Evaluations of An Algorithm for Large Multivariate Optimization

Yoel TENNE*
Ariel University, Israel

*Corresponding author email: y.tenne@post.com

Abstract. Many simulation-driven engineering and scientific problems require finding an optimum of a function with many variables. Such settings pose a challenge for standard algorithms due to the large search space which in turn can lead to poor final results. Therefore this paper proposes a new simplified approach in which the dimension of the problem is dynamically reduced during the search to formulate a problem of lower size (dimension) which is easier to solve. A main novelty of the algorithm is its simplicity. Numerical experiments show the potential of this approach.

1. Introduction
Engineering and scientific problems commonly use numerical codes to evaluate designs and are thus cast as optimization problems where the best system configuration is sought. Some examples include problems in structural, fluid dynamics, and heat transfer domains [1-4]. One main challenge is that the computer simulation is computationally intensive hence only a few simulation runs are allowed. Another challenge is the number of variables which defines a large search space. Various optimization algorithms have been proposed but were developed for low-dimensional problems (namely, having a small number of variables). However, real-world problems often involve tens or more variables and in such settings the performance of such algorithms can degrade quickly. This can yield an inefficient search and an inadequate final outcome, particularly when the resources allocated to the search process are limited. As such this paper describes an approach in which the original problem with multiple variables is reformulated into a simplified problem with a reduced set of variables which is therefore easier to solve. A dedicated search algorithm is then used to seek an optimum and the obtained solution is mapped into the original high-dimensional space. Numerical experiments show the potential of the proposed approach and its applicability to different problems. The remainder of the paper is set as follows: Section 2 provides background information, Section 3 describes the proposed approach, and Section 4 the numerical experiments and their analysis. Section 5 then concludes the paper.

2. Background
As described in Section 1 optimization problems where objective values are derived from a numerical code introduce several optimization challenges where common techniques may struggle. This has motivated the development of new heuristic approaches which often mimic phenomena in nature as solution methods [6,7]. The extensive computer resources required by each simulation run restricts the number of possible evaluations and accordingly metamodels (alternatively surrogate-models), which are mathematical approximations trained by using sampled data, can be used to obtain approximate function values more efficiently [8]. Metamodel types include polynomials, radial basis functions, and
various neural networks. Due to the limited number of possible simulations runs the obtained metamodel will likely be inaccurate and to address this it is required to check this accuracy and to update the metamodel during the search to avoid convergence to a false solution [9]. To accomplish this the proposed method uses the trust-region (TR) approach which performs a series of bounded steps and based on the success or failure of each optimization trial various updates are performed, as explained in the following section.

3. The Proposed Algorithm
The proposed algorithm combines three elements: a hybrid metaheuristic search, a trust region framework, and a dimensionality reduction procedure, as follows.

The hybrid metaheuristic search begins by training a Kriging metamodel [10] based on previously evaluated vectors such that the metamodel prediction is

\[ m(x) = \beta + r(x)^T R^{-1}(f - 1\beta) \]  

where \( \beta \) is a global “drift” coefficient, \( r(x) \) is the vector of correlation values between the new vector \( x \) and the sampled vectors, \( R \) is the matrix of correlations values between sampled vectors, and \( f \) is the vector of responses (observed function values). After the metamodel has been trained a metaheuristic search is used to find its optimum. The optimization search combined an evolutionary algorithm (EA) followed by a local gradient search to achieve a good global-local search balance. The EA used a population of 100 vectors, stochastic universal selection (75% probability), mutation (10% probability), 10% elitism, and 50 generations. The localized search was performed by a sequential quadratic programming (SQP) algorithm. As mentioned the search is performed based on a trust-region (TR) approach such that the search is confined to a region \( T \) of radius \( r \) around the best solution

\[ T = \|x - x_B\|_2 < r \]  

The solution found by the EA+SQP search is evaluated with the true objective function. If it is indeed better than the known best then the TR radius is enlarged. Otherwise either a new vector is added in the trust-region (in case the number of vectors was insufficient, to avoid premature termination [7]) or the TR radius is decreased.

With regards to the dimensionality reduction, the procedure involves a) selecting the reduced dimension \( d \) and b) selecting which \( d \) variables will be active out of the existing \( n \). This accomplished by modifying \( d \) during the search: starting from the original dimension \( n \) the initial reduced dimension is set as \( d^{(1)} = n/k \), \( k > 1 \) and a subset selection is performed to identify the \( d^{(k)} \) most important variables (as explained in the following text) and then the metamodel based search is performed. Lastly, the reduced dimension is adjusted based on the progress in the recent \( s \) iterations: if a better solution has been found than the algorithm further reduces the dimension to attempt a more intensive reduction. Otherwise the dimension is increased, namely \( d^{(i+1)} = d^{(i)} \cdot k \) so the reduced dimension is closer to the original one. Based on numerical tests the selected settings where a dimension is updated every \( s = 5 \) iterations and a dimensionality change factor of \( k = 1.1 \) which was selected to avoid too rapid changes in the dimension.

Regarding the subset selection step – given \( n \) variables and a reduced dimension \( d < n \) then there are \( \frac{n!}{d!(n-d)!} \) possible subsets which implies that the search quickly become too vast to systematically explore. Therefore to more effectively search for a good subset the algorithm uses a binary evolutionary search which is different from the real-coded EA used for search the metamodel. With this EA different combinations where represented as binary vectors (1 indicates an active variable and 0 and inactive variable) and the prediction accuracy of different combinations was assessed based on cross-validation, namely, for each perspective subset a metamodel is trained such that the sampled vectors are split in a 80–20 training-testing ratio, a metamodel is trained based on the training set and its accuracy is estimated based on the testing set. The binary evolutionary algorithm used stochastic universal selection, shuffle crossover with a probability of 0.75, discrete mutation with a probability of
0.1, a population size of 100, 10% elitism, and 50 generations. To finalize this section Algorithm 1 presents the pseudocode of the complete algorithm.

```
sample an initial set of vectors;
set an initial reduced dimension \( d \);
use subset selection to select \( d \) active variables;
repeat
    train a metamodel based on the extracted data set (sampled vectors and \( d \));
    perform a TR search based on a hybrid EA+SQP search;
    if the found optimum is better than the current best set it as the TR centre
    else if there are sufficient vectors in the TR then reduce the TR radius
    else sample a new vector in the TR;
    every \( s \) iterations
        if the optimum found is better than current best known reduce \( d \)
        else increase \( d \);
        re-select the important variables with subset selection;
until the prescribed number of function evaluations is reached;
```

Algorithm 1: The proposed algorithm

4. Evaluation
To test the proposed algorithm it was used with an established set of mathematical test functions [12] and was benchmarked against a baseline version which did not use dimensionality reduction (DR), namely identical in operation except that the dimensionality reduction step was disabled, which was done to emphasize the impact of the DR step. For each test the function dimension was \( n = 50 \), the limit of function evaluations was 100, and 30 tests were repeated with each function-algorithm combination to obtain statistically-valid results. The code was implemented in Gnu Octave.
Table 1 presents the results where lower results are better (all problems are minimization) and statistics are based on the best function values achieved over 30 tests. From analysis of the results the contribution of the DR step is evident (lower results are better as these are minimization problems). The proposed algorithm consistently achieved better final results than the baseline version across different test functions in 4 out 5 cases. The only function for which the baseline version was better was Rosenbrock which since its valley shape landscape allows to obtain the optimum effectively even in the original high-dimensional formulation. The consistent performance shows that the algorithm was effective across problems with significantly different features.

| Function   | Statistic | Proposed     | Baseline     |
|------------|-----------|--------------|--------------|
| Ackley     | Mean      | \(1.578E+1\) | \(1.729E+1\) |
|            | Median    | \(1.581E+1\) | \(1.725E+1\) |
|            | SD        | \(8.202E-1\) | \(7.351E-1\) |
| Rastrigin  | Mean      | \(5.490E+2\) | \(5.760E+2\) |
|            | Median    | \(5.375E+2\) | \(5.798E+2\) |
|            | SD        | \(6.105E+1\) | \(7.851E+1\) |
| Rosenbrock | Mean      | \(2.621E+3\) | \(1.259E+3\) |
|            | Median    | \(2.472E+3\) | \(1.025E+3\) |
|            | SD        | \(9.517E+2\) | \(1.702E+2\) |
| Schwefel   | Mean      | \(4.105E+6\) | \(4.229E+6\) |
|            | Median    | \(3.955E+6\) | \(2.677E+6\) |
|            | SD        | \(1.093E+6\) | \(2.689E+6\) |
| Weierstarss| Mean      | \(-5.159E+1\)| \(-4.721E+1\) |
|            | Median    | \(-5.181E+1\)| \(-4.882E+1\) |
|            | SD        | \(1.965E+0\) | \(1.337E+0\) |

SD: Standard deviation
The best mean result statistic is emphasized

5. Conclusion
Real-world engineering and scientific problems often involve a large number of variables and in these cases classical optimization algorithms can underperform. Therefore this study proposed an approach where a lower dimension formulation is generated to simplify the search. The extent of dimensionality reduction is dynamically adjusted during the search. A main novelty of the proposed algorithm is its simplicity which requires minimal computational overhead. Evaluations based on test functions indicates that the algorithm is effective in varied settings. Future work could improve the computational efficiency and explore more elaborate adjustments of the reduction factor.

References
[1] Y. Tenne and C. K. Goh, Eds., Computational Intelligence in Expensive Optimization Problems, ser. Evolutionary Learning and Optimization. Berlin: Springer, 2010, vol. 2.
[2] X. Dinh and K. K. Anh, “Radial basis function neural network based adaptive fast nonsingular terminal sliding mode controller for piezo positioning stage,” International Journal of Control, Automation, and Systems, vol. 15, no. 6, pp. 2892–2905, 2017.
[3] B. Talgorn, S. Le Digabel, and M. Kokkolaras, “Statistical surrogate formulations for simulation-based design optimization,” Journal of Mechanical Design, vol. 137, no. 2, 2015. Paper number MD-14-1128 (Online).
[4] T. Wortmann, A. Costa, G. Nannicini, and T. Schroepper, “Advantages of surrogate models for architectural design optimization,” Artificial Intelligence for Engineering Design, Analysis and Manufacturing, vol. 29, no. 4, pp. 471–481, 2015.
[5] T. Santhanam and M. S. Padmavathi, “Application of K-Means and Genetic Algorithms for Dimension Reduction by Integrating SVM for Diabetes Diagnosis,” Procedia Computer Science, vol. 47, pp. 76–83, 2015.
[6] C. Wang, J.-H. Wang, S.-S. Gu, X. Wang, and Y.-X. Zhang, “Elongation prediction of steel- strips in annealing furnace with deep learning via improved incremental extreme learning machine,” International Journal of Control, Automation, and Systems, vol. 15, no. 3, pp. 1466–1477, 2017.
[7] D. A. Rodriguez, P. P. Oteiza, and N. B. Brignole, “Simulated annealing optimization for hydrocarbon pipeline networks,” Journal of Industrial Engineering and Chemical Research, vol. 52, no. 25, pp. 8579–8588, 2013.
[8] F. A. C. Viana, R. T. Haftka, and L. T. Watson, “Efficient global optimization algorithm assisted by multiple surrogate technique,” Journal of Global Optimization, vol. 56, no. 2, pp. 669–689, 2013.
[9] Y. Jin, M. Olhofer, and B. Sendhoff, “A framework for evolutionary optimization with approximate fitness functions,” IEEE Transactions on evolutionary computation, vol. 6, no. 5, pp. 481–494, 2002.
[10] A. R. Conn, N. I. M. Gould, and P. L. Toint, Trust Region Methods. Philadelphia, Pennsylvania: SIAM, 2000.
[11] J. Muller and C. A. Shoemaker, “Influence of ensemble surrogate models and sampling strategy on the solution quality of algorithms for computationally expensive black-box global optimization problems,” Journal of Global Optimization, vol. 60, no. 2, pp. 123–144, 2014.
[12] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. P. Chen, A. Auger, and S. Tiwari, “Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization,” Nanyang Technological University, Singapore and Kanpur Genetic Algorithms Laboratory, Indian Institute of Technology Kanpur, India, Technical Report KanGAL 2005005, 2005.