Constraint Enforcement to Guarantee Strictly Feasible Solutions in a Surrogate Based Optimizer

Ahmed Abouhussein[1,4], Nusrat Islam[2,4] and Yulia T. Peet[3,4]

***Abstract*— Surrogate based optimization (SBO) methods have gained popularity in the field of constrained optimization of expensive black-box functions. However, constraint handling methods do not usually guarantee strictly feasible candidates during optimization. This can become an issue in applied engineering problems where design variables must remain feasible for simulations to not fail. We propose a simple constraint-handling method for computationally inexpensive constraint functions which guarantees strictly feasible candidates when using a surrogate-based optimizer. We compare our method to other SBO algorithms and an EA on five analytical test functions, and an applied fully-resolved Computational Fluid Dynamics (CFD) problem concerned with optimization of an undulatory swimming of a fish-like body, and show that the proposed algorithm shows favorable results while guaranteeing feasible candidates.**

I. INTRODUCTION

The goal of this study is to propose a constraint handling technique (CHT) for a surrogate based optimization (SBO) algorithm with inexpensive constraint functions to ensure strictly feasible candidates while keeping the number of function evaluations low. The purpose behind having such an algorithm would be to solve practical engineering problems with computationally-intensive black-box objective functions such as finding the optimum modes of locomotion for a soft-robot fish for efficient underwater propulsion.

There exists a wide spectrum of algorithms under the umbrella of SBO with varying performance depending on the choices made in developing the surrogate optimization framework [1], [2]. The choices can vary in regards of the model used to construct the surrogate, the infill strategy used to refine the surrogate over the optimization cycle and the CHT used to deal with available constraints. As the focus of this paper is feasible candidates, we restrict our discussion to the infill strategies and the CHTs associated with SBO algorithms.

A popular class of methods involves casting the constrained optimization problem as a multi-objective optimization problem where the objectives minimize the fitness function and constraint violations. One way to solve the multi-objective optimization is to use a "filter" approach [3]. The objective function and the constraints form a Pareto front, where candidates that are non-dominated, those which lay outside the Pareto front, are accepted through the filter and dominated candidates, those which lay within the Pareto front, remain unfiltered. The filter approach, along with other multi-objective optimization techniques [4], [5], relax constraint requirements during the optimization procedure, and therefor may not be easily adapted to practical problems where the constraints must be strictly enforced. The requirement for strictly feasible candidates can arise, for example, in optimization problems where constraint failures result in non physically realizable parameters in a CFD simulation. An alternative approach is to cast the constrained optimization problem into a single unconstrained objective with an added term which penalizes constraint violations. An example is the "extreme barrier" treatment, where the objective function is set to infinity in the infeasible domain [6]. Convergence and accuracy of these methods is highly dependent on the threshold of the penalty parameter [7]. A high threshold can dominate the objective function and deteriorate convergence speed, while a low threshold could produce infeasible candidates. Other CHTs have been developed around different infill strategies. For example, it has been suggested that for the Expected Improvement (EI) infill criteria, used in the Efficient Global Optimization (EGO) algorithm by Jones, Scholnau and Welch [8], the EI could be set to zero when any constraint is violated [9], [10]. When the search is solely guided by the EI criteria, there will be no exploration of the infeasible regime. Another possible solution arises when considering optimization problems where simulation failure can not be determined *a priori* [11]. The suggested treatment [11] is to impute a value that is proportional to the sum of the surrogate response prediction and the prediction uncertainty where the simulation fails. Although other CHTs, aimed at problems with expensive constraints, rely on creating surrogate predictions of the constraint functions [12], they are generally inadvisable when constraint functions are inexpensive [13]. We propose a simple rejection based CHT for inexpensive constraints, capable of insuring strictly feasible candidates, for the globally convergent Metric Stochastic Response Surface (MSRS) infill sampling criteria [14]. We choose an Ordinary Kriging (OK) surface [15] as the surrogate model to exploit with the constrained MSRS infill strategy and the resulting SBO algorithm variant is referred to as OK-CMSRS.

We compare the OK-CMSRS algorithm with several other algorithms which include: 1) The EGO algorithm with zero EI for infeasible candidates, referred to as EGO-Z 2) an SBO algorithm with an OK model, a MSRS infill strategy and a data imputation treatment [12], referred to as OK-IMSRS 3) an SBO algorithm with an OK model, a MSRS infill strategy,

*This work was supported by NSF CMMI Award No. 1762827

[1]Ph.D. student, email: aabouhus@asu.edu

[2]Research Assistant, email: nislam4@asu.edu

[3]Assistant Professor, email: ypeet@asu.edu

[4]School of Engineering, Matter, Transport and Energy, Arizona State University

and a simple penalty outside the feasible domain, referred to as OK-PMSRS and, finally, 4) the CMAES evoluationary algorithm which treats infeasible solutions by resampling [16]. The algorithms are benchmarked on the following six test problems: 1) the analytical Rosenbrock function [17], 2) the Shifted Rotated Rastrigin's Function [18], 3) the Tension/Compression Spring Design (RC17) problem [19], 4) Gear Train Design (RC31) problem [19] and 5) optimization of the locomotion of a thunniform bio-inspired propulsor based on Computational Fluid Dynamics (CFD) results. Algorithm performance metrics include a function evaluation count and a solution error to assess accuracy and convergence speed. A formal problem presentation is given next in Section 2. A method description of the OK-CMSRS algorithm as well as the constraint handling approach is presented in Section 3. Proofs of convergence for the MSRS infill strategy as well as the for proposed CHT variant will be discussed in Section 4. Details regarding the computational set-up of the problems and the results are given in Section 5. Finally, a discussion of the results follows in Section 6.

II. PROBLEM FORMULATION

The general formulation of the optimization problem considered can be expressed as follows,

$$\begin{aligned} &\text{minimize} \quad f(\mathbf{x}) \\ &\text{subject to} \quad \mathbf{x} \in \mathbb{R}^n \end{aligned} \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is the objective function, $\mathbf{x}$ is a vector of design parameters. The set $S \subseteq \mathbb{R}^n$ contains the n-dimensional search space, which would define a rectangle in $\mathbb{R}^2$ or a rectangular cuboid in $\mathbb{R}^3$:

$$l(i) \leq x_i \leq u(i), \quad 1 \leq i \leq n \tag{2}$$

where $l(i)$ and $u(i)$ represent lower and upper bounds, respectively, on a design parameter in the ith dimension. The set $C \subseteq \mathbb{R}^n$ contains a set of $m \geq 0$ constraints:

$$\begin{aligned} &g_r(\mathbf{x}) \leq 0, \quad r = 1, ..., q, \\ &h_r(\mathbf{x}) = 0, \quad r = q+1, ..., m \end{aligned} \tag{3}$$

where $g_r(\mathbf{x})$ and $h_r(\mathbf{x})$ are referred to as the inequality and equality constraint sets, respectively, on the design parameter vector, $\mathbf{x}$.

A. Test Problem I

The Rosenbrock function is a canonical optimization test function known for containing a global minimum within a wide basin. The two-dimensional form of the function used in this study is given as follows:

$$f(x_1, x_2) = (a - x_1)^2 + b(x_2 - x_1^2)^2 \tag{4}$$

with global minimum located at $(x_1^*, x_2^*) = (a, a^2)$, where $f(x_1^*, x_2^*) = 0$. The parameters were chosen to be $a = 0.35$ and $b = 100$ to insure that the global minimum resides within the constraint domain as shown in Fig. 1. The C constraint set, chosen to resemble the natural constraints in the last test problem, is given below:

$$\begin{aligned} g_1(\mathbf{x}) &= x_2 + 2.5x_1^2 - 0.5 \leq 0 \\ g_2(\mathbf{x}) &= -x_2 - x_1 + 0.4 \leq 0 \end{aligned} \tag{5}$$

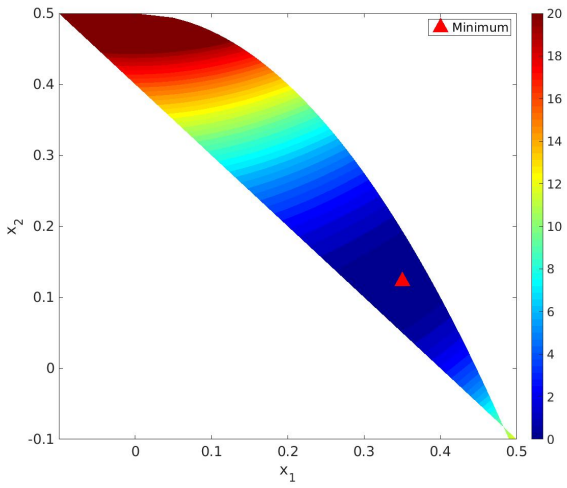

Fig. 1: 2-D Rosenbrock over the constrained domain

B. Test Problem II

The second analytical function considered is the shifted rotated Rastrigin function used as one of the benchmark functions for CEC2005 competition [18]. This variant of the Rastrigin function provides a non-linear, non-separable, highly multi-modal challenging test function (Figure 2). The 2-D form used in this study is given as follows:

$$f(x_1, x_2) = \sum_{i=1}^{2} (z_i^2 - 10\cos(2\pi z_i)) - 33 \tag{6}$$

where $\mathbf{z} = (\mathbf{x} - \mathbf{o})\mathbf{M}$, $\mathbf{x} = [x_1, x_2]$, $\mathbf{o} = [o_1, o_2]$ is the shifted global optimum, with $f(\mathbf{o}) = -33$, $\mathbf{M}$ is a linear transformation matrix with condition number = 2. The C constraint set in this case is:

$$\begin{aligned} g_1(\mathbf{x}) &= x_2 + 0.15x_1^2 - 2 \leq 0 \\ g_2(\mathbf{x}) &= -x_2 - x_1 - 1 \leq 0 \end{aligned} \tag{7}$$

C. Test Problem III & IV

The third and fourth test problem were chosen to be design problems RC17 and RC31, respectively, from the 2020 CEC constrained optimization competition [19]. These multi-dimensional non-linear problems with demanding constraint requirements are of practical relevance to mechanical design problems in fields such as spring and compound gear design.

TABLE I: Optimization Algorithms

Software Language	Solver	Comments
MATLAB	EGO-Z	EGO available through DACE [21]
MATLAB	OK-IMSRS	MSRS available through MATSuMoTo [22]
MATLAB	OK-PMSRS	MSRS available through MATSuMoTo [22]
MATLAB	CMAEES	See Hansen [16]
MATLAB	OK-CMSRS	See Section 3

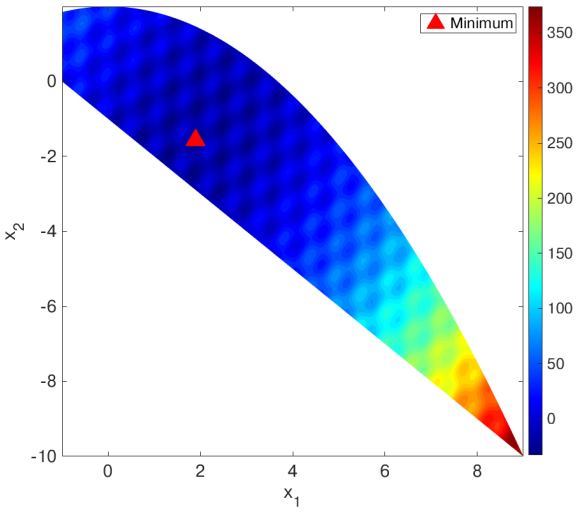

Fig. 2: 2-D Shifted Rotated Rastrigin over the constrained domain

D. Test Problem V

The last optimization problem is based on a black-box simulation which computationally models the swimming behaviour a 2D single thunniform fish presented by Xu and Peet [20]. The optimization process aims to locate an optimum fishing mode related to a kinematic gait that maximizes the start-up propulsive force with minimal energy expended, referred to as the propulsive efficiency. The fish undulation is described in terms of its center-line position:

$$y_c(p,t) = [x_1\frac{p}{L} + x_2(\frac{p}{L})^2]\sin\Big(2\pi(\frac{p}{\lambda L} - ft)\Big) \qquad (8)$$

where x_1 and x_2 are dimensional undetermined linear and quadratic wave amplitude coefficients, p is the streamwise fish position, y is the spanwise fish position, λ is the body wave length which is 1.1, f is the body wave tail-beat frequency taken to be 1 Hz, L is the dimensional length of the fish taken to be 0.3 m and t is time. Consequently, the coefficients $\{x_1, x_2\}$ span a range of swimming modes. In order to allow for only physically realizable modes, the following constraint set was imposed [23]–[25]:

$$\begin{aligned} g_1(\mathbf{x}) &= 0.4x_2L + x_1^2 \leq 0 \\ g_2(\mathbf{x}) &= |x_2 + x_1| - 0.1L \leq 0 \\ g_3(\mathbf{x}) &= x_2 \leq 0 \end{aligned} \qquad (9)$$

The propulsive efficiency, η, is defined as follows:

$$\eta(x_1, x_2, t) = \frac{\int_0^t \oint_{body} -\sigma \cdot n_p \cdot U \mathrm{d}p\mathrm{d}t}{\int_0^t \oint_{body} -\sigma \cdot n_y \cdot \nu \mathrm{d}p\mathrm{d}t} \qquad (10)$$

where σ is he total Cauchy stress tensor which includes viscous and pressure forces, $n = \{n_p, n_y\}$ is the outer unit normal vector on the body surface, $\nu(p,t) = \partial y_m(p,t)/\partial t$ is the surface transverse velocity due to undulation, and $U(t)$ is the propulsive forward velocity. During optimization, the negative value of the propulsive efficiency is taken as the objective function and the $\{x_1, x_2\}$ coefficients are taken as design parameters subject to the feasible parameter space.

III. Method Description

SBO algorithms rely on approximating solutions based on a surrogate model of the objective function. First an initial surrogate model of the objective function is created using a data set of true function evaluations sampled with a space filling strategy. Then, with each iteration an infill criteria, which attempts to balance global and local exploration, is used to refine the surrogate with multiple surrogate function evaluations and one true function evaluation. In this work, we choose a sampling technique and a surrogate model from the DACE [21] Matlab tool box. We define an infill strategy according to the Metric Stochastic Response Surface (*MSRS*) method proposed by Regis and Shoemaker [14]. Finally, we extend the *MSRS* method to produce strictly feasible candidates at each iteration step.

We present below the SBO algorithm, OK-CMSRS, used in this study. Let the feasible domain, D, be defined as $D = S \cap C$. Let M_k and $s_k(x)$ be defined as the set of candidates used for "true" function evaluations and the surrogate model, respectively, at iteration k. Define maximum number of iterations, k_{max}, and a tolerance, tol.

Step 1: Sample a finite set of evaluation points $T \subset D$ using Latin Hypercube Sampling [26], where card$(T) = 20$. Evaluate $f(T)$, where $f(x)$ is the true objective function. Identify the current best point x_0. Set $M_0 = T$.

Step 2: Fit a Kriging model surrogate, $s_0(x)$, with a Gaussian correlation function and 0-th order regression polynomial. In other words, the surrogate model is assumed to have a constant mean and a stochastic error term that is modeled by a Gaussian process. This is referred to as ordinary kriging and allows for a flexible and reliable prediction method [15].

Step 3: While $(k < k_{max})$

a) Create a set of strictly feasible candidate points, X_k, according to the proposed algorithm (section 3.A.) and evaluate $s_k(X_k)$.
b) Use the *MSRS* method which assigns a weighted score to each point in set X_k based on two criteria: 1) the distance of points in X_k to M_k, and 2) the surrogate response values, $s_k(X_k)$. The weighted score insures that the next candidate point has a low objective value that is far away from already sampled points. The point with the best weighted score is identified as the next evaluation point, x_k.
c) Evaluate $f(x_k)$. If tol is met: **break**.
d) Set $M_{k+1} = M_k + x_k$. Re-fit s_{k+1} with M_{k+1}. Set $k = k + 1$.

Step 4: Return x_k

A. Constrained Candidate Sampling

The candidate points are split into two categories [14]:

1) Uniformly sampled global points: The first set, U_k, is generated by a uniform random sampling of points from the box-constrained domain such that $U_k \subset S$. We set $\text{card}(U) = 2000$.
2) Normally sampled local points: The second set N_k is generated by adding perturbations to x_k drawn from a random normal distribution with zero mean and unit variance. The are three perturbation rates chosen: one-tenth, one-hundredth and one-thousandth of the smallest variable range. The smallest variable range is defined as $\min(u - l)$. We set $\text{card}(N) = 2000$.

We define the possibly unfeasible candidate set as $X'_k = U_k + N_k$. It follows that $\text{card}(X'_k) = 4000$. We note that $X'_k \subset S$, however it may be that $X'_k \not\subset S \cap C$. We then enforce all constraints, linear and/or nonlinear, through the following algorithm:

Step 1: Evaluate $g_r(X'_k)$ for $r = 1, ..., q$.
Step 2: Define 'penalty' vector, J, for each candidate point:

$$j_i = \sum_{r=1}^{m} \max(0, g_r(x_i)), \quad i = 1, ..., \text{card}(X'_k) \tag{11}$$

where j_i are the entries of set J.
Step 3: The candidate points set, X_k, is simply defined as: $X_k = X'_k(J = 0)$. In other words, the X_k candidates are the candidates in X'_k with a zero penalty.

This method does not guarantee that the candidate set will always be the of expected $\text{card}(X'_k)$ but the final candidate set X_k is always feasible for all possible candidate solutions. If it happens that $X_k \subset M_k$, then the candidates are resampled.

IV. Convergence Proofs

Let $\mathcal{E}_k = \{T, X'_1, ..., X'_k\}$. In the work of [14], two conditions are introduced for use in a latter theorem which proves global convergence:

[C1] For all k, elements of X'_k are conditionally independent given the random vectors in $\mathcal{E}_{k-1}$.

[C2] For any $x_i \in S \cap C, i = 1, ..., \text{card}(X'_k)$ and $\delta > 0$, there exists $\nu_i(x, \delta) > 0$ such that

$$P[X'_k \in S \cap C \cap B(x, \delta) | \sigma(\mathcal{E}_{k-1})] \geq \nu_i(x, \delta)$$

where $B(x, \delta)$ is the open ball of radius δ and center x and $\sigma(\mathcal{E}_{k-1})$ is the σ-algebra generated by the random vectors in $\mathcal{E}_{k-1}$. C2 insures that no region of $S \cap C$ is left unexplored. The theorem then states:

Theorem 1: Let f be a function defined on $S \cap C \subseteq \mathbb{R}^d$ and suppose that x^* is the unique global minimizer of f on $S \cap C$ in the sense that $f(x^*) = \inf_{x \in S \cap C} f(x) > -\infty$ and $\inf_{x \in S \cap C, ||x - x^*|| \geq \eta} f(x) > f(x^*)$ for all $\eta > 0$. Suppose further that the SRS method generates X'_k which satisfies both [C1] and [C2] conditions. Define the sequence of random vectors $\{x^*_k\} := x^*_k = x_k$ if $f(x_k) < f(x^*_{x-1})$ while $x^*_k = x^*_{k-1}$ otherwise. Then $x^*_k \to x^*$ almost surely.

Proof: For $\epsilon > 0$ and $k \geq 1$, the event $[x_k \in S \cap C : f(x_k) < f(x^*) + \epsilon] = [x_k \in S \cap C : |f(x_k) - f(x^*)| < \epsilon]$. Since f is continuous on x^*, there exists $\delta(\epsilon) > 0$ such that $|f(x) - f(x^*)| < \epsilon$ whenever $||x - x^*|| < \delta(\epsilon)$. Hence, $[x_k \in S \cap C : ||x_k - x^*|| < \delta(\epsilon)] \subseteq [x_k \in S \cap C : |f(x_k) - f(x^*)| < \epsilon]$ and so,

$$\begin{aligned} &P[x_k \in S \cap C : |f(x_k) - f(x^*)| < \epsilon | \sigma(\mathcal{E}_{k-2})] \\ &\geq P[x_k \in S \cap C : ||x_k - x^*|| < \delta(\epsilon) | \sigma(\mathcal{E}_{k-2})] \\ &= P[x_k \in S \cap C \cap B(x^*, \delta(\epsilon)) | \sigma(\mathcal{E}_{k-2})] \end{aligned} \tag{12}$$

If $X'_k \in S \cap C \cap B(x^*, \delta(\epsilon))$, then it must be that $x_k \in S \cap C \cap B(x^*, \delta(\epsilon))$. Hence,

$$\begin{aligned} &P[x_k \in S \cap C \cap B(x^*, \delta(\epsilon)) | \sigma(\mathcal{E}_{k-2})] \\ &\geq P[X'_k \in S \cap C \cap B(x^*, \delta(\epsilon)) | \sigma(\mathcal{E}_{k-2})] \\ &= \prod_{i=1}^{\text{card}(X'_k)} P[X'_{i^k} \in S \cap C \cap B(x^*, \delta(\epsilon)) | \sigma(\mathcal{E}_{k-2})] \quad \text{[C1]} \\ &\geq \prod_{i=1}^{\text{card}(X'_k)} \nu_i(x^*, \delta(\epsilon)) =: L(\epsilon) > 0 \quad \text{[C2]}, \end{aligned} \tag{13}$$

Combining 12 and 13 leads to: $P[x_k \in S \cap C : f(x_k) < f(x^*) + \epsilon | \sigma(\mathcal{E}_{k-2})] \geq L(\epsilon)$. Following the same argument in the proof of the theorem in p. 40 of [27], $x^*_k \to x^*$ almost surely. ■

Therefor any candidate generation method, which satisfies [C1] and [C2] in the SRS framework of [14], converges to the global optima in a probabilistic sense. The authors show that the uniformly sampled global (see III A. 1) and the normally sampled local (see III A. 2) points satisfy conditions [C1] and [C2] and we propose no changes to the candidate generation scheme here.

After candidate generation, we argue that rejecting infeasible candidates, as was done in Section III A, does not affect

TABLE II: Optimization results for test problems I - IV. FE: Function Evaluations Count, FE MOE: Function Evaluations Margin of Error and BF: Best Function Value. [1] Due to a high function evaluation count, the EGO surrogate for the Rosenbrock and Spring problems could not be efficiently constructed on the desktop computer.

Algorithm	Rosenbrock			Rastrigin			Spring			Gear		
	FE	FE MOE	BF	FE	FE MOE	BF	FE	FE MOE	BF	FE	FE MOE	BF
EGO-Z	NA[1]	NA	NA	191	22	-30.9	NA[1]	NA	NA	32	4	2.74×10^{-4}
OK-IMSRS	69	13	4.88×10^{-4}	290	43	-30.8	889	84	1.66×10^{-3}	16	1	3.32×10^{-4}
OK-PMSRS	460	84	5.11×10^{-4}	317	60	-30.7	968	45	1.97×10^{-2}	16	1	2.92×10^{-4}
CMA-ES	60	6	4.40×10^{-4}	130	51	-30.6	243	90	1.36×10^{-2}	25	3	1.95×10^{-4}
OK-CMSRS	33	2	5.54×10^{-4}	127	15	-30.9	105	45	1.35×10^{-2}	14	1	2.98×10^{-4}

convergence, since:

$$\begin{aligned} &P[x_k \in S \cap C \cap B(x^*, \delta(\epsilon))|\sigma(\mathcal{E}_{k-2})] \\ &\geq P[X_k \in S \cap C \cap B(x^*, \delta(\epsilon))|\sigma(\mathcal{E}_{k-2})] \\ &\geq P[X'_k \in S \cap C \cap B(x^*, \delta(\epsilon))|\sigma(\mathcal{E}_{k-2})] \\ &= \prod_{i=1}^{\text{card}(X'_k)} P[X'_{i^k} \in S \cap C \cap B(x^*, \delta(\epsilon))|\sigma(\mathcal{E}_{k-2})] \quad [C1] \\ &\geq \prod_{i=1}^{\text{card}(X'_k)} \nu_i(x^*, \delta(\epsilon)) =: L(\epsilon) > 0 \quad [C2], \end{aligned} \tag{14}$$

In fact, the authors of [14] use the same argument to treat simple box constraints, where $X_k = X'_k$ if $X'_k \in S$ while $X_k = \min(\max(l(i), X_k), u(i))$ for $i = 1, ..., \text{card}(X'_k)$ if X'_k not in S.

V. Computational Set-up & Results

A. Test Problems I - IV

Numerical simulations were performed on a desktop with a CPU featuring six 3.20GHz Intel processors on a Linux environment. The optimization algorithms were run on Matlab 2020a. The convergence criteria, tol, was set to $0.9f^*(x)$, with $f^*(x)$ being the *a priori* identified global function minimum. Exceptions include Test Problem I and Test Problem IV where tol is set to 1×10^{-3} since $f^*(x)$ in those cases is 0. All algorithms were set to terminate at 1000 iterations, regardless of tol. The function evaluations count as well as the best function value, achieved after convergence, are reported in Table 2. Additionally, a margin of error (MOE) on the function evaluation count is calculated with 95% confidence for all solvers. as follows:

$$\begin{aligned} SD &= \sqrt{\frac{\sum_{i=1}^{m}(z_i - \mu_z)^2}{N-1}} \\ MOE &= \frac{SD}{\sqrt{N}} * 1.96 \end{aligned} \tag{15}$$

where z_i is a random variable at iteration i, μ_z is the sample mean, N is the number of realizations taken to be 50, SD is the standard deviation and MOE is the margin of error.

B. Test Problem V

The simulations were performed on three nodes of a super computing cluster with each compute node containing two Intel Xeon E5-2680 v4 CPUs running at 2.40GHz. Simulations of the swimming fish were performed using Nek5000, a high-fidelity open source CFD solver [28]. The optimization was set to be terminated at a relative convergence tolerance of 1×10^{-6}. The relative convergence tolerance is defined by the relative change of the objective function between successive iterations. MATLAB and Linux bash scripts were used to automate the optimization procedure to allow for no user interference. The final Kriging surrogate was used to create an objective function landscape contour found in Fig. 3. The results for this problem are shown in Table 3.

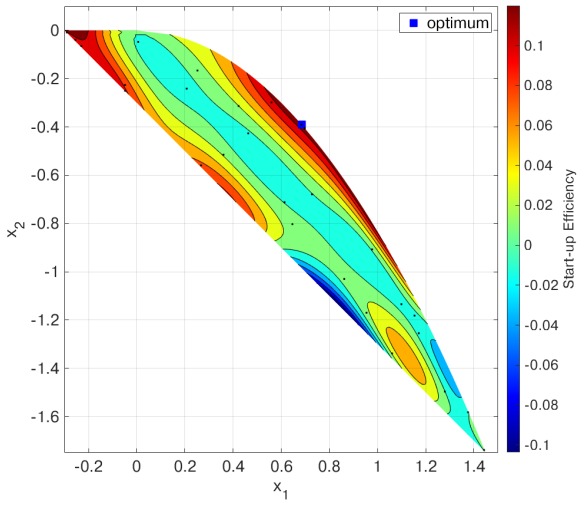

Fig. 3: 2-D start-up efficiency function over the constrained domain

VI. Discussion

OK-CMSRS reported lower function evaluations when compared to all other solvers on all four benchmark problems, while also maintaining error levels within specified

TABLE III: Black-box optimization

Algorithm	Function Evaluation	Start-up Efficiency	Optimized Parameter Set $\{x_1, x_2\}$
OK-CMSRS	58	13.1%	$\{0.68, -0.39\}$

tolerance. The EGO algorithm, with a zero EI CHT, typically demands high function evaluations and for some test problems could become inefficient to run on a desktop computer. Algorithms with an OK surrogate function require less memory footprint but displayed varying performance based on the selected CHT. For example, the OK-CMSRS required less than half the function evaluations for the Rosenbrock function when compared to OK-IMSRS. The OK-PMSRS required more than 12 times the amount of function evaluations for the same problem, which could be due to the "cliff" effect associated with penalty functions, as discussed in [10]. The CMAES algorithms is shown to be more efficient at arriving to a solution within tolerance when compared to other algorithms, with the only exception being OK-CMSRS. Moreover, the CMAES results in a higher MOE on function evaluation counts, especially for the Rastrigin function, when compared to the OK-CMSRS. Since, the OK-CMSRS showed the least variation in the best solution reported (lowest MOE) as well as the lowest function evaluations across the 50 optimization runs, it was chosen solver for the black-box optimization problem. The OK-CMSRS algorithm converged to a solution within 58 iterations and a resulting swimmer efficiency of 13.1%. The results show indicate that the OK-CMSRS is a promising tool for expensive simulations-based black-box optimization problems with inexpensive constraints.

VII. Acknowledgements

This research is supported by NSF CMMI grant # 1762827.

References

[1] S. Razavi, B. A. Tolson, and D. H. Burn, "Review of surrogate modeling in water resources,' Water Resources Research, vol. 48, no. 7, 2012.

[2] C. Wang, Q. Duan, W. Gong, A. Ye, Z. Di, and C. Miao, "An evaluation of adaptive surrogate modeling based optimization with two benchmark problems," Environmental Modelling & Software, vol. 60, pp. 167-179, 2014.

[3] C. Audet, J. Denni, D. Moore, A. Booker, and P. Frank, "A surrogate-model-based method for constrained optimization," 8th Symposium on Multidisciplinary Analysis and Optimization, 2000.

[4] J. Müller and J. D. Woodbury, "GOSAC: global optimization with surrogate approximation of constraints," Journal of Global Optimization, vol. 69, no. 1, pp. 117–136, 2017.

[5] R. G. Regis, "Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points," Engineering Optimization, vol. 46, no. 2, pp. 218–243, 2013.

[6] C. Audet and J. E. Dennis, "Mesh Adaptive Direct Search Algorithms for Constrained Optimization," SIAM Journal on Optimization, vol. 17, no. 1, pp. 188–217, 2006.

[7] N. V. Queipo, R. T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and P. Kevin Tucker, "Surrogate-based analysis and optimization," Progress in Aerospace Sciences, vol. 41, no. 1, pp. 1–28, Jan. 2005.

[8] D. R. Jones, M. Schonlau, and W. J. Welch, Journal of Global Optimization, vol. 13, no. 4, pp. 455–492, 1998.

[9] J. R. Gardner, M. J. Kusner, Z. E. Xu, K. Q. Weinberger, and J. P. Cunningham, "Bayesian Optimization with Inequality Constraints," ICML, vol. 2014, pp. 937–945.

[10] A. Keane, A. Forrester, and A. Sobester, "Engineering Design via Surrogate Modelling: A Practical Guide," 2008.

[11] A. Forrester, A. Sóbester, and A. J. Keane, "Optimization with missing data," Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 462, no. 2067, pp. 935–945, 2006.

[12] J. M. Parr, A. J. Keane, A. I. J. Forrester, and C. M. E. Holden, "Infill sampling criteria for surrogate-based optimization with constraint handling," Engineering Optimization, vol. 44, no. 10, pp. 1147–1166, 2012.

[13] M. J. Sasena, P. Y. Papalambros, and P. Goovaerts, "The use of surrogate modeling algorithms to exploit disparities in function computation time within simulation-based optimization," Constraints, vol. 2, no. 5, 2001.

[14] R. G. Regis and C. A. Shoemaker, "A Stochastic Radial Basis Function Method for the Global Optimization of Expensive Functions," INFORMS Journal on Computing, vol. 19, no. 4, pp. 497–509, Nov. 2007.

[15] A. A. Eldeiry and L. A. Garcia, "Comparison of Ordinary Kriging, Regression Kriging, and Cokriging Techniques to Estimate Soil Salinity Using LANDSAT Images," Journal of Irrigation and Drainage Engineering, vol. 136, no. 6, pp. 355–364, Jun. 2010

[16] N. Hansen, "The CMA evolution strategy: A tutorial," arXiv:1604.00772, 2016.

[17] H. H. Rosenbrock, "An Automatic Method for Finding the Greatest or Least Value of a Function," The Computer Journal 3, no. 3 (March 1, 1960): 175–84, https://doi.org/10.1093/comjnl/3.3.175.

[18] 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 Technol. Univ., Singapore, Tech. Rep., IIT Kanpur, Kanpur, India, KanGAL Rep. #2005005, May 2005

[19] A. Kumar, G. Wu, M. Z. Ali, R. Mallipeddi, P. N. Suganthan, and S. Das, "A test-suite of non-convex constrained optimization problems from the real-world and some baseline results," Swarm and Evolutionary Computation, vol. 56, p. 100693, 2020.

[20] Y. Q. Xu and Y. Peet, "Optimum gaits of 2D thunniform locomotion for efficient swimming and performance of fish pair," AIAA paper 2018-2915, 24th AIAA Fluid Dynamics Conference, June 2018

[21] S.N. Lophaven, H.B. Nielsen, and J. Søndergaard. DACE a Matlab kriging toolbox. Technical report, Technical Report IMM-TR-2002-12, 2002.

[22] J. Mueller, "MATSuMoTo: The MATLAB surrogate model toolbox for computationally expensive black-box global optimization problems", arXiv preprint arXiv:1404. 4261, 2014.

[23] P. V. y Alvarado and K. Youcef-Toumi, "Soft-Robot Fish". In Bio-inspired Fishlike Underwater Robots, Springer, 2015, pp. 161–191

[24] I. Borazjani and F. Sotiropoulos, "On the role of form and kinematics on the hydrodynamics of self-propelled body/caudal fin swimming," J. Exp. Biol., vol. 213, no. 1, pp. 89–107, 2009.

[25] M. Hultmark, M. Leftwich, and A. J. Smits, "Flowfield measurements in the wake of a robotic lamprey," Exp. Fluids, vol. 43, no. 5, pp. 683-690, 2007.

[26] M. D. Mckay, R. J. Beckman, and W. J. Conover, "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," Technometrics, vol. 21, 1979.

[27] J. C. Spall, Introduction to stochastic search and optimization: estimation, simulation, and control, vol 65. John Wiley & Sons, 2005.

[28] P. Fischer, J. Lottes, S. Kerkemeier, O. Marin, K. Heisey, A. Obabko, E. Merzari and Y. Peet, Nek5000 User's manual, http://nek5000.mcs.anl.gov/files/2015/09/NEK_doc.pdf, 2015