round off these values to the nearest integers. The closer the value of a continuous variable is to an integer, the probability that rounding off will be in that direction is increased proportionally. Whenever an unrelaxable constraint is violated (an undefined value results from a constraint or function evaluation), a random feasible location (in that specific dimension) is generated

from the nearest allowable value and the current range. Violations of relaxable constraints are resolved using a dynamic linear penalty function shown in Equation 6, which is either added to (for minimization problems) or subtracted from (for maximization problems) the objective function.

$$P = (PIF)^{l-1}(PC)(\sum_{i=1}^{n}|h_i| + \sum_{i=1}^{n} max(0, g_i)) \quad (6)$$

The penalty coefficient *PC* is multiplied to the total sum of penalties for appropriate weight against the objective function, while *PIF* represents the penalty increase factor which increases after each iteration. Following the notation used in Equations 1-3, the first summation term refers to the total equality constraint violation, and the second summation term refers to the total inequality constraint violation. The flowchart of the modified LJ-ARS is shown in Figure 1.

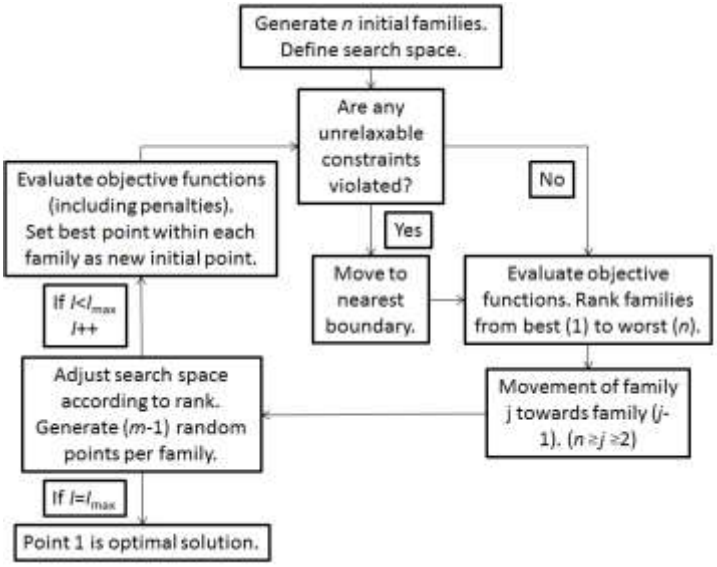


Figure 1: Flowchart of Modified LJ-ARS Algorithm

## 4 EVALUATION OF ALGORITHM

The program to be used for demonstration has been written using Microsoft Excel and Visual Basic for Applications (VBA). The modified LJ-ARS and the original LJ-ARS have been used to solve 11 chemical engineering optimization problems from related literature (Ryoo and Sahinidis, 1995) to compare their relative performances. The criteria for comparison were accuracy, measured as the average solution over 10 runs; and consistency, measured as the standard deviation over the same 10 runs. Data on each problem's global optimum is available from related literature (Ryoo and Sahinidis, 1995). A maximum number of function evaluations per obtained final solution was used as a stopping criterion. The comparison of results for both algorithms is shown in Table 1. Initial trial-and-error was used to determine algorithm parameters such as penalty coefficients and space reduction factors.

Table 1 Comparison of Modified LJ-ARS and Original LJ-ARS on Sample Problems

| Problem | Global Optimum | Original LJ-ARS | | Modified LJ-ARS | | Function Evaluations |
|---|---|---|---|---|---|---|
| | | Average Solution | Standard Deviation | Average Solution | Standard Deviation | |
| 1 | 2 | 2.0476 | 0.1058 | 2.0004 | 0.0006 | 750 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 0 | 0.3990 | 1.976 | 0.02537 | 0.03683 | 3,000 |
| 3 | 5,194.9 | 19,713.5 | 14,600 | 6,542.9 | 440.82 | 15,000 |
| 4 | -4.5142 | -4.4644 | 0.0093 | -4.4573 | 0.0174 | 12,000 |
| 5 | 7,049.2 | 8,872.2 | 7,655.8 | 17697 | 7,922.6 | 15,000 |
| 6 | -0.3888 | -0.3649 | 0.0017 | -0.3684 | 0.0022 | 12,000 |
| 7 | -13.402 | -12.898 | 0.3429 | -12.353 | 1.326 | 15,000 |
| 8 | 4.5796 | 4.6726 | 0.2075 | 4.5875 | 0.0100 | 3,000 |
| 9 | -1,161 | 73,425 | 92495 | 49,235 | 35,358 | 15,000 |
| 10 | -400 | -341.33 | 109.47 | -297.33 | 62.282 | 15,000 |
| 11 | 12,292 | 25,719 | 8,696.2 | 14,771 | 2,764 | 30,000 |

The modified LJ-ARS algorithm finds a significantly better (percent difference above 5%) average solution than the original algorithm for 6 of the 11 test problems (including 3 of the 4 largest problems). For 3 other problems, both algorithms yield average solutions within five percent difference or less. The modified LJ-ARS shows better consistency in solutions for 7 of the 11 problems (including the 4 largest problems), having a standard deviation significantly less (percent difference above 5%) than that of the original algorithm. Both algorithms show a similar standard deviation for Problem 5. The use of a dynamic penalty coefficient only improved the results for Problem 1. The modified algorithm demonstrates accuracy and consistency for Problem 2, an equilibrium model, which the original LJ-ARS is known to be inconsistent for (Fernandez-Vargas et al., 2013).

Interestingly, in each problem where the modified algorithm finds a significantly better average solution than the original algorithm (Problems 1, 2, 3, 8, 9, and 11), it also demonstrates a significant improvement in consistency. For both mixed-integer nonlinear programming models (Problems 1 and 8), the modified algorithm performs better in terms of both the average solution and consistency.

## 5 CONCLUSIONS AND RECOMMENDATIONS

For more than half of the test problems, the modified LJ-ARS displays a significant improvement over the original algorithm in terms of accuracy and consistency. For the remaining problems, the modified algorithm may hold an advantage or may show no significant difference in one aspect, but it is never inferior to the original algorithm with respect to both criteria. This indicates that the modified LJ-ARS may be used for optimization of nonlinear programming (NLP) models and mixed-integer nonlinear programming (MINLP) models.

The modified LJ-ARS can be improved further by adding a reset mechanism to explore the search space more thoroughly. This may be particularly advantageous for problems with a large number of variables. Another possibility is the use of a dynamic search space factor that allows increase of the search space, such as when the total infeasibility of the model is relatively high. This can allow an algorithm to locate feasible regions more rapidly, especially for models containing a large number of equality constraints. A heuristic strategy can be also formulated for selection of a good initial point, as opposed to a completely random initialization.

## REFERENCES

Boyd, S., & Vandenberghe, L. (2004). Convex Optimization. Cambridge University Press, New York.

Chen, G. (2008). Power station short-term generation optimal dispatch based on local random search PSO algorithm. Electric Power Automation Equipment, 28:5, 52-55.

Fernandez-Vargas, J. A., Bonilla-Petriciolet, A. B., & Segovia-Hernández, J. G. (2013). An improved ant colony optimization method and its application for the thermodynamic modeling of phase equilibrium. Fluid Phase Equilibria, 353, 121-131.

Gao, L., Gao H., & Zhou, C. (2004). Particle swarm optimization based algorithm for machining parameter optimization. Proceedings of the World Congress on Intelligent Control and Automation (WCICA), 4, 2867-2871.

Holland, J. H. (1975). Adaptation in Natural and Artificial Systems. The University of Michigan Press. Ann Arbor, Michigan.

Jeżowski, J., Bochenek, R., & Ziomek, G. (2005). Random search optimization approach for highly multi-modal nonlinear problems. Advances in Engineering Software, 36:8, 504-517.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. Proc. IEEE Int. Conf. Neural Netw., 1942-1948. Piscataway, NJ.

Kirkpatrick, S., Gelatt, C. D. Jr., & Vecchi, M. P. (1983). Optimization by simulated annealing. Science, 220, 671-680.

Liao, B., & Luus, R. (2005). Comparison of the Luus-Jaakola optimization procedure and the genetic algorithm. Engineering Optimization, 37:4, 381-398.

Litinetski, V. V., & Abramzon, B. M. (1998). Mars – a multistart adaptive random search method for global constrained optimization in engineering applications. Engineering Optimization, 30:2, 125-154.

Luus, R. (1975). Optimization of system reliability by a new nonlinear integer programming procedure. IEEE Transactions on Reliability, 24:1, 14-16.

Luus, R. (1996). Handling difficult equality constraints in direct search optimization. Hung. J. Indus. Chem., 24:4, 285-290.

Luus, R. (2000). Handling difficult equality constraints in direct search optimization. Part 2. Hung. J. Indus. Chem., 28:3, 211-215.

Luus, R. (2003). Effect of region collapse parameter on convergence rate of LJ optimization procedure. Proceedings of the IASTED International Conference on Intelligent Systems and Control, 51-56. Salzburg, Austria.

Luus, R., & Jaakola, T. H. I. (1973). Optimization by direct search and systematic reduction of the size of search region. AIChE Journal, 19:4, 760-766.

Luus, R., & Wyrwicz, R. (1996). Use of penalty functions in direct search optimization. Hungarian Journal of Industrial Chemistry, 24:4, 273-278.

Martelli, E., & Amaldi, E. (2013). A novel hybrid direct search method for constrained non-smooth black-box problems. Computer-Aided Chemical Engineering, 32, 295-300.

Poplewski, G., Jeżowski, J. M., & Jeżowska, A. (2011). Water network design with stochastic optimization approach. Chemical Engineering Research and Design, 89:10, 2085-2101.

Price, K., & Storn, R. (1997). Differential evolution – A simple evolution strategy for fast optimization. Dr. Dobb's Journal, 22, 18-24, 78.

Ryoo, H. S., & Sahinidis, N. V. (1995). Global optimization of nonconvex NLPs and MINLPs with applications in process design. Computers and Chemical Engineering, 19, 551-566.

Sacco, W. F., Filho, H. A., & Platt, G. M. (2008). The Luus-Jaakola algorithm applied to a nuclear reactor core design optimization. International Journal of Nuclear Science and Technology, 4:1, 1-10.

Salcedo, R., Goncalves, M. J., & Feyo de Azevedo, S. (1990). An improved random-search algorithm for non-linear optimization. Computers and Chemical Engineering, 14:10, 1111-1126.

Yan, L. X., & Ma, D. X. (2001). Global optimization of non-convex nonlinear programs using Line-up Competition Algorithm. Computers and Chemical Engineering, 25, 1601-1610.

Yan, L. X., Shen, K., & Hu, S. H. (2004). Solving mixed integer nonlinear programming problems with line-up competition algorithm. Computers and Chemical Engineering, 28, 2647-2657.