A Generalizable Hybrid Search Framework for Optimizing Expensive Design Problems using Surrogate Models

Journal:	*Engineering Optimization*
Manuscript ID	GENO-2020-0264.R1
Manuscript Type:	Original Article
Date Submitted by the Author:	n/a
Complete List of Authors:	Cosenza, Zachary; University of California, Chemical Engineering Block, David; University of California Davis, Department of Viticulture and Enology
Keywords:	hybrid surrogate, meta-modeling, DYCORS, experimental optimization, radial basis functions

SCHOLARONE™
Manuscripts

Attribution 4.0 International (CC BY 4.0)

Zachary Cosenza[a] and David E Block[b*]

[a]*Department of Chemical Engineering, University of California, Davis, USA;*
[b]**Department of Viticulture and Enology, University of California, Davis, USA*

*David E Block, Department of Viticulture and Enology, University of California, 1 Shields Avenue, Davis CA 95616, deblock@ucdavis.edu.

[a]Zachary Cosenza; Email: zacosenza@ucdavis.edu, ORCID: https://orcid.org/0000-0002-3944-7948.

[b]David E Block; Email: deblock@ucdavis.edu. ORCID: https://orcid.org/0000-0003-1582-6641

A Generalizable Hybrid Search Framework for Optimizing Expensive Design Problems using Surrogate Models

Abstract:

Experimental optimization of physical and biological processes is a difficult task. To address this, sequential surrogate models combined with search algorithms have been employed to solve nonlinear high dimensional design problems with expensive objective function evaluations. In this article, a hybrid surrogate framework was built to learn optimal parameters of a diverse set of simulated design problems meant to represent real world physical and biological processes in both dimensionality and nonlinearity. The framework uses a hybrid radial basis function / genetic algorithm with dynamic coordinate search response, utilizing the strengths of both algorithms. The new hybrid method performs better or as good as its constituent algorithms in 19 of 20 high dimensional test functions, making it a very practical surrogate framework for a wide variety of optimization design problems. Experiments also show that the hybrid framework can be improved even more when optimizing processes with simulated noise.

Keywords: hybrid surrogate, meta-modeling, DYCORS, experimental optimization, radial basis function

1. Introduction

The design and optimization of engineering systems often requires the use of high-fidelity simulations and/or experiments. These black box systems often have nonlinear responses, high dimensionality, and have many local optima. This makes these systems costly and time consuming to model, understand, and optimize when simulations take hours or experiments performed in the lab require extensive time and resources.

The first attempt to improve over simple optimization methods, such as ‘one-factor-at-a-time’ and random experiments, was through the field of Design of Experiments (DOE). Techniques in DOE have been adapted to many computational (Giunta, Wojtkiewicz, and Eldred 2003) and experimental fields (Weuster-Botz 2000;

Singh et al. 2017) in order to reduce the number of samples needed for system design and optimization. These methods often involve performing experiments or simulations at the vertices of the design space hypercube (the boundaries of the system). Full-Factorial Designs are arguably the simplest to implement, where data is collected at all potential combinations of parameters p for all levels l requiring l^p samples in total. Even when $l = 2$ (for 'high' and 'low' levels of the design space) the number of experiments or simulations quickly becomes infeasible so Fractional-Factorial Designs using l^{p-k} experiments for k 'generators' are often used to reduce computational or experimental burden, at the expense of convolving interaction effects of various parameters on the response. Other techniques include designing experiments or simulations using Latin Hypercubes, which prioritize space-filling properties, often used in the construction of posteriors in Bayesian statistics or models in computational design and inference (Cioppa and Lucas 2007; Pholdee and Bureerat 2015). This is a popular method of initializing databases for statistical inference and optimization problems. DOE techniques are also often combined with Response Surface Methodologies (RSM) such as a polynomial model, to iteratively move the sampling location, improve model fidelity as more data is collected (Saval, Pablos, and Sanchez 1993), and focus experiments in regions of interest. Stochastic optimization methods such as Genetic Algorithms (Haupt and Haupt, n.d.), Particle Swarm Optimization (Agrawal et al. 2008), and Differential Evolution (Storn and Price 1997) have also been used to explore design spaces and perform optimization on both simulated and experimental datasets (Bapat and Wangikar 2004; Havel et al. 2006; Franco-Lara, Link, and Weuster-Botz 2006), often requiring fewer experiments than traditional DOE-RSM techniques.

For Peer Review Only

The quickly developing field of surrogate optimization attempts to leverage robust response surface modeling techniques, such as radial basis functions (RBF) (Regis and Shoemaker 2012) or Kriging models (Wang and Shan 2007), to optimize nonlinear systems. These methods often employ a stochastic (G. Zhang, Olsen, and Block 2007), uncertainty-based (Jones, Schonlau, and W. J. Welch 1998), or Bayesian (Kathryn Chaloner 1996) search algorithm to intelligently select new sample points to query for experimentation or simulation. Because these stochastic optimizers require many function evaluations, and the design problems we are focusing on are expensive to evaluate, the surrogate model provides a cheap approximation of the true system. Due to the variety of modeling techniques and search algorithms available, hybrid algorithms, which attempt to leverage the strengths of various surrogate modeling techniques, have proliferated (Gu, Li, and Dong 2012; J. Zhang, Chowdhury, and Messac 2012). These hybrid approaches usually involve taking ensembles of various surrogate model predictions. New queries are then conducted at points weighted in favor of regions/surrogates with low sample variance or optimal response values. The drawback of many of these algorithms is that they are not always generalizable to design problems of diverse dimensionality and nonlinearity.

We present a surrogate optimization algorithm which uses an evolving RBF model and hybrid search algorithm. This search algorithm selects half of its query points using a Euclidean distance metric truncated to provide diversity in suggested query points. This is based on a *Neural Network Genetic Algorithm* (NNGA) developed for bioprocess optimization (G. Zhang and Block 2009b), which has been shown to be more efficient than traditional DOE-RSM methods. The other half of the query points are selected using a *Dynamic Coordinate Search for Response Surface Methods* algorithm (DYCORS) based on work developed for computationally expensive

simulation (Regis and Shoemaker 2012). DYCORS has been shown to perform better than a variety of popular surrogate optimization techniques. The performance of the NNGA-DYCORS hybrid algorithm is tested against NNGA and DYCORS separately. Further evaluation is performed to probe potentially useful extensions of the hybrid algorithm (i) to address simulated experimental noise, (ii) to improve algorithm convergence over time, and (iii) to address cases in which certain groups of parameters have a greater influence on the response values than others.

2. Methods

2.1 RBF Surrogate Model

The surrogate model used is the RBF interpolation model (Powell 1990), which provides a cheap approximation of the process. A cubic RBF $\phi(x) = r^3$ with a linear tail $p(x)$ is used to map input data $X \in \mathbb{R}^{nxp}$ to output data $Y \in \mathbb{R}^{nx1}$ given a number of datapoints n with input dimensionality p. The form of the RBF interpolation $s(x)$ is shown (Equation 1).

$$s_n(x) = \Sigma_{i=1}^{n} \lambda_i \phi\left(\left|\left|x - x_i\right|\right|\right) + p(x) \quad (1)$$

Substituting $\phi(x)$ and $p(x)$ gives Equation 2,

$$s_n(x) = \Sigma_{i=1}^{n} \lambda_i \left(\left|\left|x - x_i\right|\right|\right)^3 + c_0 + \Sigma_{j=1}^{p} c_j x_j \quad (2)$$

where $|| x - x_i ||$ is the Euclidean norm of a given point x and all RBF nodes x_i. The number of nodes in an RBF model is often tuned to give low bias (more nodes) or low variance (fewer nodes). For training, the coefficients of the RBF $\lambda \in \mathbb{R}^{nx1}$ and the linear tail $c \in \mathbb{R}^{(d+1)\,x\,1}$ (for a d dimensional design problem) are determined by solving to the following system of linear equations shown in Equation 3.

$$\begin{pmatrix} \Phi & P \\ P^T & 0_{(p+1)\,x\,(p+1)} \end{pmatrix} \begin{pmatrix} \lambda \\ c \end{pmatrix} = \begin{pmatrix} Y \\ 0_{d+1} \end{pmatrix} \quad (3)$$

The matrix $\Phi \in \mathbb{R}^{nxn}$ consists of components $\Phi_{ij} = \phi\left(\left|\left|x_i - x_j\right|\right|\right)$. The matrix $P \in \mathbb{R}^{nx(d+1)}$ is comprised of the rows of $[1, x_i^T]$. The output of datapoint i is y_i contained in Y. The coefficient vector can be inverted using singular value decomposition, solving the linear transformation for input data X and output data Y. Modifying the equation to exclude the linear terms requires solving $\Phi\lambda = Y$, which will be used in Section 3.2.

2.2 NNGA

The NNGA algorithm is based on an RBF assisted genetic algorithm. The NNGA uses an RBF model to suggest points that are *close to* but not directly *on top of* optima using a truncated genetic algorithm (TGA). One advantage genetic algorithms have over gradient-based methods is that their randomness allows them to efficiently explore both global and local regions of optimality. This makes them very attractive for an optimization framework attempting to look for global optima while facing uncertainty associated with a sparsely explored parameter space, and thus untrustworthy RBF models. This framework is shown in Figure 1a and the TGA is illustrated in Figure 2.

[Figure 1]

First, a database of inputs X and outputs Y of N_o total queries is collected (often through a DOE, random queries, or Latin Hypercube design). An RBF model is constructed using the training regime discussed in Section 2.1. Next a TGA is ran using a randomly initiated population of potential query points with the goal of minimizing the RBF predicted output. In each iteration of the TGA, queries expected to perform the

For Peer Review Only

best survive a culling process and have their information propagated into the next iteration by a pairing, crossover, and random mutation step. After each iteration, the best predicted query is recorded. When the average normalized Euclidean distance between the TGA's current predicted best query and its next $N-1$ predicted best queries, $d_{av,norm}$, is less than or equal to the critical distance parameter $CD = 0.2$, the TGA is considered to be converged and submits this list of N best points for potential querying (or if the maximum number of iterations has been reached). This TGA is run a total of $k_{max} = 4$ times, and its query selections from all rounds of TGA are clustered down to a final averaged query list of size N using *K-Means Clustering*. The final list is queried to give the next set of data for simulation or experiments. The number of queries per batch N, total number of batches b_{max}, and critical distance parameter CD, which controls the degree of truncation, are set by the user.

[Figure 2]

2.3 DYCORS

The DYCORS generates a large list of potential query points based on gaussian perturbations of the current best point in the training dataset. It is dynamic because, as the training dataset increases in size, the number of parameters perturbed decreases. In this manner, DYCORS narrows its search of the parameter space over time. The DYCORS process is shown in Figure 1b and the parameters used in the algorithm are presented in the discussion below.

First a database with inputs X and outputs Y of N_o total queries is collected, and an RBF model constructed. Next, the best point in the current database x^* is selected and perturbed by a truncated multivariate normal distribution (Botev 2017), bounded by

the parameter's bounds $[\Delta_{low}, \Delta_{high}]$ and using standard deviation $l_b * \Delta_j$ for each parameter j and current batch step size l_b. This is repeated on $d = \min\{100p, 5000\}$ copies of x^* and is equivalent to taking the best solution and looking in the general $l_b * \Delta_j$ region around them for the next points to query. The perturbation appears in the form of Equation 4 for a parameter j to be perturbed for a given i copy of x^*.

$$x_{i,j} = x_{i,j} + \mathcal{N}(0, l_b * \Delta_j) \tag{4}$$

DYCORS is modulated by the *Step Size Selection Algorithm* shown in Figure 1d which counts consecutive successful $\mathcal{C}_{succ}$ and failed $\mathcal{C}_{fail}$ batches of queries and either doubles (if $\mathcal{C}_{succ} \geq \mathcal{T}_{succ} = 3$) or halves (if $\mathcal{C}_{fail} \geq \mathcal{T}_{fail} = \max\{p, 5\}$) the step size l_b for the next batch based on thresholds $\mathcal{T}$. This heuristic is employed based on the logic that, if numerus consecutive failures to improve are seen, a minimum parameter set has likely been reached. Thus, the search space is narrowed. In addition to altering l_b over time (with an initial $l_o = 0.2$ and minimum $l_{min} = 0.2(0.5)^6$), DYCORS also reduces the probability that a point $x_{i,j}$ will be perturbed by Equation 5 which is dependent on the current number of queries in the database N_b. This has the effect of 'narrowing down' the amount of perturbations per batch as time goes on. After the perturbations are made and step size l_b is updated, the N best perturbations of x^* are selected to be queried. The process is shown in detail in Figure 1b. The primary way that this implementation of DYCORS differs from the original work is that the N best perturbations of x^* are selected for querying rather than the single best perturbation. For a given N_{max} (total amount of queries) this makes this implementation of DYCORS less efficient, but allows for multiple queries to be generated at once, and thus parallel computations / experiments to be made.

$$P(N_b) = \min\left\{\frac{20}{p}, 1\right\} * \left(1 - \frac{\ln(N_b - N_o + 1)}{\ln(N_{max} - N_o)}\right) \quad (5)$$

2.4 NNGA-DYCORS Hybrid

To combine the NNGA and DYCORS surrogate optimization algorithms, we simply run them in parallel with a shared dataset $\{X, Y\}$ and RBF model. This is shown by a flowchart in Figure 1c. By having access to the same data, the two algorithms can make different conclusions about new optimal queries. To form the new dataset, the suggested queries are combined from each of the constituent algorithms, and the new queries are conducted. The user can determine how many queries each algorithm suggests each batch. In this article, the NNGA and DYCORS arms of the hybrid find the same number of optimal queries.

2.5 Test Functions and Algorithm Assessment

To test the ability of these algorithms to learn arbitrary complex relationships between X and Y and find the minima of the resulting surfaces, optimization is done on several test functions as shown in Table 1.

[Table 1]

Simulations were performed on 10-D and 50-D dimensional variants of each test function to simulate low and high dimensional optimization problems. For each evaluation, all algorithms were run 15 times with a randomly selected initial database of size $N_o = 50$ and $N = 10$ queries per batch (with $N_{NNGA} = 5$ and $N_{DYCORS} = 5$ queries per batch from NNGA and DYCORS respectively in the case of the hybrid NNGA-DYCORS algorithm). The total number of batches was $b_{max} = 15$ which made for a total of $N_{max} = 200$ simulated experimental datapoints as the size of the final dataset.

To evaluate the optimization algorithms, learning curves were plotted to demonstrate the average optimal (minimum) output of each batch of queries including error bars that indicate the standard deviation in the minimum output of each batch for the 15 runs. The mean, median, minimum, and standard deviations of the final batch of queries is shown in Tables S1 and S2.

2.6 Software and Hardware

Hardware used: Dell Precision 5820 Tower, Intel Xeon W-2145 DDR4-2666 Processor (3.7 GHz), 32 GB Memory. Software used: MATLAB R2019a with Bioinformatics Package.

3. Results

3.1 The Hybrid Framework versus Constituent Algorithms

The NNGA-DYCORS algorithm was tested against its constituent algorithms, NNGA and DYCORS individually. Examining the performance of the constituent algorithms (Figure 3 and Table S1 and S2), the NNGA often outperforms the DYCORS in the higher 50-D problems, while the DYCORS outperforms NNGA in the lower 10-D problems. This was the case both over time (Figure 3) and at the final optimal query points (Table S1 and S2). Given these differences in performance, it stands to reason that a hybrid approach would provide a sensible route to a more robust algorithm that could be used on a wider variety of dimensions. As seen in Figure 3, the hybrid NNGA-DYCORS often outperforms or performs similar the next best constituent algorithm in each experiment. This is reinforced by the data in Table S1 and S2, where the final optima of the hybrid NNGA-DYCORS is less than or equal to the final optima of the next best constituent algorithm in 19 of 20 experiments (all but the Michalewicz 50-D).

An optimum may be considered "better" if its upper bound (its mean plus standard deviation) is less than the mean of another algorithms optimum. While this is a rough approximation of the comparative performance of the algorithm, it strongly indicates that the NNGA-DYCORS is robust on a wild variety of problem sets and dimensions. In intermediate cases (those between 10-D and 50-D), the NNGA-DYCORS continued to outperform or perform as well as its most competitive constituent algorithm (data not shown), showing its usefulness in design optimization problems where it is not obvious a priori what dimensionality counts as 'high' and 'low'.

[Figure 3]

3.2 Algorithm Performance in the Presence of Simulated Experimental Noise

To test the effect of random noise on the ability of the surrogate optimization algorithms to find optimal parameters, random noise e was added to each function evaluation to simulate experimental noise.

$$y = y + \mathcal{N}(0, e * y) \tag{6}$$

It is common practice, especially in noisy and data-sparse models, to improve the out-of-sample generalizability by model selection procedures such as cross-validation to avoid overfitting. To address this, a hyperparameter optimization loop for the number of nodes n_{nodes} in the RBF model was added to the NNGA-DYCORS algorithm, where cross-validation over the database was used to select the optimal n_{nodes}. In this case we deliberately trade higher bias for lower variance to reduce overfitting. As can be seen in Figure 4, application of a node optimization scheme either improved or did not degrade the learner's performance over the regular scheme (where $n_{nodes} = N_b$) for $e = 0.2$ (20% simulated response noise) in all but one test function. It

should be noted that in these experiments, the linear tail of the RBF was excluded, so Equation 3 was modified to be $\Phi\lambda = Y$.

[Figure 4]

3.3 Evaluating the Effect of Convergence Parameters on Algorithm Performance

Both NNGA and DYCORS have adjustable convergence parameters that control their design space exploration/exploitation tradeoff. In other words, both algorithms have a means of avoiding premature convergence to local minima, as predicted by an early (i.e. less accurate and generalizable) surrogate approximation. Here we test the effect of changing these internal search parameters l_b (DYCORS) and CD (NNGA). For DYCORS, this has already been suggested using a time-varying strategy (Jiang, Shoemaker, and Liu 2018) for current database size N_b. In this method, the step size is dynamically recalculated as $l_{b+1} = C_b l_b$.

$$\theta(N_b) = 2\left(1 - \frac{\ln(N_b - N_o + 1)}{\ln(N_{max} + N_o)}\right) \tag{7}$$

$$C_b = \begin{Bmatrix} 1 & \theta \geq 1 \\ \theta & 0.5 < \theta < 1 \\ 0.5 & \theta \leq 0.5 \end{Bmatrix} \tag{8}$$

For NNGA, if one defines a maximum and minimum critical distance parameter $CD_1 = 0.2$ and $CD_2 = 0.05$ respectively, then CD can be changed linearly over time using the following formula,

$$CD(N_b) = \left(\frac{CD_2 - CD_1}{N_{max} - N_o}\right) * (N_b - N_o) + CD_1 \tag{9}$$

where N_{max} and N_o are the maximum and initial database sizes respectively. The result of implementing this dynamic parameter approach (Figure 5) was that the hybrid learner did not have substantially better performance over the regular hybrid learner. This suggests that the internal search parameters l_b and CD do not need to be substantially altered over the course of optimization.

[Figure 5]

3.4 Evaluating the Effects of a Parameter Subset Selection Algorithm

In engineering systems, certain parameters influence the response more significantly than others, and often few parameters matter at all. To simulate this variable response sensitivity while maintaining the nonlinearity and dimensionality of the test problems, an 'sensitivity vector' γ was used to scale the test problems. This vector scales each problem as $x_{scaled,j} = \gamma_j x_j$ for parameter j so that 20% of the parameters are scaled up by $\gamma = 2$, 30% are unscaled, 20% are scaled down by $\gamma = 0.5$, and 30% are neglected in the deterministic function. An example of the implementation of γ and a scaled 2-D Ackley Function is shown in Figure 6.

$$\gamma = [2,2, \ldots 1,1, \ldots 0.5,0.5, \ldots 0,0]$$

[Figure 6]

Previous work in applying a decision tree-based subset selection strategy to the NNGA algorithm (G. Zhang and Block 2009a) reduced the number of queries needed in optimization. To explore this further, an RBF-based subset selection strategy was developed. After $b = 7$ batches of queries (roughly halfway through the entire set of queries), p RBF models are trained with $q = 1 \ldots p$ parameters being "dropped out". For each neglected parameter q, a cross-validated average correlation coefficient R_{av}^2 is

found using a separate hold-out-set of data. The most 'important' parameters should have the lowest R_{av}^2 assuming the RBF model is robust for the database. In experiments using this technique, the DYCORS algorithm selects the most important parameters and only uses that subset in the coordinate-wise perturbation, while the NNGA operates normally. The result was that, while this subset selection method was able to speed up learning in some cases (Ackley Function), it was not able to do so consistently (Michalewicz Function).

[Figure 7]

4. Discussion

There are a seemingly infinite number of modeling techniques, search optimization algorithms, and initialization/infill strategies in the literature to facilitate optimizing expensive objective functions. However, the characteristics of the experimental system and design space are never really known a priori, so having an algorithm that is more efficient than traditional methods and able to work with a wide variety of problems is advantageous. Therefore, the goal of this article was to develop a surrogate optimization framework that could be successfully applied to test problems with a wide range of dimensionality and degrees of nonlinearity. The NNGA-DYCORS algorithm runs two surrogate optimization algorithms in parallel. The NNGA uses a Euclidean distance-based metric to truncate a genetic algorithm, whose best members are K-means cluster distilled into a final query list. This acts as a global optimization process because the internal genetic algorithm searches over the entire design space. The DYCORS algorithm perturbs the best previous queries using a dynamic Gaussian distribution, where the perturbations are adjusted based on cumulative success and the total number of queries in the database. Thus, DYCORS acts as a local search method in the region

For Peer Review Only

defined by a gaussian centered at its best queries. Both arms of the hybrid algorithm use an RBF for prediction.

The result was that the NNGA-DYCORS hybrid algorithm was statistically equal to or outperformed its constituent algorithms in the 19 of 20 test problems. This demonstrates the robustness of the NNGA-DYCORS, as it performs as a 'best-case-scenario' on a variety of test problem dimensions and shapes. This is important because, in real experimental problems, one does not know the shape of the surface a priori, highlighting the utility of a generalizable optimization framework such as the NNGA-DYCORS. Additionally, it is never clear what constitutes a 'high' and 'low' dimensionality design problem, so an algorithm that performs well in arbitrary dimensions should have large practical value. The DYCORS algorithm was already shown to be competitive compared to other heuristics (Regis and Shoemaker 2012), and the NNGA was demonstrated to be significantly more efficient than traditional experimental optimization methods (G. Zhang and Block 2009b). It stands to reason that this hybrid framework should extend the usefulness of both algorithms to test problems of arbitrary dimensionality and degree of nonlinearity.

Using a node optimization scheme to reduce model variance during query selection improves hybrid algorithm performance, especially for noisy surfaces (as could be the case in experimental situations). Practitioners should therefore consider built-in regularization to avoid overfitting of the data when dealing with expensive, data-sparse, and noisy systems. Optimizing the number of nodes was specific to this RBF variant, but the optimization loop in Section 3.2 could be applied to any model hyperparameter. In the next set of experiments, our method of making the NNGA-DYCORS convergence parameters dynamic during query selection did not improve performance. This indicates (i) it may not be fruitful to pursue extensive algorithm

parameter adjustments/heuristics for this algorithm and (ii) there is little sensitivity in the selection of algorithm convergence parameters on the outcome, unlike the results in previous articles (Jiang, Shoemaker, and Liu 2018; G. Zhang and Block 2009a). Finally, to mimic typical engineering scenarios where response sensitivity varies with the inputs, the test functions were scaled with a 'sensitivity' vector. A subset selection strategy was unable to consistently improve on the regular NNGA-DYCORS performance by 'focusing' the coordinate search on the most sensitive sets of parameters. This may be because the RBF does not adequately model a given test function so does not correctly identify the most important parameters in the database, or the coordinate search method does not properly exploit the narrowed parameter space. Generically, it may be useful to reduce the dimensionality of the parameter space, but the strategy of doing so using model adherence "drop-out" experiments was not uniformly successful.

This article demonstrates that the NNGA-DYCORS hybrid learning algorithm outperforms its constituent algorithms in the important criteria of robustness and generalizability to different kinds of problems. Thus, this algorithm can be applied to a wide variety of physical and biological design optimization problems with a degree of assurance that parameter estimates will be optimal while minimizing necessary resources. Additionally, as this hybrid is both robust and highly generalizable to many types of design problems, it should be useful for practitioners who are not experts in surrogate optimization methods, and work on a variety of problems of diverse complexity.

Acknowledgements

This work was supported by New Harvest Inc. Graduate Fellowship Program under Grant A19-4213 and the Ernest Gallo Endowed Chair in Viticulture and Enology.

Disclosure Statement

No potential conflict of interest was reported by the authors.

References

Agrawal, Shubham, Yogest Dashora, Manoj Tiwari, and Young Jun Son. 2008. "Interactive Particle Swarm: A Pareto-Adaptive Metaheuristic to Multiobjective Optimization." *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans* 38 (2): 258–77. https://doi.org/10.1109/TSMCA.2007.914767.

Bapat, Prashant M., and Pramod P. Wangikar. 2004. "Optimization of Rifamycin B Fermentation in Shake Flasks Via a Machine-Learning-Based Approach." *Biotechnology and Bioengineering* 86 (2): 201–8. https://doi.org/10.1002/bit.20056.

Botev, Z. I. 2017. "The Normal Law under Linear Restrictions: Simulation and Estimation via Minimax Tilting." *Journal of the Royal Statistical Society. Series B: Statistical Methodology* 79 (1): 125–48. https://doi.org/10.1111/rssb.12162.

Cioppa, Thomas M., and Thomas W. Lucas. 2007. "Efficient Nearly Orthogonal and Space-Filling Latin Hypercubes." *Technometrics* 49 (1): 45–55. https://doi.org/10.1198/004017006000000453.

Franco-Lara, Ezequiel, Hannes Link, and Dirk Weuster-Botz. 2006. "Evaluation of Artificial Neural Networks for Modelling and Optimization of Medium Composition with a Genetic Algorithm." *Process Biochemistry* 41 (10): 2200–2206. https://doi.org/10.1016/j.procbio.2006.06.024.

Giunta, Anthony A., Steven F. Wojtkiewicz, and Michael S. Eldred. 2003. "Overview of Modern Design of Experiments Methods for Computational Simulations." In *41st Aerospace Sciences Meeting and Exhibit*, 1–17. https://doi.org/10.2514/6.2003-649.

Gu, J., G. Y. Li, and Z. Dong. 2012. "Hybrid and Adaptive Meta-Model-Based Global Optimization." *Engineering Optimization* 44: 87–104. https://doi.org/10.1080/0305215X.2011.564768.

Haupt, Randy L., and Sue Ellen Haupt. n.d. *Practical Genetic Algorithms*.

Havel, Jan, Hannes Link, Michael Hofinger, Ezequiel Franco-Lara, and Dirk Weuster-Botz. 2006. "Comparison of Genetic Algorithms for Experimental Multi-Objective Optimization on the Example of Medium Design for Cyanobacteria." *Biotechnology Journal* 1 (5): 549–55. https://doi.org/10.1002/biot.200500052.

Jiang, P., C. A. Shoemaker, and X. Liu. 2018. "Time-Varying Hyperparameter Strategies for Radial Basis Function Surrogate-Based Global Optimization Algorithm." *IEEE International Conference on Industrial Engineering and Engineering Management* 2017-Decem: 984–88. https://doi.org/10.1109/IEEM.2017.8290039.

Jones, D. R., M. Schonlau, and W. J. Welch. 1998. "Efficient Global Optimization of Expensive Black-Box Functions." *Journal of Global Optimization* 13: 455–92.

Kathryn Chaloner, Isabella Verdinelli. 1996. "Bayesian Experimental Design Review." *Statistical Science* 10 (3): 273–304.

Pholdee, Nantiwat, and Sujin Bureerat. 2015. "An Efficient Optimum Latin Hypercube Sampling Technique Based on Sequencing Optimisation Using Simulated Annealing." *International Journal of Systems Science* 46 (10): 1780–89. https://doi.org/10.1080/00207721.2013.835003.

Powell, Michael. 1990. *The Theory of Radial Basis Function Approximation in 1990*.

Regis, Rommel G., and Christine A. Shoemaker. 2012. "Combining Radial Basis Function Surrogates and Dynamic Coordinate Search in High-Dimensional Expensive Black-Box Optimization." *Engineering Optimization* 45 (5): 529–55. https://doi.org/10.1080/0305215X.2012.687731.

Saval, S., L. Pablos, and S. Sanchez. 1993. "Optimization of a Culture Medium for Streptomycin Production Using Response-Surface Methodology." *Bioresource Technology* 43 (1): 19–25. https://doi.org/10.1016/0960-8524(93)90077-O.

Singh, Vineeta, Shafiul Haque, Ram Niwas, Akansha Srivastava, Mukesh Pasupuleti, and C. K.M. Tripathi. 2017. "Strategies for Fermentation Medium Optimization: An in-Depth Review." *Frontiers in Microbiology* 7: 1–16. https://doi.org/10.3389/fmicb.2016.02087.

Storn, Rainer, and Kenneth Price. 1997. "Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces." *Journal of*

Global Optimization 11 (3): 341–59. https://doi.org/10.1071/AP09004.

Wang, G. Gary, and S. Shan. 2007. "Review of Metamodeling Techniques in Support of Engineering Design Optimization." *Journal of Mechanical Design, Transactions of the ASME* 129 (4): 370–80. https://doi.org/10.1115/1.2429697.

Weuster-Botz, Dirk. 2000. "Experimental Design for Fermentation Media Development: Statistical Design or Global Random Search?" *Journal of Bioscience and Bioengineering* 90 (5): 473–83. https://doi.org/10.1016/S1389-1723(01)80027-X.

Zhang, Guiying, and David E. Block. 2009a. "Integration of Data Mining Into a Nonlinear Experimental Design Approach for Improved Performance." *AIChE Journal* 55 (11): 3017–21. https://doi.org/10.1002/aic.

———. 2009b. "Using Highly Efficient Nonlinear Experimental Design Methods for Optimization of Lactococcus Lactis Fermentation in Chemically Defined Media." *Biotechnology Progress* 25 (6): 1587–97. https://doi.org/10.1002/btpr.277.

Zhang, Guiying, Matthew Olsen, and David E. Block. 2007. "New Experimental Design Method for Highly Nonlinear and Dimensional Processes." *AIChE Journal* 56 (8): 2013–25. https://doi.org/10.1002/aic.

Zhang, Jie, Souma Chowdhury, and Achille Messac. 2012. "An Adaptive Hybrid Surrogate Model." *Structural and Multidisciplinary Optimization* 46: 223–38. https://doi.org/10.1007/s00158-012-0764-x.

Supplemental Tables

		(Section 3.1)			(Section 3.2)		(Section 3.3)		(Section 3.4)	
		NNGA	**DYCORS**	**NNGA-DYCORS**	**Node Opt**	**Regular**	**Dynamic**	**Regular**	**Subset Selector**	**Regular**
Ackley 10-D	Mean	-13.94	-18.27	**-18.32**	-17.95	-17.20	-18.65	-18.32	-22.16	-19.06
	Median	-14.15	-18.94	-18.61	-18.19	-17.41	-18.79	-18.61	-22.51	-19.32
	Min	-15.21	-19.96	-20.02	-21.12	-21.36	-20.12	-20.02	-22.72	-19.90
	St. Dev	0.85	1.56	1.23	1.71	3.07	0.97	1.23	0.70	0.74
Ackley 50-D	Mean	-12.39	-8.39	**-13.28**	-11.69	-10.99	-13.19	-13.28	-18.04	-13.50
	Median	-12.22	-8.50	-13.35	-11.54	-10.64	-13.28	-13.35	-17.56	-13.39
	Min	-13.62	-9.75	-14.06	-13.55	-15.69	-14.19	-14.06	-21.44	-14.64
	St. Dev	0.55	0.95	0.50	0.87	1.41	0.49	0.50	1.74	0.56
Rastrigin 10-D	Mean	-33.92	-57.33	**-50.71**	-71.91	-49.79	-49.52	-50.71	-92.48	-77.94
	Median	-29.94	-61.37	-50.71	-71.50	-47.29	-52.19	-50.71	-94.88	-78.08
	Min	-49.29	-82.89	-67.66	-107.03	-81.55	-68.68	-67.66	-100.00	-91.32
	St. Dev	8.73	16.13	13.64	14.96	14.17	11.94	13.64	6.79	7.44
Rastrigin 50-D	Mean	-29.59	-24.12	**-74.51**	-105.19	-54.98	-73.01	-74.51	-412.62	-191.60
	Median	-25.22	-26.93	-66.56	-103.66	-46.03	-67.93	-66.56	-426.23	-191.22
	Min	-79.98	-113.43	-132.07	-152.68	-150.11	-127.28	-132.07	-456.88	-270.62
	St. Dev	21.66	45.37	27.63	28.13	39.83	31.12	27.63	36.60	28.65
Griewank 10-D	Mean	15.69	1.17	**1.12**	3.05	3.03	1.15	1.12	0.41	1.44
	Median	15.57	1.18	1.10	2.65	2.79	1.14	1.10	0.28	1.38
	Min	10.72	1.02	1.03	1.52	1.02	1.04	1.03	0.00	0.97
	St. Dev	2.56	0.08	0.06	1.33	1.63	0.12	0.06	0.43	0.35
Griewank 50-D	Mean	89.89	313.22	**42.00**	104.51	179.87	40.60	42.00	16.97	72.46
	Median	93.10	319.37	41.23	102.41	172.40	38.85	41.23	12.92	69.59
	Min	56.83	231.28	29.23	77.00	71.24	28.55	29.23	1.40	54.64
	St. Dev	15.64	45.69	8.67	16.69	70.25	8.21	8.67	16.08	13.40
Levy 10-D	Mean	1.32	1.87	**0.35**	0.42	0.96	0.45	0.35	0.67	0.50
	Median	1.25	2.00	0.24	0.34	0.70	0.22	0.24	0.68	0.48
	Min	0.82	0.14	0.05	0.20	0.18	0.07	0.05	0.49	0.20
	St. Dev	0.39	1.37	0.28	0.23	0.81	0.42	0.28	0.10	0.15
Levy 50-D	Mean	8.79	35.65	**6.47**	6.06	26.34	6.17	6.47	6.57	9.64
	Median	8.65	34.53	6.16	5.74	25.80	5.79	6.16	6.39	9.39
	Min	6.83	21.76	3.74	4.11	14.67	4.62	3.74	5.23	7.03
	St. Dev	1.56	7.13	1.27	1.30	8.55	1.44	1.27	1.27	2.07
Michalewicz 10-D	Mean	-3.83	-5.32	**-4.67**	-4.03	-5.28	-4.76	-4.67	-2.89	-3.27
	Median	-3.82	-5.31	-4.62	-4.06	-5.27	-4.54	-4.62	-2.93	-3.30
	Min	-4.47	-6.29	-6.37	-5.01	-6.77	-5.98	-6.37	-3.46	-4.52
	St. Dev	0.27	0.62	0.69	0.61	0.77	0.69	0.69	0.48	0.55
Michalewicz 50-D	Mean	-12.45	-16.29	-14.46	-15.91	-15.62	-15.17	-14.46	-8.83	-9.76
	Median	-12.29	-16.63	-14.26	-15.34	-15.22	-15.24	-14.26	-8.70	-9.80
	Min	-14.64	-18.46	-16.32	-21.16	-19.42	-16.73	-16.32	-12.50	-11.00
	St. Dev	0.80	1.42	0.82	2.13	1.74	1.03	0.82	1.21	0.98

Table S1. Final Algorithm Performance Part I. Experiments from Section 3.1 where NNGA-DYCORS performed better than or as well as the next best constituent algorithm by one standard deviation have their mean bolded. Variants of NNGA-DYCORS are tested in Sections 3.2 – 3.4. All data shown are from the final batch of experiments for all algorithms.

		(Section 3.1)			(Section 3.2)		(Section 3.3)		(Section 3.4)	
		NNGA	**DYCORS**	**NNGA-DYCORS**	**Node Opt**	**Regular**	**Dynamic**	**Regular**	**Subset Selector**	**Regular**
Rosenbrock 10-D	Mean	1549.17	371.38	**314.70**	560.34	406.12	307.57	314.70	523.17	1634.10
	Median	1603.75	264.02	259.76	504.14	315.14	309.88	259.76	199.76	1297.98
	Min	514.90	139.76	113.65	230.31	89.97	77.47	113.65	11.15	161.31
	St. Dev	561.26	344.11	168.60	235.90	283.75	145.75	168.60	763.77	1748.49
Rosenbrock 50-D	Mean	75690.36	242306.98	**38456.52**	98999.76	73734.80	35725.07	38456.52	51608.16	102779.11
	Median	74088.12	248151.64	37217.99	89868.23	74216.32	32142.01	37217.99	48846.00	102600.79
	Min	50297.29	131056.62	21309.17	47744.74	37723.98	16783.97	21309.17	3074.55	64081.57
	St. Dev	12389.95	87955.49	15421.30	36205.98	22396.93	10067.54	15421.30	36082.25	29036.23
Dixon-Price 10-D	Mean	112.35	-3.63	**-8.95**	7.91	11.81	-9.61	-8.95	-9.04	-11.95
	Median	118.76	-6.89	-8.95	6.31	12.69	-9.67	-8.95	-9.65	-12.44
	Min	58.20	-9.06	-9.96	-2.47	-8.08	-10.52	-9.96	-10.95	-12.99
	St. Dev	28.35	6.26	0.86	6.96	14.48	0.72	0.86	1.47	1.29
Dixon-Price 50-D	Mean	4821.50	18991.14	**2662.18**	2006.23	6529.50	2239.03	2662.18	162.90	1294.59
	Median	4923.68	18334.97	2720.11	2060.22	6449.41	2187.08	2720.11	94.48	1267.19
	Min	3656.99	14642.27	2060.92	1108.66	4098.76	1414.94	2060.92	-2.49	732.76
	St. Dev	695.21	3066.97	453.33	779.12	1951.07	478.78	453.33	157.33	281.06
Styblinski-Tang 10-D	Mean	-298.18	-326.69	**-333.67**	-378.58	-361.62	-329.51	-333.67	-192.14	-228.76
	Median	-298.32	-320.13	-333.31	-360.75	-361.14	-327.43	-333.31	-191.82	-229.24
	Min	-342.77	-361.26	-378.67	-461.34	-430.62	-374.52	-378.67	-235.31	-253.52
	St. Dev	22.28	19.57	22.60	44.76	31.31	26.62	22.60	19.08	20.47
Styblinski-Tang 50-D	Mean	-968.58	-1147.15	**-1081.74**	-1240.77	-1234.98	-1071.24	-1081.74	-537.72	-647.90
	Median	-969.09	-1154.69	-1103.16	-1221.53	-1256.26	-1074.18	-1103.16	-522.02	-655.64
	Min	-1048.73	-1275.31	-1204.57	-1431.93	-1455.61	-1148.52	-1204.57	-669.89	-818.40
	St. Dev	50.06	68.43	66.59	108.97	106.62	39.64	66.59	62.99	67.06
Sphere 10-D	Mean	4.09	0.06	**0.04**	0.76	1.05	0.03	0.04	0.01	0.12
	Median	4.46	0.04	0.04	0.64	0.87	0.02	0.04	0.01	0.07
	Min	1.91	0.01	0.02	0.28	0.13	0.01	0.02	0.00	0.02
	St. Dev	1.14	0.05	0.02	0.61	0.97	0.02	0.02	0.02	0.11
Sphere 50-D	Mean	23.44	84.55	**13.44**	11.96	30.45	12.36	13.44	5.10	17.84
	Median	24.46	84.31	12.75	12.24	31.75	12.85	12.75	4.39	16.58
	Min	15.06	56.38	10.69	5.44	17.66	8.95	10.69	0.64	10.45
	St. Dev	3.39	15.24	2.53	3.04	11.59	1.79	2.53	3.67	5.24
Zakharov 10-D	Mean	115.15	90.29	**92.33**	54.99	85.25	101.44	92.33	25.16	45.66
	Median	113.79	76.68	79.57	53.80	87.63	94.76	79.57	16.15	41.29
	Min	61.37	43.08	55.38	26.94	23.27	78.55	55.38	0.00	13.27
	St. Dev	33.86	33.90	31.78	13.17	29.66	26.12	31.78	22.56	24.37
Zakharov 50-D	Mean	2991494.52	946.09	**4997.76**	2273.53	20975.65	2189.24	4997.76	1082.03	1014.78
	Median	1008940.58	875.85	1199.42	741.27	1916.37	943.91	1199.42	992.05	791.85
	Min	844.41	694.97	754.38	427.68	474.36	673.69	754.38	395.69	424.28
	St. Dev	4392168.34	362.74	8991.23	5131.20	36463.61	4457.90	8991.23	711.68	588.30

Table S2. Final Algorithm Performance Part II. Experiments from Section 3.1 where NNGA-DYCORS performed better than or as well as the next best constituent algorithm by one standard deviation have their mean bolded. Variants of NNGA-DYCORS are tested in Sections 3.2 – 3.4. All data shown are from the final batch of experiments for all algorithms.

Function	Bounds	
Ackley	*[-15,20]*	$y = -20\exp\left(-0.2\sqrt{p^{-1}\Sigma_i^p x_i^2}\right) - \exp\left(p^{-1}\sqrt{\Sigma_i^p \cos(2\pi x_i)}\right) + 20 + \exp(1)$
Rastrigin	*[-4,5]*	$y = 10p + \Sigma_i^p (x_i^2 - 10\cos(2\pi x_i))$
Griewank	*[-500,700]*	$y = \frac{\Sigma_i^p x_i^2}{4000} + \Pi_i^p \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
Levy	*[-5,5]*	$y = \sin^2(\pi w_1) + \Sigma_i^{p-1}(w_i - 1)(1 + 10\sin^2(\pi w_i + 1) + (w_p - 1)^2(1 + \sin^2(2\pi w_p))$ $w_i = 1 + \frac{x_i - 1}{4}$
Michalewicz	*[0,π]*	$y = -\Sigma_i^p \sin(x_i)\sin^{20}\left(\frac{ix_i^2}{\pi}\right)$
Rosenbrock	*[-5,10]*	$y = \Sigma_i^{p-1} 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2$
Dixon-Price	*[-10,10]*	$y = (x_1 - 1)^2 + \Sigma_{i=2}^p (2x_i^2 - x_{i-1})^2$
Styblinski-Tang	*[-5,5]*	$y = \frac{1}{2}\Sigma_i^p x_i^4 - 16x_i^2 + 5x_i$
Sphere	*[-5.12,5.12]*	$y = \Sigma_i^p x_i^2$
Zakharov	*[-5,10]*	$y = \Sigma_i^p x_i^2 + \left(\Sigma_i^p 0.5ix_i\right)^2 + \left(\Sigma_i^p 0.5ix_i\right)^4$

Table 1. Test Functions.

197x166mm (150 x 150 DPI)

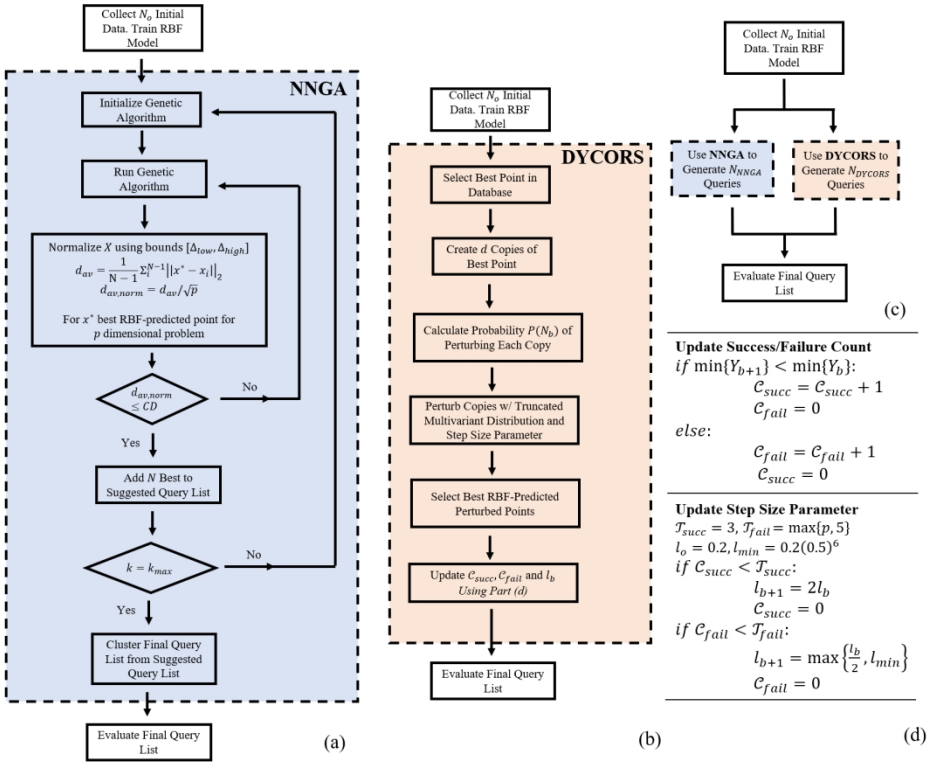

Figure 1. Flow Charts of Optimization Algorithms. (a) NNGA and (b) DYCORS are used to create (c) the hybrid NNGA-DYCORS. (d) The Step Size Adjustment and Success/Failure Count method used in (b) is displayed as well.

324x251mm (150 x 150 DPI)

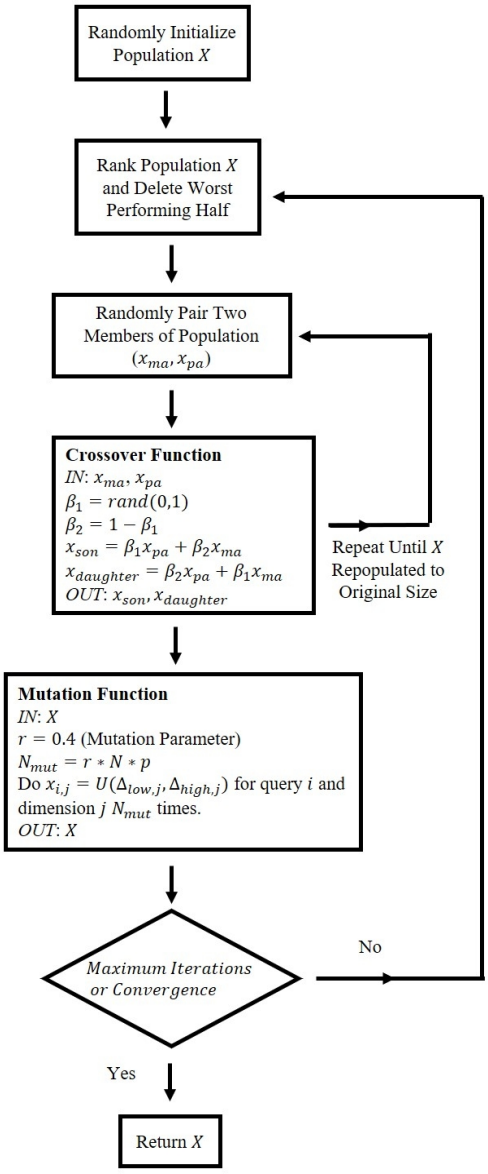

Figure 2. Truncated Genetic Algorithm. Used as stochastic optimizer for NNGA based on ranking, pairing, crossover, and mutation steps to generate optimal parameter combinations. Maximum iterations set at 100, CD and r set by user.

58x124mm (300 x 300 DPI)

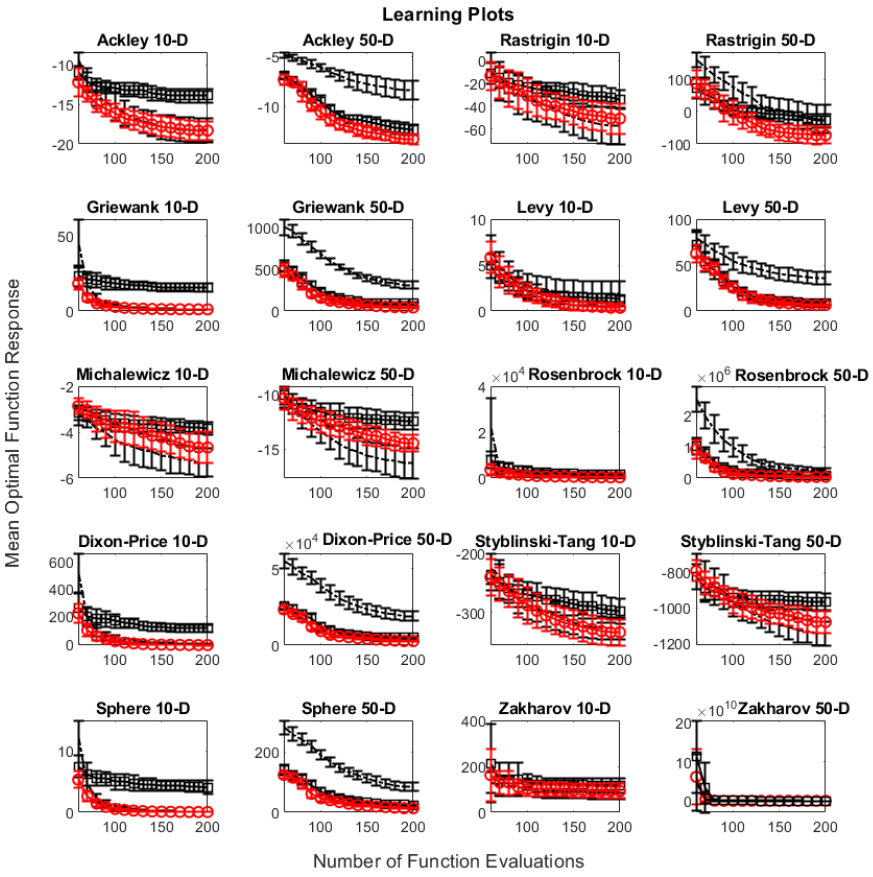

Figure 3. Hybrid Algorithm Performance. NNGA black squares, DYCORS black dotted line, NNGA-DYCORS red circles. Shown is average minimum of response for each of the test functions in Table 1 plotted against cumulative queries. NNGA-DYCORS hybrid performs as well as best NNGA and DYCORS performances.

253x245mm (96 x 96 DPI)

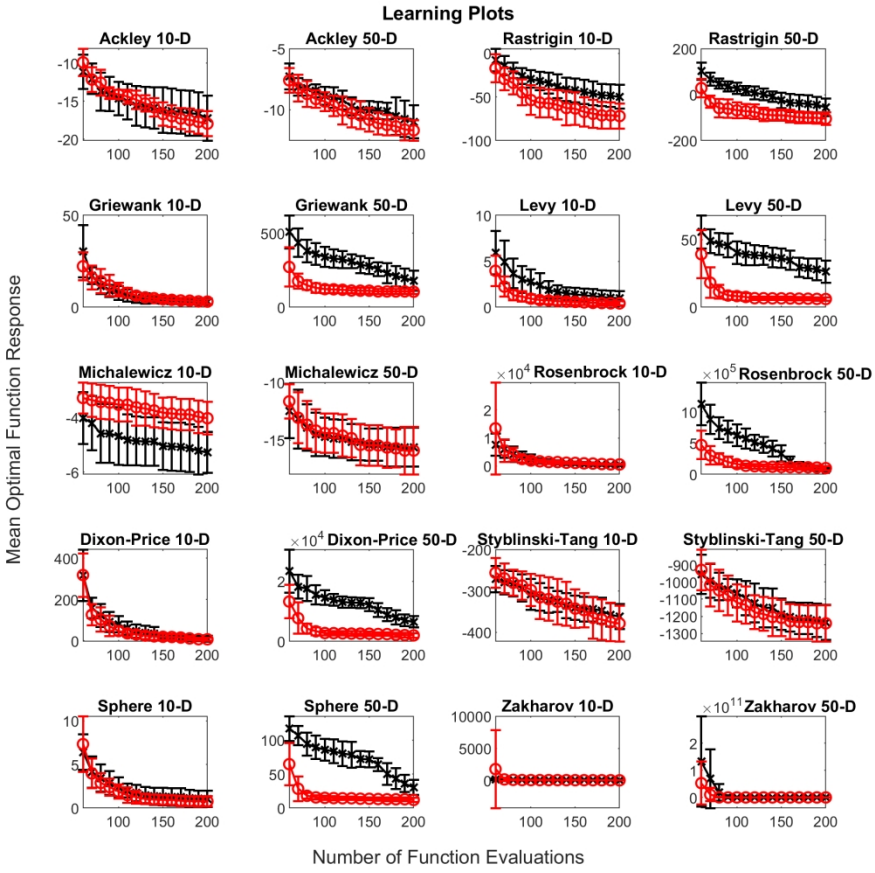

Figure 4. Algorithm Performance in the Presence of Noise. NNGA-DYCORS with node optimization shown in red circles, NNGA-DYCORS without scheme in black X's. Shown is average minimum of response for each of the test functions in Table 1 plotted against cumulative queries for noise level (20% of response). Node optimization improves generally learner performance in the presence of simulated experimental noise.

202x196mm (300 x 300 DPI)

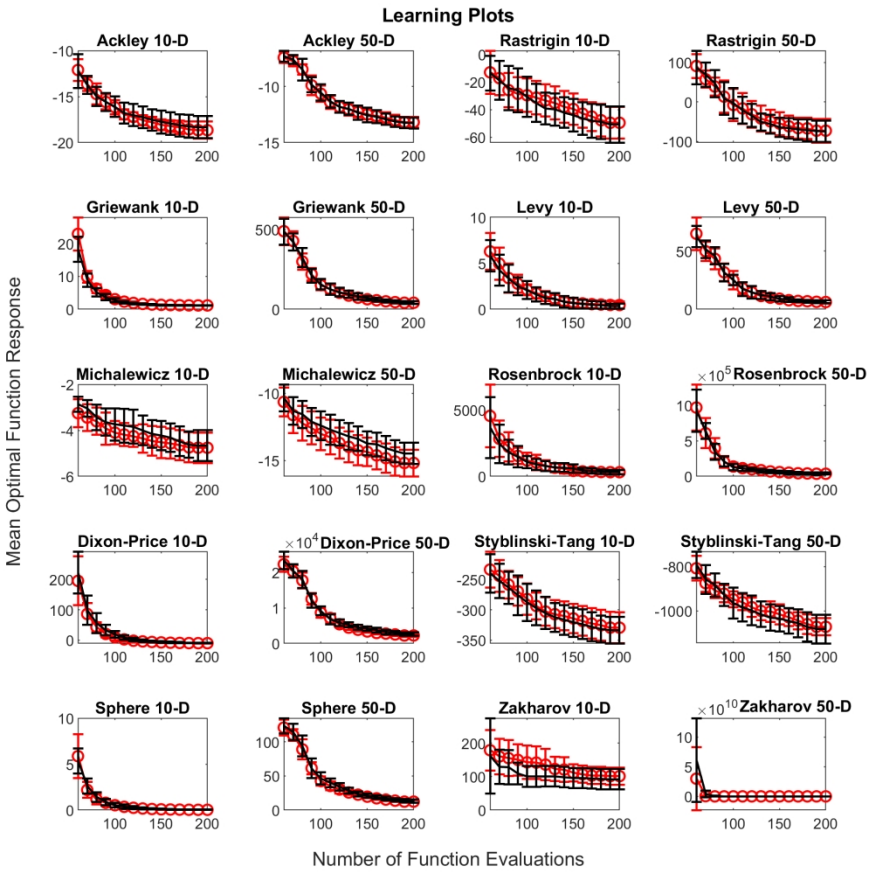

Figure 5. Algorithm Performance using Dynamic Convergence Parameter Strategy. NNGA-DYCORS with dynamic convergence parameter strategy in red circles, NNGA-DYCORS without strategy in black. Shown is average minimum of response for each of the test functions in Table 1 plotted against cumulative queries. Performance is not much improved by using the dynamic strategy.

202x196mm (300 x 300 DPI)

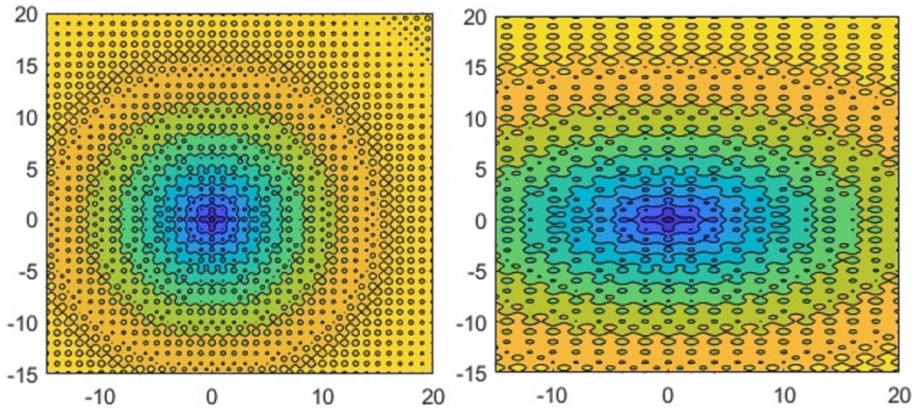

Figure 6. Effect of Scaling on Functions. Left figure Ackley Function, right is Ackley Function with ordinate axis modified by $\gamma = 0.5$.

117x51mm (300 x 300 DPI)

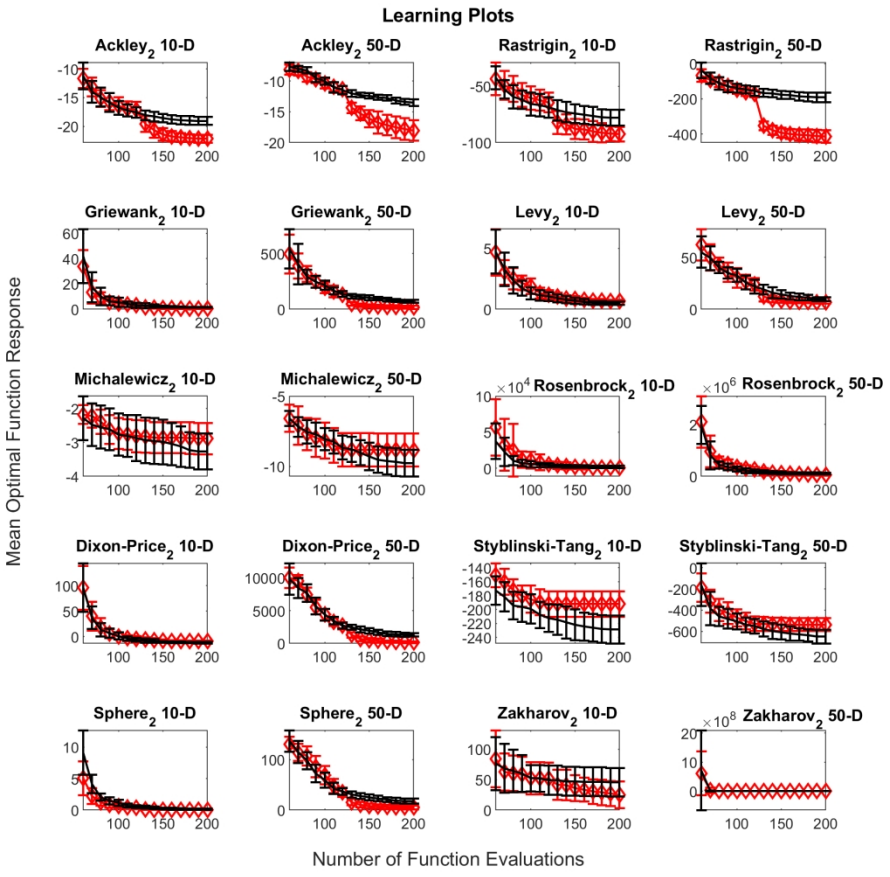

Figure 7. Algorithm Performance using Subset Selection. NNGA-DYCORS with subset selection in red diamond, regular NNGA-DYCORS in solid black line. Shown is average minimum of response for each of the test functions in Table 1 plotted against cumulative queries. Subset selection does not have a consistently positive effect on algorithm performance. The subscript (2) indicates the sensitivity vector has been applied to the test problems.

202x196mm (300 x 300 DPI)