How to Solve the Dilemma of Margin-Based Equality Handling Methods

Samineh Bagheri[1], Wolfgang Konen[1], Thomas Bäck[2]

[1]Department of Computer Science
TH Köln (University of Applied Sciences)
Steinmüllerallee 1
51643 Gummersbach
E-Mail: {wolfgang.konen, samineh.bagheri}@th-koeln.de,

[2]Department of Computer Science
Leiden University, LIACS,
2333 CA Leiden, The Netherlands
Email: T.H.W.Baeck@liacs.leidenuniv.nl

Abstract

Addressing black-box constrained optimization problems (COPs) in an efficient manner is a challenging task in real-world applications. Although population-based optimization heuristics including evolutionary strategies and genetic algorithms appear to be successful in absence of derivative information, they often require too many function evaluations which may not be affordable in practice. The surrogate-assisted technique SACOBRA [1, 2] aims to reduce the required number of function evaluations by building mathematical models for objective and constraint functions.

But for COPs with equality constraint(s), it is highly unlikely to find a fully feasible solution, because the feasibility ratio ρ of this type of problems is exactly zero. The feasibility ratio ρ is the probability that a point randomly chosen from the search space is feasible. In order to overcome this problem, many equality handling techniques consider a small feasibility margin ε and try to find the best solution within the given margin. In other words, they transform an equality constraint $h(\vec{x}) = 0$ into an inequality constraint $|h(\vec{x})| - \varepsilon \leq 0$.

The dilemma with most margin-based equality handling techniques is the fact that they often report 'solutions' better than the true optimum which are however infeasible in the equality constraint(s) with a constraint violation of order ε.

The equality-handling approach in SACOBRA [1] uses a decrementing margin. Moreover, a refine mechanism moves the solutions within the given margin towards the subspace of the equality constraint(s). The refine mechanism applies a conjugate-gradient optimizer on the surrogate, so that no extra function evaluation is imposed. In this work we show that SACOBRA with refine step finds solutions very close to the true optimum, needing only relatively few function evaluations. Furthermore, we show that SACOBRA is able to find a range of different solutions (a set of Pareto-optimal solutions minimizing both the objective function and the constraint violation) for a given margin.

1 Introduction

An optimization problem can be defined as the minimization of an objective function (fitness function) f subject to inequality constraint function(s) $g_1, \ldots, g_m$ and equality constraint function(s) $h_1, \ldots, h_r$:

$$\begin{array}{lll} \text{Minimize} & f(\vec{x}), & \vec{x} \in [\vec{a}, \vec{b}] \subset \mathbb{R}^d \\ \text{subject to} & g_i(\vec{x}) \leq 0, & i = 1, 2, \ldots, m \\ & h_j(\vec{x}) = 0, & j = 1, 2, \ldots, r \end{array} \tag{1}$$

where $\vec{a}$ and $\vec{b}$ define the lower and upper bounds of the search space (a hypercube). By negating the fitness function f a maximization problem can be transformed into a minimization problem without loss of generality.

The most common way of tackling equality constraints in a black-box manner is to consider a small artificial feasibility margin of size ε_0 as it is done in several works [7, 11, 12]. Differential evolution optimizers [6, 14] often use a decremental margin ε which gradually shrinks to ε_f. The same mechanism is found in swarm optimizers [13]. Other approaches transform an equality constraint into two inequalities [5] or they consider one side of the the equality constraint as the artificial feasible area [10]. They do not need any feasibility margin, but they cannot solve all types of equality COPs [1, Fig. 2].

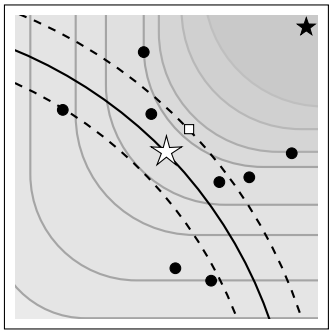

Figure 1: The shaded contours depict the objective function f (darker = smaller). The black curve shows the equality constraint. Feasible solutions are restricted to this line. The black star shows the global optimum of the objective function which is different from the optimum of the constrained problem shown as the white star $\vec{x}^*$. The area enclosed by the dashed curves is the artificially feasible area, given a fixed feasibility margin ε_0. The white square marks the optimal solution $\vec{x}^*_{af}$ in the artificial feasible region. Black dots mark some infill points.

Other works [3, 4] use analytical information about the equality constraints to transform the optimization problem into a feasible subspace where the equality constraints are implicit. But the absence of such analytical information in black-box optimization does not allow us to solve the problem in a feasible subspace.

Now we describe in more detail the dilemma of margin-based equality COP solvers: Once there is a feasibility margin greater than zero, we have the following dilemma: Each COP solver has a whole set of possible solutions to choose from. Should we prefer a solution with better objective value but larger violation of the true equality constraint over another one with worse objective value but smaller violation? There is no clear answer to this, it is a multi-criteria problem. Nonetheless, most of the numerical constrained optimizers are designed to search for the solution with the best objective value $\vec{x}^*_{af}$ inside the artificially feasible region and they avoid approaching the true optimum $\vec{x}^*$ (see Fig. 1). The distance between these two solutions $\vec{x}^*_{af}$ and $\vec{x}^*$ can be pretty large, both in input space and objective space.[4] Of course the dilemma would

[4]The distances depend on ε_f and the steepness of the equality functions $h_j(\vec{x})$ and the objective function $f(\vec{x})$.

disappear if ε_f approaches zero, but many optimizer have problems with too small margins (they may not find feasible solutions).

Another aspect of the dilemma is this: The G-function suite used in the CEC2006 competition [8] is a set of COPs which is commonly used as a benchmark for constrained optimizers. Most of the solvers assume a feasibility margin of size $\varepsilon_f = 10^{-4}$ and, as a result, they often report solutions with an objective better than the optimal value for problems with equality constraints. The error measure in objective space becomes negative, which is not very logical. Other optimizers (among them SACOBRA) tend to stick closer to the true optimum, of course with a slightly worse objective value. This makes it difficult to compare solvers for equality COPs. A possible way out is to show and compare a **set** of solutions in the multi-objective space {objective function, constraint violation}.

SACOBRA also uses a decrementing margin strategy. But additionally, each solution will be pushed in the direction of the true feasible subspace by means of a refine mechanism.

This work aims to show the benefits of using the refine step in approaching the true optimum. Moreover, we want to emphasize the importance of reporting a **set** of solutions (Pareto set) instead of one final solution. This should become a good practice for benchmarking optimizers on equality-constrained problems.

This paper is organized as follows: Sec. 2 briefly describes the equality-handling procedure in SACOBRA. After explaining in Sec. 3 the experimental setup, we show in Sec. 4 the impact of the refine step on solving several G-problems by representing a set of solutions in the neighborhood of the optimum. Sec. 5 summarizes the paper.

2 Methods

SACOBRA is a surrogate-assisted constrained optimizer which takes advantage of radial basis function interpolation techniques to reduce the expensive evaluations of objective and constraint functions [2].

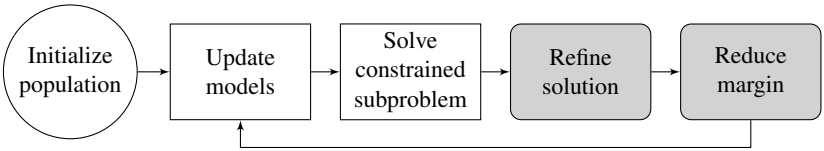

Figure 2: Main steps of SACOBRA with the equality handling mechanism.

SACOBRA tends to approach the optimal solution iteratively. After generating a population of initial points, SACOBRA builds surrogate models for the objective and constraint functions. Based on solving a so-called *internal COP* on the surrogate models, SACOBRA suggests the next infill point (the optimum returned from the internal COP). The new infill point will be evaluated on the real functions and added to the population of points, so the models can be updated. The algorithm goes through this loop until the budget is exhausted, as depicted in Fig. 2. Handling equalities in SACOBRA has two main ingredients: I. a decreasing feasibility margin, II. a refine step which aims to move the solutions towards the subspace of equality constraint(s).

2.1 Decrementing Margin

The zero-volume feasible space attributed to the j-th equality constraint $h_j(\vec{x}) = 0$ is expanded to a tube-shaped region around it: $|h_j(\vec{x})| - \varepsilon^{(n)} \leq 0$ by the proposed algorithm. By gradually reducing the margin $\varepsilon^{(n)}$ the solutions are smoothly guided towards the real feasible subspace. The equality margin ε can be reduced in different fashions described in [1]. We use a simple exponential decay scheme. In each iteration n the feasibility margin will be reduced as follows:

$$\varepsilon^{(n+1)} = \max(\varepsilon_f,\ \varepsilon^{(n)}\beta), \tag{2}$$

where the lower bound ε_f is often set to a small value close to machine accuracy.

2.2 Refine Mechanism

The **refine step** is done by minimizing the squared sum of all equality constraint violations by means of a conjugate-gradient (CG) method. This minimization step as described in Eq. (3) is done based on the surrogates of the equality constraints and not the real equality functions; therefore, no extra real function evaluation are imposed by this step.

$$\text{Minimize} \quad \sum_{j=1}^{r} (s_j^{(n)}(\vec{x}))^2, \qquad \vec{x} \in [\vec{a}, \vec{b}] \subset \mathbb{R}^d, \tag{3}$$

where $s_j^{(n)}(\vec{x})$ is the surrogate of the j-th equality. Although in black-box COPs we do not know the analytical form of the equalities, the refine step tries to move the best found solution in each iteration towards the feasible subspace with assistance of the estimated models of the equality functions $h(\vec{x})$. Furthermore, the refine step can be helpful in not losing a good feasible solution when decreasing the margin after an iteration. More algorithmic details can be found in [1].

3 Experimental Setup

We use $4 \cdot d$ points generated by LHS to initialize SACOBRA, where d is the size of the variable space. According to [1] we know that SACOBRA is not very sensitive to the choice of the decaying factor β, but it performs best in $\beta \in [0.90, 0.94]$, therefore we use $\beta = 0.92$. The internal COPs are addressed by `Cobyla()` from the NLOPTR package in R [9]. The refine step is done with assistance of the `optim()` function from the STATS package, the method is set to L-BFGS-B and the maximum number of refine iterations is set to 10^4. A maximum of 400 function evaluations is permitted. SACOBRA sets the final feasibility margin to $\varepsilon_f = 10^{-8}$. We compare two different configurations of SACOBRA with and without the refine step and also a differential evolution (DE) strategy with automatic parameter adjustment as proposed by Brest [6].

Table 1: Characteristics of G-problems with equality constraints: d: dimension, LI: the number of linear inequalities, NI: number of nonlinear inequalities, LE: the number of linear equalities, NE: the number of nonlinear equalities, α: number of active constraints.

Fct.	d	type	LI	NI	LE	NE	α
G03	20	nonlinear	0	0	0	1	1
G05	4	nonlinear	2	0	0	3	3
G11	2	nonlinear	0	0	0	1	1
G13	5	quadratic	0	0	0	3	3

For DE results, we run our experiments using the DEOPTIMR package. The function `JDEoptim()` in DEOPTIMR applies the same adaptive equality margin found in [14] but with a more aggressive updating scheme. The final feasibility margin is set to $\varepsilon_f \in \{10^{-4}, 10^{-8}\}$ in different DE runs.

The characteristics of the G-problems used in this study is listed in Tab. 1. We select here those problems from the G-function suite that have equality constraints.

4 Results and Discussion

The result in Fig. 3 shows that SACOBRA with only 400 fe (function evaluations) is able to populate the Pareto front nicely (we take the points with and without refine together, which are both available within the same run). On the other hand, DE needs more function evaluations (3300 fe, upper plot) to come into the vicinity of the Pareto front, however, many DE points have a constraint violation larger than 10^{-4}. Only if we add more fe to DE (7800 fe, lower plot), then the DE points will more or less densely populate the region near the Pareto front.

Although the refine step is a very simple step, it is essential for our algorithm. As shown in Fig. 3, SACOBRA with refine is doing a better job in reducing the constraint violation in comparison with the SACOBRA without the refine step. On the one hand, the refine step helps us to move the best solution found in each margin towards the real feasible subspace in each iteration. Although a wrong approximation of the equality constraints in the early iterations can cause a

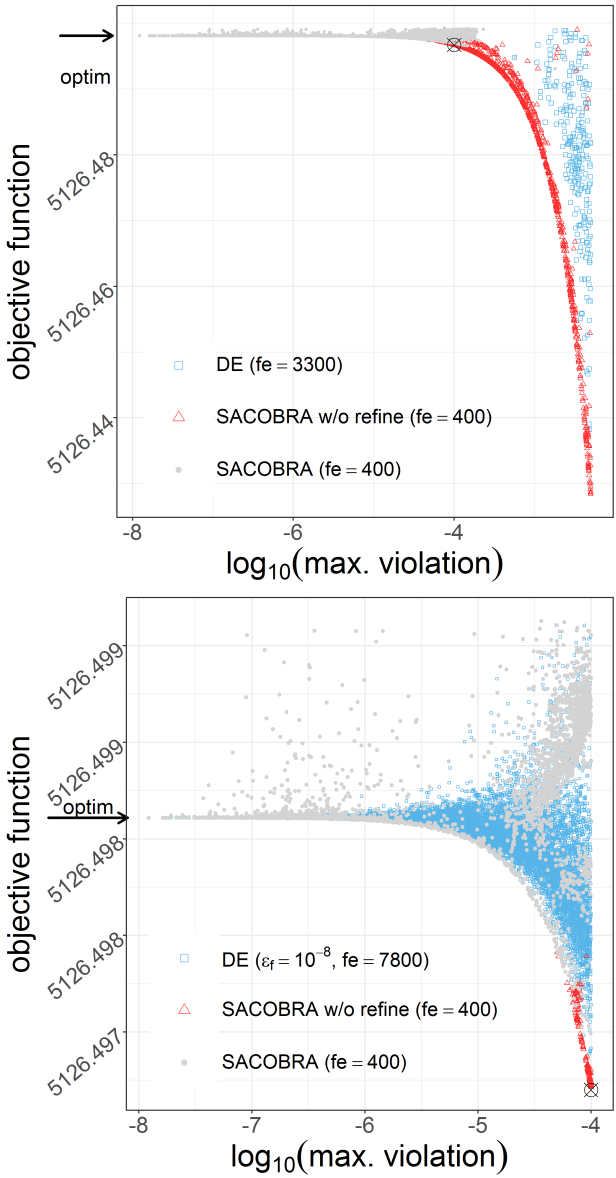

Figure 3: G05 problem: Infill points generated by different algorithms (DE, SACOBRA w/o refine, SACOBRA) during the optimization process. The final equality margin for all three algorithms is set to $\varepsilon_f = 10^{-8}$.

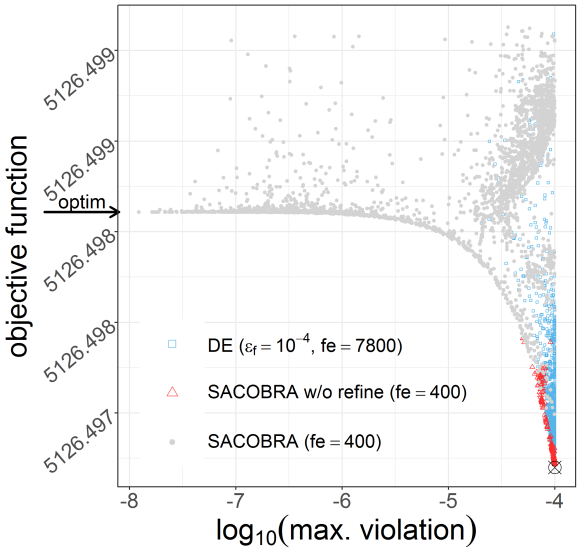

Figure 4: Same as Fig. 3, but for DE the feasibility margin is set to the value suggested by CEC2006 [8] $\varepsilon_f = 10^{-4}$.

shift towards more violated regions, the model(s) of equality constraint(s) gradually improve by learning about these regions and eventually the refine step can guide the solutions towards the correct direction. On the other hand, since only one new point will be added to the population in each iteration, usually this point will sit at the border of the artificial feasible region after the optimization step. If we now shrink the artificial feasible region without refining, we would lose this point and jump to another feasible point, if any, probably with a much larger objective value.

Comparing a set of found solutions instead of only the final one, helps us to have a better comparison for the performance of the equality constrained algorithms. Additionally, providing a set of solutions for COPs with equality constraints can be beneficial in practice, because the user then can decide to take a solution which fits best to his/her application.

Figs. 5–7 show similar results for the other G-problems. Tab. 2 shows the results from all runs in tabular form.

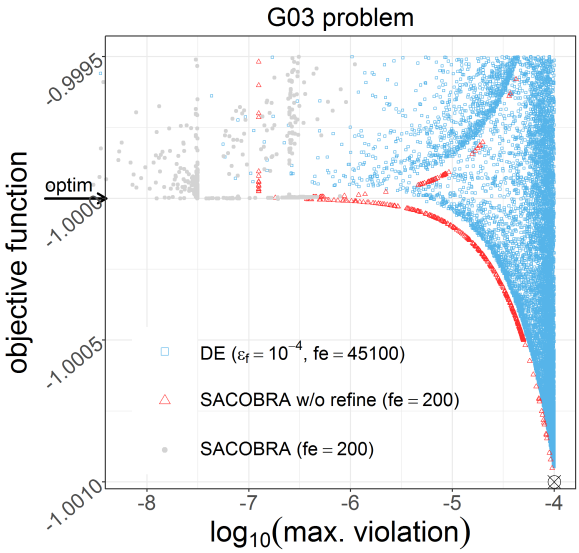

Figure 5: Same as Fig. 3, but for G03 problem.

Fig. 4 compares SACOBRA and DE when the final feasibility margin is set to $\varepsilon_f = 10^{-4}$ for DE. As expected DE even after thousands of function evaluations, converges to the optimum of the artificial feasible area $\vec{x}_{af}$.

5 Conclusion

In this work we have shown that SACOBRA is capable of approaching the true solution of equality-constrained optimization problems in less function evaluations than other optimizers. SACOBRA finds significantly better solutions in terms of constraint violation, vicinity to the optimum and efficiency, compared to DE, a well-known algorithm from evolutionary strategies. It avoids – at least to a large extent – the often seen dilemma of equality-constraint optimization that a margin leads to solutions 'better than the optimum' by violating some of the equality constraints. We have seen that the refine step – which may be also a useful building block for other optimization schemes – is essential for achieving this goal.

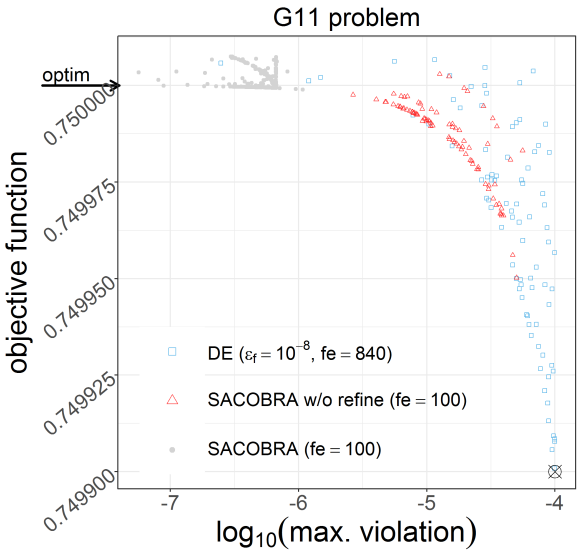

Figure 6: Same as Fig. 3, but for G11 problem.

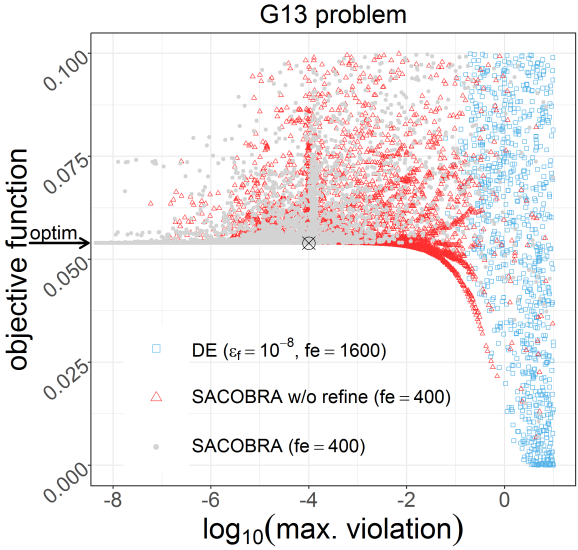

Figure 7: Same as Fig. 3, but for problem G13. As shown in Tab. 2 DE cannot find good solutions with even more function evaluations.

Table 2: Results for the different G-problems: true objective value, median and standard deviation of objective values from 30 runs of the solvers, mean and median (over 30 runs) of the maximum equality constraint violation, averaged number of function evaluations (fe). Note that DE was run here for only $200 \cdot d$ iterations on each problem, which leads to bad objective values in the case of G03 and G13. If more iterations were used for DE, it would find near optimal solutions for G03 but not for G13.

problem	trueObj	medObj	sd.obj	meanViol	medViol	fe
			SACOBRA			
G03	-1.0000	-1.0000	5e-07	5e-08	3e-08	200
G05	5126.4981	5126.4981	6e-04	1e-07	7e-08	400
G11	0.7500	0.7500	2e-05	5e-06	7e-07	100
G13	0.0539	0.0540	2e-01	1e-08	7e-09	400
			DE ($\varepsilon_f = 10^{-4}$, $200 \cdot d$ iterations)			
G03	-1.0000	-0.8728	4e-02	6e-05	7e-05	13124
G05	5126.4981	5126.4967	2e-06	1e-04	1e-04	7391
G11	0.7500	0.7499	6e-08	1e-04	1e-04	1902
G13	0.0539	0.4267	2e-01	4e-02	4e-02	2359
			DE ($\varepsilon_f = 10^{-8}$, $200 \cdot d$ iterations)			
G03	-1.0000	-0.8061	5e-02	4e-05	4e-05	12428
G05	5126.4981	5126.4981	4e-05	3e-07	2e-07	6612
G11	0.7500	0.7500	6e-11	1e-08	1e-08	2425
G13	0.0539	0.3670	2e-01	5e-02	5e-02	2274

Showing a set of best solutions minimizing both, maximum constraint violation and objective function, is helpful to have a fair comparison of different algorithms and also it is more practical for real-world applications because the user has a chance to select the solution which suits the application best.

References

[1] Bagheri, S., Konen, W., Bäck, T.: Equality constraint handling for surrogate-assisted constrained optimization. In: 2016 IEEE Congress on Evolutionary Computation (CEC). pp. 1924–1931 (July 2016)

[2] Bagheri, S., Konen, W., Emmerich, M., Bäck, T.: Self-adjusting parameter control for surrogate-assisted constrained optimization under limited budgets. Applied Soft Computing 61, 377 – 393 (2017)

[3] Beyer, H.G., Finck, S.: On the design of constraint covariance matrix self-adaptation evolution strategies including a cardinality constraint. IEEE Transactions on Evolutionary Computation 16(4), 578–596 (2012)

[4] Beyer, H.G., Finck, S., Breuer, T.: Evolution on trees: On the design of an evolution strategy for scenario-based multi-period portfolio optimization under transaction costs. Swarm and Evolutionary Computation 17, 74–87 (2014)

[5] Bhattacharjee, K.S., Ray, T.: A novel constraint handling strategy for expensive optimization problems. In: 11th World Congress on Structural and Multidisciplinary Optimization. Sydney, Australia (June 2015)

[6] Brest, J., Greiner, S., Boskovic, B., Mernik, M., Zumer, V.: Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. Trans. Evol. Comp 10(6), 646–657 (Dec 2006)

[7] Deb, K.: An efficient constraint handling method for genetic algorithms. Computer Methods in Applied Mechanics and Engineering 186(2), 311 – 338 (2000)

[8] Liang, J., Runarsson, T.P., Mezura-Montes, E., Clerc, M., Suganthan, P., Coello, C.C., Deb, K.: Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization. Journal of Applied Mechanics 41, 8 (2006)

[9] R Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2014), `http://www.R-project.org`

[10] Regis, R.G.: Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points. Engineering Optimization 46(2), 218–243 (2013)

[11] Runarsson, T.P., Yao, X.: Stochastic ranking for constrained evolutionary optimization. IEEE Transactions on Evolutionary Computation 4(3), 284–294 (2000)

[12] Runarsson, T.P., Yao, X.: Search biases in constrained evolutionary optimization. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 35(2), 233–243 (2005)

[13] Xie, X.F., Zhang, W.J., Bi, D.C.: Handling equality constraints by adaptive relaxing rule for swarm algorithms. Congress on Evolutionary Computation (CEC) pp. 2012–2016 (2004)

[14] Zhang, H., Rangaiah, G.: An efficient constraint handling method with integrated differential evolution for numerical and engineering optimization. Computers & Chemical Engineering 37, 74–88 (2012)