On the Exploitation of Search History and Accumulative Sampling in Robust Optimisation

Khulood Alyahya
K.Alyahya@exeter.ac.uk
University of Exeter, EX4 4QF, UK

Kevin Doherty
K.Doherty@exeter.ac.uk
University of Exeter, EX4 4QF, UK

Jonathan E. Fieldsend*
J.E.Fieldsend@exeter.ac.uk
University of Exeter, EX4 4QF, UK

Ozgur E. Akman
O.E.Akman@exeter.ac.uk
University of Exeter, EX4 4QF, UK

ABSTRACT

Efficient robust optimisation methods exploit the search history when evaluating a new solution by using information from previously visited solutions that fall in the new solution's uncertainty neighbourhood. We propose a full exploitation of the search history by updating the robust fitness approximations across the entire search history rather than a fixed population. Our proposed method shows promising results on a range of test problems compared with other approaches from the literature.

CCS CONCEPTS

•**Theory of computation → Evolutionary algorithms;** •**Computing methodologies → Uncertainty quantification;**

KEYWORDS

Robust optimisation; uncertain optimisation; sampling bias.

ACM Reference format:
Khulood Alyahya, Kevin Doherty, Jonathan E. Fieldsend, and Ozgur E. Akman. 2017. On the Exploitation of Search History and Accumulative Sampling in Robust Optimisation. In *Proceedings of GECCO '17 Companion, Berlin, Germany, July 15-19, 2017,* 2 pages.
DOI: http://dx.doi.org/10.1145/3067695.3076060

1 INTRODUCTION

Robust designs are needed if, e.g., we cannot manufacture with complete accuracy, or there are a set of different scenarios which a design must operate in. This is distinct from coping with noisy problems, where there is error in the cost function itself. Algorithms have been developed for both robust and noisy problems [1, 6, 8].

Here we consider robust optimisation where we know the underlying distribution of our scenario sets and we seek a solution with the best expected fitness which we refer to as the *effective fitness*. For a given minimisation problem, $\min f(\mathbf{x})$, we seek a design $\mathbf{x}$ that minimises the function $f_{\text{eff}}(\mathbf{x}) = \int_{-\infty}^{\infty} f(\mathbf{x}+\delta)P(\delta)\mathrm{d}\delta$, where $P(\delta)$ is the probability of a disturbance, δ. Here a discrete set of designs in a predefined neighbourhood (the *disturbance set*) represents our scenario set, on which we estimate the effective fitness. The designs to include in the disturbance set could be selected by Monte-Carlo sampling, but this is computationally expensive. We may utilise previously evaluated designs visited during the search process, but as these are not independent and identically distributed (i.i.d.), they introduce bias into (and corrupt) the robust approximation.

*Corresponding author

This work was supported by the Engineering and Physical Sciences Research Council [grant number EP/N017846/1].

GECCO '17 Companion, Berlin, Germany

 978-1-4503-4939-0/17/07…$15.00
DOI: http://dx.doi.org/10.1145/3067695.3076060

2 PROPOSED ADVANCES

We combine elite accumulative sampling (EAS), used in noisy optimisation problems [4, 5, 7], with updating the robust fitness approximations across the entire search history (Algorithm 1). The effective fitness of a design is estimated by weighting previously evaluated points in its disturbance neighbourhood. We use the weighting approach proposed by Branke & Fei [2] based on the Wasserstein distance. When an elite member is chosen for re-sampling, a point in its disturbance neighbourhood is chosen for evaluation also following the method in [2]. This maximal use of information allows us to get better estimates for relatively little computational cost — often refining multiple solutions for each new location query.

3 EXPERIMENTAL DESIGN AND RESULTS

We empirically assess three versions of Algorithm 1 here. **UH:** updating history only, the lines from 23 to 28 in Algorithm 1 will not be executed. **EAS:** re-sampling only with no history updates (exploiting the ASA framework from [2]). the following lines will not be executed: 6, 21, and in line 27 only the elite that has been sampled is updated by the value of the new sampled design in its disturbance neighbourhood. **EAS+UH:** All the lines are executed.

We use the following set-up. Initially $10d$ designs, where d is the number of dimensions, are generated using Latin hypercube sampling. An elite set size of $\lambda = 20$. A run is for 2,500 cost function queries (including the initialisation). A crossover probability of 0.8 and simulated binary crossover [3]. A mutation rate of $1/d$ using Gaussian mutation, σ set to 10% of the range (rejection sampling for bound-violating proposals). We assess the performance on three test problems and also compare with the results in [2].

Each method is run 30 times, with $d = 5$. We denote our estimate of the robust optimal design as $\mathbf{x}_{\text{final}}$. We compute this as the mean design values of the final ten best elite solutions, i.e., $\mathbf{x}_{\text{final}} = \frac{1}{10}\sum_{i=1}^{10}\mathbf{x}_i$. The effective fitness $f_{\text{eff}}(\mathbf{x}_{\text{final}})$ is then calculated by numerically integrating over the neighbourhood of $\mathbf{x}_{\text{final}}$ using

Algorithm 1 Elite Accumulative Sampling with History Updates

Require: r *Number of random initial samples*
Require: P_{cross} *Probability of crossover*
Require: λ *Number of elite solutions to consider*
Require: max_evals *Maximum number of fitness evaluations*
Require: $N(\mathbf{x})$ *Disturbance neighbourhood of* $\mathbf{x}$
Require: $P(\delta)$ *Probability distribution of disturbances* δ*, where* $\mathbf{u} \in N(\mathbf{x}) \iff P(\delta = |\mathbf{x} - \mathbf{u}|) > 0$
1: $H \leftarrow \emptyset$ *Ordered set, w.r.t.* $\hat{f}_{\mathit{eff}}(\mathbf{x})$*, of all evaluated locations*
2: $X \leftarrow$ `initial_samples`(r) *Generate* r *random samples*
3: **for all** $\mathbf{x} \in X$ **do**
4: $y \leftarrow$ `evaluate`$(\mathbf{x})$ *Evaluate* $\mathbf{x}$
5: $\hat{y}_{\text{eff}} \leftarrow$ `estimate`$_{f_{\text{eff}}}(\mathbf{x}, H)$ *Calculate* $\hat{f}_{\mathit{eff}}(\mathbf{x})$ *using* $N_H(\mathbf{x}) = \{\mathbf{u} \in H \mid \mathbf{u} \in N(\mathbf{x})\}$*, weighted by Wasserstein distance.*
6: `update_history`$(H, (\mathbf{x}, y))$ *update* $\hat{f}_{\mathit{eff}}$ *of* $N_H(\mathbf{x})$
7: $H \leftarrow H \cup \{(\mathbf{x}, y, \hat{y}_{\text{eff}})\}$ *Add* $\mathbf{x}$ *to history*
8: **end for**
9: $evals \leftarrow r$ *Track evaluations expended*
10: **while** $evals < max\_evals$ **do**
11: $\{\mathbf{v}, \mathbf{u}\} \leftarrow$ `select_from_elite`(H) *Select two elite members*
12: **if** $uniform\_rand() < P_{cross}$ **then**
13: $\mathbf{x} \leftarrow$ `crossover`$(\mathbf{v}, \mathbf{u})$ *Create a single child via crossover*
14: **else**
15: $\mathbf{x} \leftarrow \mathbf{v}$ *Copy first parent for later mutation if no crossover*
16: **end if**
17: $\mathbf{x}' \leftarrow$ `mutate`$(\mathbf{x})$ *Mutate* $\mathbf{x}$
18: $y' \leftarrow$ `evaluate`$(\mathbf{x}')$ *Evaluate* $\mathbf{x}'$
19: $evals \leftarrow evals + 1$
20: $\hat{y}'_{\text{eff}} \leftarrow$ `estimate`$_{f_{\text{eff}}}(\mathbf{x}', H)$ *See description on line 5*
21: `update_history`$(H, (\mathbf{x}', y'))$ *update* $\hat{f}_{\mathit{eff}}$ *of* $N_H(\mathbf{x}')$
22: $H \leftarrow H \cup \{(\mathbf{x}', y', \hat{y}'_{\text{eff}})\}$ *Add* $\mathbf{x}'$ *to history*
23: $\mathbf{x}^* \leftarrow$ `get_best_elite`(H) *Get the best elite*
24: $\mathbf{x}'' \leftarrow$ `resample_best_elite`$(\mathbf{x}^*, N(\mathbf{x}^*), N_H(\mathbf{x}^*))$ *Sample* $\mathbf{x}'' \in N(\mathbf{x}^*)$ *according to the Wasserstein method in [2].*
25: $y'' \leftarrow$ `evaluate`$(\mathbf{x}'')$ *Evaluate* $\mathbf{x}''$
26: $\hat{y}''_{\text{eff}} \leftarrow$ `estimate`$_{f_{\text{eff}}}(\mathbf{x}'', H)$ *See description on line 5*
27: `update_history`$(H, (\mathbf{x}'', y''))$ *update* $\hat{f}_{\mathit{eff}}$ *of* $N_H(\mathbf{x}'')$
28: $H \leftarrow H \cup \{(\mathbf{x}'', y'', \hat{y}''_{\text{eff}})\}$ *Add* $\mathbf{x}''$ *to history*
29: **end while**

Table 1: Effective fitness: mean (standard deviation).

	TP 1	TP 2	TP 3
SEM	0.899 (0.0314)	−4.128 (0.0434)	3.357 (0.0945)
SEM+AR	0.539 (0.0069)	−4.185 (0.0420)	2.837 (0.0939)
ABRSS	0.889 (0.0451)	−4.329 (0.0388)	2.557 (0.0717)
LHS+ASA	0.527 (0.0013)	−4.423 (0.0362)	2.308 (0.0297)
UH	0.526 (0.0073)	−3.582 (0.4584)	4.337 (0.4714)
EAS	0.524 (0.0052)	−4.155 (0.4525)	2.224 (0.2499)
EAS+UH	0.523 (0.0049)	−4.023 (0.3504)	2.108 (0.0187)

global adaptive quadrature [9] with 1e−10 absolute error tolerance. The estimated $f_{\text{eff}}(\mathbf{x})$ is denoted by $\hat{f}_{\text{eff}}(\mathbf{x})$. The means and standard deviations of $f_{\text{eff}}(\mathbf{x}_{\text{final}})$ for the three methods are displayed in Table 1, alongside the results from [2]. We ran a Wilcoxon signed-rank test with the Bonferroni Correction between each pair of our methods for each test problem (significance level $p < 0.05/3 = 0.0167$). All three proposed methods perform well on TP1, are able to find the robust optimum, stay away from the original (deceptive) global optimum, and are not significantly different from each other in performance. On TP2 both EAS and EAS+UH perform better than UH alone. However, the difference between EAS and EAS+UH is not statistically significant. This is perhaps because in this particular problem the solutions are dispersed across the different optima, which may lead the history updating to have little effect. For TP3, both EAS and EAS+UH outperform UH. For this problem, updating the history does improve the performance of the optimisation significantly, as EAS+UH performs significantly better than EAS alone. Combining history updates with elite accumulative sampling (EAS+UH) always finds the exact robust optimum and performs better than the mean of LHS+ASA in all of the 30 runs.

4 CONCLUSION

We have demonstrated that elite accumulative sampling, a method previously used with success in noisy optimisation problems, can be effectively utilised for robust problems. In addition to this, we have investigated the effect of updating our robust fitness estimates for our entire history of solutions and we have shown that this can improve the performance of a robust optimisation algorithm as assessed by its application to a set of established robust test problems. We found that the relative performance of elite accumulative sampling and updating the history depend on the function to be optimised. On two of our three test problems (TP2 & TP3) elite accumulative sampling results in better performance than updating the history alone. However, in some cases, the combination of both can be very powerful (as in TP3). Interestingly, we found that on one of our test problems (TP1), just updating the history without any resampling results in good performance and is better even than the current state-of-the-art. On other problems (e.g. TP3) this approach performs very poorly in the absence of resampling.

REFERENCES

[1] H.-G. Beyer and B. Sendhoff. 2007. Robust optimization-A comprehensive survey. *Computer Methods in Applied Mechanics and Engineering* 196, 33-34 (2007), 3190–3218.
[2] J. Branke and X. Fei. 2016. Efficient Sampling When Searching for Robust Solutions. In *Parallel Problem Solving from Nature – PPSN XIV: 14th International Conference*. 237–246.
[3] K. Deb and R. B. Agrawal. 1994. *Simulated Binary Crossover for Continuous Search Space.* Technical Report IITK/ME/SMD-94027. Indian Institute of Technology, Kanpur, India.
[4] J. E. Fieldsend. 2015. Elite Accumulative Sampling Strategies for Noisy Multi-objective Optimisation. In *Evolutionary Multi-Criterion Optimization: 8th International Conference, EMO 2015*. 172–186.
[5] J. E. Fieldsend and R. M. Everson. 2015. The Rolling Tide Evolutionary Algorithm: A Multiobjective Optimizer for Noisy Optimization Problems. *IEEE Transactions on Evolutionary Computation* 19, 1 (2015), 103–117.
[6] Y. Jin and J. Branke. 2005. Evolutionary optimization in uncertain environments-a survey. *IEEE Transactions on Evolutionary Computation* 9, 3 (2005), 303–317.
[7] T. Park and K. R. Ryu. 2011. Accumulative Sampling for Noisy Evolutionary Multi-objective Optimization. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation (GECCO '11)*. 793–800.
[8] P. Rakshit, A. Konar, and S. Das. 2016. Noisy Evolutionary Optimization Algorithms-A Comprehensive Survey. *Swarm and Evolutionary Computation* (2016).
[9] L.F. Shampine. 2008. Vectorized adaptive quadrature in MATLAB. *J. Comput. Appl. Math.* 211, 2 (2008), 131 – 140.