Trust-Region Based Multi-Objective Optimization for Low Budget Scenarios

Proteek Chandan Roy[0000−0002−1646−9239], Rayan Hussein, Julian Blank[0000−0002−2227−6476], and Kalyanmoy Deb[0000−0001−7402−9939]

Michigan State University, East Lansing, MI 48824, USA,
{royprote,husseinr,blankjul,kdeb}@egr.msu.edu
home page: http://www.coin-laboratory.com/
COIN Report Number 2018011

Abstract. In many practical multi-objective optimization problems, evaluation of objectives and constraints are computationally time-consuming, because they require expensive simulation of complicated models. Researchers often use a comparatively less time-consuming surrogate or metamodel (model of models) to drive the optimization task. Effectiveness of the metamodeling method relies not only on how it manages the search process (to find infill sampling) but also how it deals with associated error uncertainty between metamodels and the true models during an optimization run. In this paper, we propose a metamodel-based multi-objective evolutionary algorithm that adaptively maintains regions of trust in variable space to make a balance between error uncertainty and progress. In contrast to other trust-region methods for single-objective optimization, our method aims to solve multi-objective expensive problems where we incorporate multiple trust regions, corresponding to multiple non-dominated solutions. These regions can grow or shrink in size according to the deviation between metamodel prediction and high-fidelity computed values. We introduce two performance indicators based on hypervolume and achievement scalarization function (ASF) to control the size of the trust regions. The results suggest that our proposed trust-region based methods can effectively solve test and real-world problems using a limited budget of solution evaluations with increased accuracy.

Keywords: Surrogate modeling · metamodel · trust-region method · multi-objective optimization.

1 Introduction

Most real-world problems involve time-consuming experiments and simulations that cause optimization to be increasingly expensive. To face this challenge and to reduce the computational cost, metamodels as approximations of exact or high-fidelity based computational models are used for the optimization task. There are a few challenges and decision factors in metamodel-based multi-objective optimization. First, given a multi-objective optimization problem with

M number of objectives and J number of constraints, one can model each objective and constraint separately thus having a total of $(M + J)$ metamodels. Also, one can combine all objectives using a scalarization method, e.g., weighted-sum, ϵ-constraint, Tchebychev, or achievement scalarization function (ASF) [16, 26] and metamodel them separately, thereby reducing the total number of metamodels to $(1 + J)$. The choice of metamodeling methodologies are discussed in recent papers [2, 3, 5, 9, 13, 19, 21, 22] by the authors. Second, a great deal of research has been done to formulate the criteria for finding infill or subsequent points for high-fidelity evaluation during optimization. For example, Emmerich et al [11] has generalized the concept of probability of improvement and the expected improvement to find infill solutions. Next, computational cost of constructing surrogates is a practical issue that prohibits us to build a large number of metamodels. Finding the best metamodel or approximation method is another concern for metamodel-based optimization. There is a wide variety of metamodels, such as Kriging, neural network, support vector regression, polynomial approximation and others, used in past studies [14]. Interestingly, the choice of metamodeling method may vary according to early, intermediate or late stage of the optimization process and is certainly not known a priori. Therefore, researchers have attempted to use multiple surrogate models in few efforts [25].

Although most existing methods are directed towards proposing more accurate metamodels or introducing efficient search schemes, there is a need for managing error uncertainty of one particular under-performing metamodel during optimization. A better management of a metamodel can, not only restrain the model from becoming worse, but also boost the performance by recognizing the inherent complexity of search regions. In this paper, we introduce a trust region concept for multi-objective optimization to reduce model uncertainty during metamodel-based optimization. This may allow a continuous convergence to the Pareto-front in some cases. Therefore, we don't completely rely on the assumptions made by the metamodel from the first iteration on.

The rest of this paper is organized as follows. Section 2 presents the previous works that are relevant to the trust region, uncertainty of metamodeling and overall metamodel-based algorithms. Section 3 discusses the new concepts introduced in this paper. Based on those concepts, the algorithm is presented in Section 4. Experimental settings and results are presented in Section 5. Section 6 concludes our study and suggests future work.

2 Related Studies

There have been several studies in metamodel-based multi-objective evolutionary algorithms for constrained and unconstrained problems. ParEGO [15], MOEA/D-EGO [27], SMS-EGO [18] and KRVEA [4] use scalarization methods (e.g., Tchebycheff) to combine multiple objectives into one and solve multiple scalarized versions of them to find a trade-off set of solutions. While these methods are mostly useful for unconstrained problems, they need to be modified for constrained scenarios. Hypervolume-based expected improvement [10] and maximum hypervolume contribution [18] are used as a performance criteria for infill points. Few recent studies [4, 19] outperformed standard evolutionary multi-objective optimization methods for unconstrained test problems.

Trust region methods are an effective mechanism to identify new infill points with a specific certainty. A few researchers have suggested using metamodel-based optimization with a trust region concept [1, 17]. They proposed a trust region framework for using approximation models with varying fidelity. Their approach is based on the trust region concept from nonlinear programming literature and was shown to be provably convergent for some of the original high-fidelity problems. A sequential quadratic approximation model was used in their study. In [17], a global version of the trust region method — Global Stochastic Trust Augmented Region (G-STAR) was proposed. The trust region was used to focus on simulation effort and balance between exploration and exploitation. They used Kriging as a metamodel for unconstrained single-objective optimization problems only. Few recent studies have considered for bi-objective [24] and multi-objective [12] problems with a convergence guarantee under mild conditions.

3 Trust Region Method for Single-Objective Optimization

The classical trust region method for single-objective optimization proceeds by building a metamodel $\hat{f}(.)$ for the original objective function $f(.)$. The prediction of the metamodel $\hat{f}(.)$ is minimized to obtain new infill points [1]:

$$\text{Minimize}_q \quad \hat{f}(q), \quad \text{Subject to} \quad \|q - p\| \leq \delta_k. \tag{1}$$

Here p is the current iterate (solution) and q is the new predicted point that can replace p in the next iteration. Typically, a quadratic model is used as $\hat{f}(.)$. The search is restricted within a radius δ_k from the current point p so that the metamodel approximates f well. The distance $\|q - p\|$ can be calculated using any norm. Without loss of generality, we use the Euclidean norm here. The trust region is updated by comparing the exact and the predicted value of the new point ($f(q)$ and $\hat{f}(q)$) with respect to the old point p by the following equation [1]:

$$r = \frac{f(p) - f(q)}{f(p) - \hat{f}(q)}. \tag{2}$$

Depending on the performance indicator r, the trust region might increase, decrease or remain the same. To decide what operation should be performed, two constants r_1 and r_2 are defined and the trust region is adapted as follows:

- If the model fails to improve objective value (that is, $r < r_1$), we reduce the trust region by multiplying existing δ_k with c_1 (< 1) and do not replace p with the new point q.
- If the model performs good in predicting function improvement from previous solution (that is, $r > r_2$), we increase δ_k for the next iteration by multiplying existing δ_k with c_2 (> 1) and we replace the old point p by new point q.

- Otherwise, we leave the trust region size δ_k as it was before.

We replace the old point p with the new point q, whenever q is a better point. The current point (p or q) is always associated with the updated trust radius. Suitable values of c_1 and c_2 are used.

3.1 Challenges and Motivation for Multi-Objective Optimization

The main challenges for applying the trust region concept in multi-objective evolutionary algorithms (MOEA) are handling multiple objectives and constraints. In addition, since MOEAs are population based methods, we also need to deal with multiple solutions and their individual trust regions. Moreover, there is a need for a meaningful performance metric to adapt trust radii of multiple high-fidelity solutions.

4 Proposed Trust Region in Metamodel-based Multi-objective Evolutionary Algorithm

A multi-objective optimization problem can be formulated as follows. Here, we omit the vector notation of $\{x, p, q\}$ and F to denote a multi-dimensional point or objective vector.

$$\begin{aligned} \text{Minimize} \quad & F(x) = (f_1(x), f_2(x), \ldots, f_M(x)) \\ \text{Subject to} \quad & g_j(x) \geq 0, \quad \forall j \in \{1, \ldots, J\} \\ & x \in \Omega \subseteq \mathbb{R}^n \quad \text{and,} \quad F \in \Lambda \subseteq \mathbb{R}^M \end{aligned} \tag{3}$$

Here, feasible variable space and respective feasible objective space are defined by Ω and Λ, respectively. The goal of this optimization is to find the best trade-off hyper-surface.

4.1 Proposed Trust Region Concept

We propose several modifications on the classical trust region method in order to make it applicable to metamodel-based multi-objective evolutionary algorithms:

1. We store all high fidelity solutions in an archive A, instead of replacing them with better solutions.
2. We maintain an independent trust region in the variable space for each solution. The regions may overlap with each other. They can either grow or shrink in size independently during optimization according to the quality of prediction. The algorithm restricts its search within the combined trust regions of A.
3. To compare a newly predicted point q with the neighbor point p (q is within trust region of p), we define two performance indicators PI that calculate r (analogous to Equation 2) for a multi-objective problem. Moreover, we propose a novel scheme to compare between feasible and infeasible solutions.
4. If the new point q is within the trust regions of multiple points $P \subseteq A$, then we update the trust radius δ_k for each of them using pair-wise performance metric (PI). The trust radius of point q will be the minimum of trust radii of P.

Thus, we optimize the following metamodel-based optimization to obtain a set of new infill points:

$$\begin{aligned} \text{Minimize}_{q\in\Omega} \quad & \hat{f}_1(q), \ldots, \hat{f}_M(q) \\ & \hat{g}_j(x) \geq 0, \quad \forall j \in \{1, \ldots, J\} \\ \text{Subject to} \quad & \|q - p\| \leq \delta_k^p, \quad \exists p \in A \end{aligned} \tag{4}$$

Here $p \in A$ are the exactly evaluated solutions from the current archive. Figure 1 illustrates the population based extension of the trust region method. Five exactly evaluated points $\{P_1, P_2, P_3, P_4, P_5\}$ with their trust regions (regions within the circles) are shown. Say, a new point P_{new} is predicted by the algorithm after optimizing on the model space. Note that P_{new} is inside the trust regions of P_1 and P_2. Assuming that the performance indicator reports an improvement of P_{new} over P_2, but no improvement over P_1. Then we reduce the size of the trust region of P_2 and increase that of P_1. The trust radius of the new point will be the smaller of the trust radii of P_1 and P_2.

4.2 Performance Indicators for Updating Trust Radius

To update the trust radius of solutions, we propose two performance indicators (PI).

Scalarization based Performance Indicator (PI_{ASF}): Scalarization method is used to convert a multi-objective problem into a number of parameterized single-objective optimization problems. We use the achievement scalarization function (ASF) [26] as a performance indicator. The scalarization is based on a weight vector w and a reference point z. The ASF formulation is given below:

$$\text{ASF}(x) = \max_{i=1}^{M} \frac{f_i(x) - z_i}{w_i}. \tag{5}$$

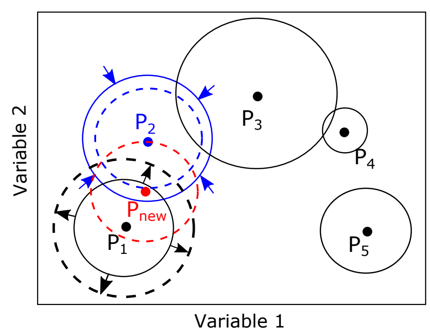


Fig. 1: Adaptive trust region concept for multiple solutions.

The proposed performance criteria using ASF function for trust radius update is presented as follows:

$$PI_{ASF}(q) = \frac{ASF(p) - ASF(q)}{ASF(p) - \widehat{ASF}(q)}. \tag{6}$$

Here $\widehat{ASF}$ is obtained from predicted objectives. The estimated improvement may differ for different reference directions.

Hypervolume based Performance Indicator (PI_{HV}): Hypervolume [10] is a widely used indicator in multi-objective optimization. It takes a set of solutions and a reference point, and computes the dominated region (in objective space) enclosed by the set and the reference point. In order to find the improvement of a new point over old point, we calculate the difference of their absolute hypervolume measures. We include archive points (A) as a common ground for computation. We then compute the ratio between actual improvement and predicted improvement and adjust the trust radii of old points. The predicted hypervolume is calculated by the objective values evaluated in model space using $\hat{F}(.)$. Since larger values indicate better hypervolume, we use negative of the hypervolume:

$$PI_{HV}(q) = \frac{HV(F(A) \cup F(q)) - HV(F(A))}{HV(F(A) \cup \hat{F}(q)) - HV(F(A))}. \tag{7}$$

Performance Indicator for Constrained Problems: We use constrained violation CV function [7], by accumulating violation of each constraint function ($g_j(x) \geq 0$), given as: $\mathrm{CV}(x) = \sum_{j=1}^{J} \langle \bar{g}_j(x) \rangle$, where the bracket operator $\langle \alpha \rangle$ for g is $-\alpha$ if $\alpha < 0$ and zero, otherwise. The functions $\bar{g}_j$ are the normalized version of constraint functions g_j [7].

$$PI_{CV}(q) = \frac{CV(G(p)) - CV(G(q))}{CV(G(p)) - CV(\hat{G}(q))} \tag{8}$$

Here, G and $\hat{G}$ are the vector representations of constraint functions $G = (g_1, \ldots, g_J)$ and $\hat{G} = (\hat{g}_1, \ldots, \hat{g}_J)$, respectively.

4.3 Overall Trust Region Adaptation

We now describe the procedure of updating the trust regions using the performance indicators described above. Assume that solution p is one of the high-fidelity points and q is the predicted new point which is within the trust region of p. We measure the performance improvement by the following equation.

$$r = \begin{cases} PI_{HV}(q) \text{ or } PI_{ASF}(q), & \text{if both } p \text{ and } q \text{ feasible},\\ r_2 + \epsilon, & \text{if } p \text{ infeasible, } q \text{ feasible},\\ r_1 - \epsilon, & \text{if } p \text{ feasible, } q \text{ infeasible},\\ PI_{CV}(q), & \text{otherwise.} \end{cases} \tag{9}$$

Here $\epsilon > 0 \in \mathcal{R}$ is a small positive number. The pre-defined positive constants $0 < r_1 < r_2 < 1$ are the hyper-parameters that regulate expansion and contraction of the trust regions. After estimating performance indicator PI of a new point q with respect to old point p we update trust radius of p by the following rule.

$$\delta^p_{k+1} = \begin{cases} c_1 \delta^p_k & \text{if } r < r_1\\ \min\{c_2 \delta^p_k, \Delta_{max}\} & \text{if } r > r_2\\ \delta^p_k & \text{otherwise} \end{cases} \tag{10}$$

The positive constants $0 < c_1 < 1$ and $c_2 > 1$ controls the size of subsequent trust radius. As mentioned earlier, we assign the trust radius of q to be the smaller of the trust radii of all neighboring solutions of which q is inside their trust regions. The parameter Δ_{max} is the largest allowed trust radius for the solutions.

5 Proposed Overall Algorithm

We now present trust region based algorithm for multi-objective optimization for low-budget problems. We refer our algorithm to TR-NSGA-II.

The overall procedure is described in Algorithm 1, the metamodeling algorithm starts with an archive of ρ initial population members created using the Latin hypercube sampling (LHS) method on the entire search space. The trust radii of initial solutions are then set to a predefined initial value δ_{init}. Thereafter, these solutions are evaluated exactly (high-fidelity) and metamodels are constructed for all M objectives ($\hat{f}_i(x); i = 1, \ldots, M$) and J constraints ($\hat{g}_j(x); j = 1, \ldots, J$). Then, a multi-objective evolutionary algorithm NSGA-II [6] with faster non-dominated sorting algorithm [20, 23] is run for τ generations starting with μ initial random solutions in model space. The NSGA-II algorithm returns $min(\mu, E - e)$ solutions where e is the current number of high-fidelity solution evaluations. The solutions are then evaluated using high-fidelity simulation and included in the archive (line 13). Then, new metamodels are then build from scratch and the process is repeated until termination. The trust radii are updated after each NSGA-II run, for new and old points according to the update rules discussed before. We have used both Hypervolume based and ASF based performance indicator alternatively for updating trust radius. ASF values are computed using reference point set W. PI_{ASF} is calculated using the best ASF values for the new solutions.

Algorithm 1: Trust Region Based Algorithm or TR-NSGA-II

Input : Obj: $[f_1, \ldots, f_m]^T$, Constr: $[g_1, \ldots, g_J]^T$, n (vars), ρ (sample size), E (max. high-fidelity SEs), NSGA-II (multi-obj EA) with pop-size μ, number of generation for model optimization τ, other parameters of NSGA-II Γ, Constraint violation function **CV**, Trust region parameters $\delta_{init}, \Delta_{max}, c_1, c_2, r_1$ and r_2

Output: Solution set P_T

1 $t, e \leftarrow 0$;
2 $P_t, F_t, G_t \leftarrow \emptyset$;
3 $P_{new} \leftarrow \text{LHS}(\rho, n)$`// Initial solutions`
4 $\delta^\ell \leftarrow \delta_{init}, \forall \ell \in \{1, \ldots, \rho\}$;
5 **while** *True* **do**
6 $\quad F^i_{new} \leftarrow f_i(P_{new}), \forall i \in \{1, \ldots, M\}$`// eval obj.`
7 $\quad G^j_{new} \leftarrow g_j(P_{new}), \forall j \in \{1, \ldots, J\}$`// eval constr.`
8 $\quad$**if** $t > 0$ **then**
9 $\quad\quad \widehat{F}^i_{new} \leftarrow \widehat{f}^i_t(P_{new}), \forall i \in \{1, \ldots, M\}$`// predicted`
10 $\quad\quad \widehat{G}^j_{new} \leftarrow \widehat{g}^j_t(P_{new}), \forall j \in \{1, \ldots, J\}$`// predicted`
11 $\quad\quad \delta \leftarrow \text{UPDATE\_TRUSTREGION}(F_t, \widehat{F}_{new}, G_t, \widehat{G}_{new}, \delta)$
12 $\quad$**end**
13 $\quad P_{t+1}, F_{t+1}, G_{t+1} \leftarrow (P_t \cup P_{new}), (F_t \cup F_{new})$ and $(G_t \cup G_{new})$;
14 $\quad e \leftarrow e + |P_{new}|$;
15 $\quad$**break** if $e \geq E$;
16 $\quad \widehat{f}^i_{t+1} \leftarrow \text{METAMODEL}(F^i_{t+1}), \forall i \in \{1, \ldots, M\}$`// metamodel obj.`
17 $\quad \widehat{g}^j_{t+1} \leftarrow \text{METAMODEL}(G^j_{t+1}), \forall j \in \{1, \ldots, J\}$`// metamodel constrt.`
18 $\quad P_{new} \leftarrow \text{NSGA-II}(\widehat{f}_{t+1}, \widehat{g}_{t+1}, \mu, \tau, \Gamma, E - e, \mathbf{CV}, \delta)$; `// Optimize model space`
19 $\quad t \leftarrow t + 1$;
20 **end**
21 **return** $P_T \leftarrow$ filter the best solutions from P_{t+1}

In one epoch $|W|$ solutions are returned directly by running NSGA-II while in another epoch we choose $|W|$ solutions (after running NSGA-II) such that they minimizes ASF according to $w \in W$ reference directions. In the end, trust regions are updated according to Hypervolume and ASF respectively. The major steps of this method are outlined in Algorithm 1.

6 Results

We present experimental results obtained by running four different optimization algorithms. We refer our algorithm as TR-NSGA-II. We compare the proposed algorithm with three other baseline algorithms: a) M1-2 [9] which works similar to our TR-NSGA-II (Algorithm 1) but without the trust region, and b) state-of-the-art multi-objective evolutionary method NSGA-II [8] and recently proposed K-RVEA [4]. We got source code of K-RVEA from the authors. The code currently doesn't handle constraints, thus we don't apply it to constrained problems. In NSGA-II, we use the binary tournament selection operator, simulated binary crossover (SBX), and polynomial mutation with parameters as follows: population size = $10n$, where n is a number of variables, number of generations = 100, crossover probability = 0.95, mutation probability = $1/n$, distribution index for SBX operator = 15, and distribution index for polynomial mutation operator = 20. The NSGA-II procedure, wherever used, uses the same parameter values. Initial trust radius is $\delta_{init} = 0.75\Delta_{max}$ for all problems, where $\Delta_{max} = \sqrt{n}$ is the largest diagonal of an n-dimensional unit hypercube. We take $c_1 = 0.75, c_2 = 1.10, r_1 = 0.9, r_2 = 1.05$ for all the problems. All the distances calculated here are in the normalized space. We perform 11 runs for each algorithm on all test and engineering design problems.

For NSGA-II, we have used population size 20 to maximize the evolution effect and that provided the best results for these low-budget problems. Other parameters are kept identical across all algorithms to provide a representative performance of each algorithm. Median IGD values and p-values of Wilcoxon rank sum test are provided in Table 1.

6.1 Two-Objective Unconstrained Problems

First, we apply our proposed method to two-objective unconstrained problems ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6 with ten ($n = 10$) variables, $|W| = 21$ reference directions, and with a maximum of only $E = 500$ high-fidelity solution evaluations. The obtained non-dominated solutions are shown in Figure 3(a)-(e). It is evident from the figure that trust region method with hypervolume perform better than method M1-2 without trust region. Because of the lack of enough solution evaluations, NSGA-II could not converge enough to these problems. On the contrary, trust region based methods provide increased accuracy (for example TR-NSGA-II has IGD 0.00121 compared to 0.01161 of M1-2 for ZDT1) for these test problems. K-RVEA performed the third best best in ZDT1 with IGD = 0.07964. TR-NSGA-II performs the best for all ZDT problems both in terms of GD and IGD. For ZDT4, all the methods find it hard to converge and perform equivalently (with p-value 0.05) except NSGA-II. For ZDT6, K-RVEA has better GD value but TR-NSGA-II has a better distribution (IGD=0.31070).

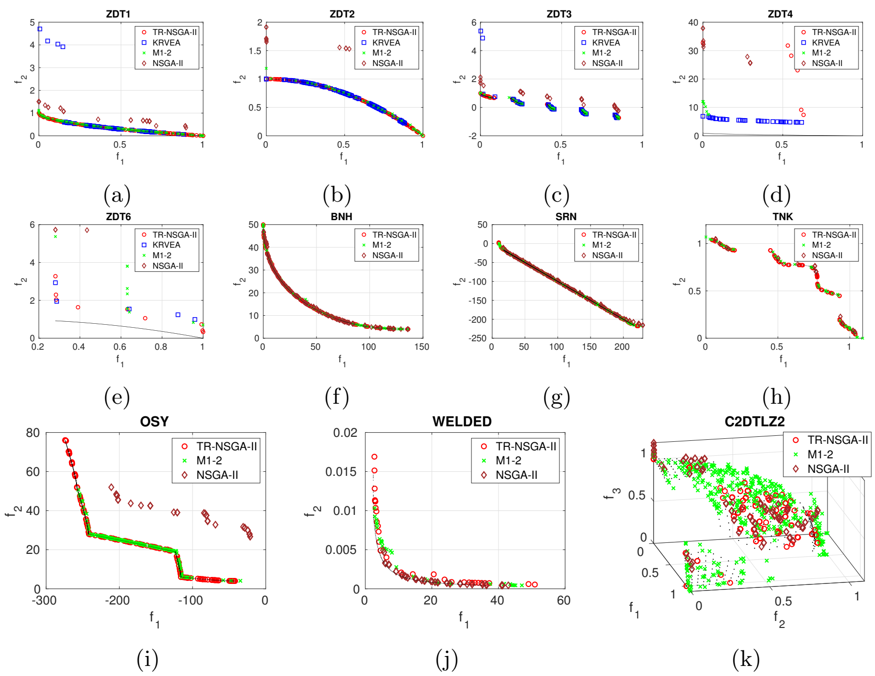


Fig. 3: Obtained non-dominated solutions of median run for 11 test problems using four different algorithms are shown.

6.2 Two-Objective Constrained Problems

Next, we apply our algorithms to two-objective constrained problems: BNH, SRN, TNK, and OSY [8], For each problem, we use $|W| = 21$ reference directions and a total of 500 solution evaluations. The obtained non-dominated solutions are shown in Figure 3(f)-(i). With the trust region method, we find better convergence as well as diversity for OSY and TNK, although no extra effort has been made to maintain diversity. BNH, SRN and TNK have only two variables and two constraints. NSGA-II, along with all other methods, performs well in BNH and SRN. We have achieved increased accuracy (IGD 0.00141 compared to 0.01543 of M1-2) for TNK problem. OSY is a difficult problem with six variables and six constraints. But our proposed method is able to find a good distribution on the true Pareto-front with only 500 solution evaluations with better IGD and GD. In SRN, method M1-2 performs the best in terms of both GD and IGD.

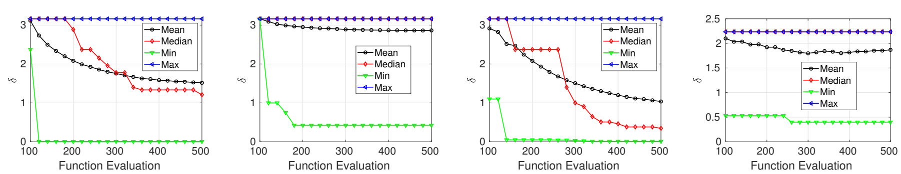
6.3 Three-Objective and Real-world Problems

We have applied three methods (except K-RVEA) to three-objective constrained problem C2DTLZ2 (Figure 3(k)). For C2DTLZ2, M1-2 without trust region performs the best. Trust region based method TR-NSGA-II performs the second best. Due to restricted search region in 3-dimensional space, our method suffers from premature convergence. We also apply our algorithm to a real-world welded beam design problem. Surprisingly, with the optimum population size, NSGA-II performs better in terms of GD, whereas our method has the best IGD.

Table 1: IGD values for 11 test problems are computed. Best algorithm and other statistically similar methods are marked in bold.

Problem/Method	NSGA-II		M1-2		TR-NSGA-II		K-RVEA	
	IGD	GD	IGD	GD	IGD	GD	IGD	GD
ZDT1	0.27131	0.34582	0.01161	0.01091	**0.00121**	**0.00122**	0.07964	0.03715
	p=1.852e-05	p=1.852e-05	p=7.7801e-04	p=7.4613e-04	-	-	p=1.852e-05	p=1.852e-05
ZDT2	0.98265	0.61637	0.00975	0.00755	**0.00057**	**0.00081**	0.03395	**0.00080**
	p=1.852e-05	p=1.852e-05	p=1.852e-05	p=1.852e-05	-	p=0.2851	p=1.852e-05	-
ZDT3	0.32080	0.38940	0.01251	0.00761	**0.00870**	**0.00230**	0.02481	0.00650
	p=1.852e-05	p=1.852e-05	p=1.852e-05	p=1.852e-05	-	-	p=1.852e-05	p=1.802e-04
ZDT4	25.24040	34.43350	**7.11881**	**10.10851**	**6.97620**	**12.92170**	**4.33221**	**4.50901**
	p=1.852e-05	p=1.852e-05	p=0.7928	p=0.1007	p=0.8955	p=0.2934	-	-
ZDT6	5.00571	4.80922	1.55861	2.27535	**0.31070**	2.84941	0.65462	**1.50551**
	p=1.852e-05	p=1.852e-05	p=1.852e-05	p=0.001	-	p=0.0151	p=1.852e-05	-
BNH	0.78981	0.19842	0.45272	0.13696	**0.09651**	**0.09092**	-	-
	p=1.852e-05	p=1.852e-05	p=1.852e-05	p=1.852e-05	-	-		
SRN	1.66162	2.11235	**0.67285**	**0.75337**	1.44045	1.74951	-	-
	p=1.852e-05	p=1.852e-05	-	-	p=1.852e-05	p=1.852e-05		
TNK	0.04182	0.01341	0.01543	0.01008	**0.00141**	**0.00201**	-	-
	p=1.852e-05	p=1.852e-05	p=1.852e-05	p=1.852e-05	-	-		
OSY	35.80211	27.43991	4.78063	0.59202	**0.16731**	**0.25063**	-	-
	p=1.852e-05	p=1.852e-05	p=1.852e-05	p=1.852e-05	-	-		
Welded Beam	1.10272	**0.21092**	0.92692	1.68806	**0.07681**	1.72811	-	-
	p=1.852e-05	-	p=1.8267e-04	p= 0.0042	-	p=0.0012		
C2DTLZ2	0.13733	0.04792	**0.03355**	**0.02373**	0.06411	0.02991	-	-
	p=1.852e-05	p=1.852e-05	-	-	p=1.8267e-04	p=1.8267e-04		

Median IGD values of 11 runs of 11 test problems are presented in Table 1 for all the algorithms. The table demonstrates that trust region methods perform usually better than non-trust region based methods whenever solutions reach to near Pareto-optimal front. It would be interesting to incorporate our method to other recently proposed multi-objective evolutionary algorithms including K-RVEA.



ZDT1 ZDT3 ZDT4 ZDT6

Fig. 5: Trust region adaptation for evolving population is presented. Minimum, median, average and maximum δ values during the optimization are shown.

6.4 Dynamics of Trust Region Adaptation

In Figure 5, we investigate the dynamics of trust region adaptation for the evolving population in different test problems. The value of δ starts from the $\sqrt{n}$ where n is number of variables. The maximum value remains the same for most ZDT problems because the obtained non-dominated solutions go beyond these regions after some epochs. In contrast, minimum, median and mean values are always decreasing throughout the optimization process. As discussed before, based on the improvement of the neighboring solutions, the regions are either expanded or contracted. In order to increase the trust region, the evolving population has to maintain $r_2 = 1.05$ or 5% Hypervolume improvement over previous generations. In general, this condition is hard to meet when solutions are converged in the end. Therefore, our method focus more on exploitation in the last stage of optimization.

7 Conclusions

In this paper, we have presented an adaptive trust region concept for multi-objective optimization with a low budget of solution evaluations. Trust regions are used as a constraint in the variable space during optimization to deal with uncertainties of metamodels. This study makes three main contributions: First, we have proposed two performance indicators based on scalarization and hypervolume to adapt appropriate trust regions. Second, a constraint handling scheme is presented in order to handle the trust region adaptation in the presence of constraints. Third, since multi-objective optimization aims to find a set of Pareto-optimal solutions, we need to manage multiple trust regions with multiple trade-off solutions compare to single best solution, as proposed in the classical literature. Our results on several test and one engineering design problems have shown that we can achieve better convergence using the proposed method than that without a trust region. While other MOEAs spend thousands of function evaluations, our trust region based method can solve test and real-world problems with limited budget yet with increased accuracy.

The current study has introduced some new parameters, such as the initial trust radius and their updating factors. Although our experiments are based on reasonable parameter settings, a detailed parameter study is a good starting point for future research. Moreover, other distance metrics besides the Euclidean norm can be used to define a trust region. Also, it needs to be ensured that the trust region concept scales up for problems with high-dimensional variable, objective, and constraint spaces. Nevertheless, this pilot first study has made one aspect of metamodeling task for multi-objective optimization clear – a balance between a trust of metamodels around high-fidelity points and progress of the overall search is essential for an efficient application.

References

1. Alexandrov, N.M., Dennis, J.E., Lewis, R.M., Torczon, V.: A trust-region framework for managing the use of approximation models in optimization. Structural optimization **15**(1), 16–23 (1998)
2. Bhattacharjee, K.S., Singh, H.K., Ray, T.: Multi-objective optimization with multiple spatially distributed surrogates. Journal of Mechanical Design **138**(9) (2016)
3. Bhattacharjee, K.S., Singh, H.K., Ray, T., Branke, J.: Multiple surrogate assisted multiobjective optimization using improved pre-selection. In: IEEE CEC (2016)
4. Chugh, T., Jin, Y., Miettinen, K., Hakanen, J., Sindhya, K.: A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization. IEEE Trans. on Evol. Comp. **22**(1), 129–142 (2018)

5. Deb, K., Hussein, R., Roy, P.C., Toscano, G.: A taxonomy for metamodeling frameworks for evolutionary multi-objective optimization. IEEE Transactions on Evolutionary Computation (in press)
6. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE Trans. on Evol. Comp. **6**(2), 182–197 (Apr 2002)
7. Deb, K.: An efficient constraint handling method for genetic algorithms. Computer methods in Applied Mechanics and Engineering (2000)
8. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. John Wiley & Sons, Inc., New York, NY, USA (2001)
9. Deb, K., Hussein, R., Roy, P., Toscano, G.: Classifying metamodeling methods for evolutionary multi-objective optimization: First results. In: EMO. Springer (2017)
10. Emmerich, M.T.M., Deutz, A.H., Klinkenberg, J.W.: Hypervolume-based expected improvement: Monotonicity properties and exact computation. In: 2011 IEEE Congress of Evolutionary Computation (CEC). pp. 2147–2154 (2011)
11. Emmerich, M.T.M., Giannakoglou, K.C., Naujoks, B.: Single- and multiobjective evolutionary optimization assisted by gaussian random field metamodels. IEEE Transactions on Evolutionary Computation **10**(4), 421–439 (2006)
12. Fliege, J., Vaz, A.I.F.: A method for constrained multiobjective optimization based on sqp techniques. SIAM J. on Optim. **26**(4), 2091–2119 (2016)
13. Hussein, R., Deb, K.: A generative kriging surrogate model for constrained and unconstrained multi-objective optimization. In: GECCO'2016. ACM Press (2016)
14. Jin, Y.: Surrogate-assisted evolutionary computation: Recent advances and future challenges. Swarm and Evolutionary Computation **1**(2), 61–70 (2011)
15. Knowles, J.: Parego: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. IEEE Trans. on Evol. Compt. (2006)
16. Miettinen, K.: Nonlinear Multiobjective Optimization. Kluwer, Boston (1999)
17. Pedrielli, G., Ng, S.: G-STAR: A new kriging-based trust region method for global optimization. IEEE Press, United States (1 2017)
18. Ponweiser, W., Wagner, T., Biermann, D., Vincze, M.: Multiobjective optimization on a limited budget of evaluations using model–assisted S–metric selection. In: Parallel Problem Solving from Nature–PPSN X, pp. 784–794. Springer (2008)
19. Roy, P., Deb, K.: High dimensional model representation for solving expensive multi-objective optimization problems. In: IEEE CEC. pp. 2490–2497 (2016)
20. Roy, P.C., Deb, K., Islam, M.M.: An efficient nondominated sorting algorithm for large number of fronts. IEEE Transactions on Cybernetics pp. 1–11 (2018)
21. Roy, P., Hussein, R., Deb, K.: Metamodeling for multimodal selection functions in evolutionary multi-objective optimization. In: GECCO'2017. ACM Press (2017)
22. Roy, P.C., Blank, J., Hussein, R., Deb, K.: Trust-region based algorithms with low-budget for multi-objective optimization. pp. 195–196. GECCO, ACM (2018)
23. Roy, P.C., Islam, M.M., Deb, K.: Best order sort: A new algorithm to non-dominated sorting for evolutionary multi-objective optimization. GECCO (2016)
24. Ryu, J.H., Kim, S.: A derivative-free trust-region method for biobjective optimization. SIAM Journal on Optimization **24**(1), 334–362 (2014)
25. Viana, F.A.C., Haftka, R.T., Watson, L.T.: Efficient global optimization algorithm assisted by multiple surrogate techniques. Journal of Global Optimization (2013)
26. Wierzbicki, A.P.: The use of reference objectives in multiobjective optimization. In: Multiple crit. decision making theory and appln., pp. 468–486. Springer (1980)
27. Zhang, Q., Liu, W., Tsang, E., Virginas, B.: Expensive Multiobjective Optimization by MOEA/D With Gaussian Process Model. IEEE Trans on Evol. Comp. **14**(3) (2010)