Proceedings Paper:

Johnson, V., Duro, J., Kadirkamanathan, V. et al. (1 more author) (2023) A distributed multi-disciplinary design optimization benchmark test suite with constraints and multiple conflicting objectives. In: Paquete, L., (ed.) Genetic and Evolutionary Computation Conference Companion (GECCO '23 Companion), July 15--19, 2023, Lisbon, Portugal. Genetic and Evolutionary Computation Conference (GECCO 2023), 15-19 Jul 2023, Lisbon, Portugal. Association for Computing Machinery , New York, NY, United States , pp. 1611-1619. ISBN 979-8-4007-0120-7

https://doi.org/10.1145/3583133.3596414

A Distributed Multi-Disciplinary Design Optimization Benchmark Test Suite with Constraints and Multiple Conflicting Objectives

Victoria Johnson
Automatic Control and Systems Engineering,
The University of Sheffield
Sheffield, United Kingdom
vjohnson2@sheffield.ac.uk

João A. Duro
Automatic Control and Systems Engineering,
The University of Sheffield
Sheffield, United Kingdom
j.a.duro@sheffield.ac.uk

Visakan Kadirkamanathan
Automatic Control and Systems Engineering,
The University of Sheffield
Sheffield, United Kingdom
visakan@sheffield.ac.uk

Robin C. Purshouse
Automatic Control and Systems Engineering,
The University of Sheffield
Sheffield, United Kingdom
r.purshouse@sheffield.ac.uk

ABSTRACT

Collaborative optimization (CO) is an architecture within the multi-disciplinary design optimization (MDO) paradigm that partitions a constrained optimization problem into system and subsystem problems, with couplings between them. Multi-objective CO has multiple objectives at the system level and inequality constraints at the subsystem level. Whilst CO is an established technique, there are currently no scalable, constrained benchmark problems for multi-objective CO. In this study, we extend recent methods for generating scalable MDO benchmarks to propose a new benchmark test suite for multi-objective CO that is scalable in disciplines and variables, called 'CO-ZDT'. We show that overly-constraining the number of generations in each iteration of the system-level optimizer leads to poor consistency constraint satisfaction. Increasing the number of subsystems in each of the problems leads to increasing system-level constraint violation. In problems with two subsystems, we find that convergence to the global Pareto front is very sensitive to the complexity of the landscape of the original non-decomposed problem. As the number of subsystems increases, convergence issues are encountered even for the simpler problem landscapes.

CCS CONCEPTS

• **Theory of computation → Optimization with randomized search heuristics**; • **Applied computing → Multi-criterion optimization and decision-making**.

KEYWORDS

collaborative optimization, multi-disciplinary design, constrained optimization

GECCO '23 Companion, July 15–19, 2023, Lisbon, Portugal

ACM ISBN 979-8-4007-0120-7/23/07…$15.00
https://doi.org/10.1145/3583133.3596414

ACM Reference Format:
Victoria Johnson, João A. Duro, Visakan Kadirkamanathan, and Robin C. Purshouse. 2023. A Distributed Multi-Disciplinary Design Optimization Benchmark Test Suite with Constraints and Multiple Conflicting Objectives. In *Genetic and Evolutionary Computation Conference Companion (GECCO '23 Companion), July 15–19, 2023, Lisbon, Portugal.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3583133.3596414

1 INTRODUCTION

To evaluate the performance of an optimization algorithm is it customary, if not compulsory, to employ benchmark problems. These problems are widely used in research to compare and evaluate the performance of different algorithms under the same conditions. By using benchmark problems, researchers can objectively evaluate the performance of optimization algorithms, identify their strengths and weaknesses, which subsequently facilitates the process of improving the design of the algorithm. Benchmark problems are typically designed to be challenging and complex, with multiple objectives, constraints, and many decision variables.

Several benchmark test suites have been proposed to evaluate the performance of algorithms in solving single- and multi-objective problems. BBOB [4] is a relatively recent test suite that contains many different types of problems, and it is organized into several variants, such as bbob-biobj with 55 bi-objective functions, and bbob-constrained with 10 single-objective functions with varying number of constraints. DTLZ [8] is another popular test suite that offers scalability in both in the number of decision variables and objectives, and it is useful in evaluating the performance of optimization algorithms in handling nonlinearity, non-convex Pareto fronts, and discontinuities. Another example is WFG [15] that offers test problems with highly customisable properties, such as non-separability, bias, multi-modality, and mixed Pareto-front shapes. Many of these characteristics, including the need to satisfy constraints, are found in many real-world problems, but existing benchmarks often lack multi-disciplinary design optimization (MDO) characteristics.

MDO problems require the optimization of a system that involves multiple disciplines, such as aerodynamic, structural, and powertrain optimization, that are often found in the automotive industry [22]. These types of problems involve complex interactions between different components and require a more integrated approach to optimization than trying to optimize the components in isolation. MDO problems are not restricted to the engineering domain, and applications in other areas such as public administration can also be found, where MDO can be used to optimize complex systems that involve multiple stakeholders, such as urban planning, labour market, or housing policy [18, 24]. Not many MDO benchmark problems can be found in the literature, and there are even fewer search results when constraints and multiple objectives are both considered.

Recently, a MDO bi-objective test suite [16, 17] was proposed, scalable in both the number of decision variables and disciplines. These benchmark problems have been constructed by using the multi-disciplinary feasible (MDF) architecture [6]—a type of architecture categorised as monolithic [21]. This implies that there is only a single optimization problem in the entire system, and the behaviour of each component, or discipline, in the system is modelled by the use of discipline analysis. Much of the literature in multi-objective MDO is concentrated on monolithic problems.

There are other architectures in the literature that are called distributed [21], where the optimization problem is decomposed into a set of smaller optimization problems, which produces the same solution when reassembled. The primary motivation for partitioning the problem is to allow different teams or (engineering groups), to work separately on a part of the problem that fits into their own expertise, following a more engineering-like environment. This paper builds up on [16, 17] by proposing a distributed version of the test suite based on the collaborative optimization (CO) [3] architecture. We also introduce constraints at each subproblem, resulting in more realistic MDO problems.

The survey paper by Martins and Lambe [21] proposes and uses a taxonomy for different variations of the MDO architectures (see Fig. 7). We will use the proposed taxonomy in this paper to refer to the MDO architectures.

The paper is structured as follows: Section 2 covers the key concepts and literature in the field, including an introduction to multi-objective MDO, collaborative optimization, and constrained MDO. Section 3 introduces the proposed problem and some of the relevant nomenclature. Section 4 goes through the proposed method for solving the problem. Section 5 gives details on the experimental setup, and Section 6 shows the results for case studies one and two. Section 7 provides conclusions, discussion, and future work in the area.

2 RELATED LITERATURE

2.1 Introduction to Multi-Objective MDO

MO-MDO is a growing field where some of the concepts used in multi-objective optimization are applied to problems in MDO.

Most benchmarks and existing problems in MO-MDO are single-objective. Some of the multi-objective problems are derived directly from the single-objective problems, for example the NASA MDO test suite [25], from which a multi-objective version of the Golinski speed reducer has been derived [11, 12]. An MO-MDO test suite based on the ZDT problems was proposed in [16] and [17], and is aimed at monolithic MDO problems, that is, where only the subsystem analyses are treated as distributed components. Many of the other test problems that exist in the literature have overly specific applications or rely on multi-disciplinary analysis (MDA) equations that are difficult to implement.

A large proportion of the literature or distributed multi-objective MDO centres around multi-objective concurrent subspace optimization (CSSO) [9, 13, 14, 26] and CO [10, 23, 27, 29].

Constrained MO-MDO. Constrained MO-MDO is a subset of general MO-MDO literature, and is therefore very limited. While constraints are often included as part of the written optimization problem in the relevant research literature, very little work has been dedicated to constraints in multi-objective MDO.

Some of the research articles on multi-objective CO [27] have adapted the Golinski speed reducer test problem from the MDO NASA test suite [25] (adapted in [12]) which include constraints on the design variables at the subsystem levels. The multi-objective Sellar problem [28] is adapted and used in [10] to demonstrate the authors' method for handling MO-MDO problems, with one inequality constraint in each of the two subsystems.

Aside from the above, to the knowledge of the authors, there is no literature focusing on the use of constraints in MO-MDO, and none at all in scalable MO-MDO.

2.2 Background: Collaborative Optimization

CO, proposed in 1996 [3], is a distributed MDO architecture where optimizers work at both the system and the subsystem levels to obtain separate distributed goals – i.e. one goal for the system-level problem, and another (a consistency objective) for each of the subsystem-level problems. This involves different objective functions, constraints, and design variables in each problem.

Copies of design and linking variables, denoted with a hat ˆ, are used to ensure consistency between the variables at the subsystem and system levels. Copies of the global variables are established at the subsystem levels, while copies of the local and linking variables are established at the system level.

The subsystem problems work to minimise the sum of squared errors between the copies of the design variables received from the system-level optimizer, and the design variables obtained at the subsystem level. Constraints may be in action at the subsystem level, providing some conflict between the system and subsystem-level objectives.

The aim of CO is to allow the optimizers at the subsystem level to work with some degree of independence from other teams. Information-sharing is limited, as each subsystem only has access to its own design variables and the copies of linking variables of other subsystems. This may be useful in real-world applications where data privacy is important. For example, in applications such as decision-making in government, sharing data can be an issue due to security concerns. Similarly, in industrial applications, commercially sensitive information may need to be kept within departments or disciplines to minimise risk of information leaks. Because of this, CO may be a good option for an MDO optimization architecture in such fields.

However, CO has some key difficulties. It has also been shown that single-objective CO is generally slow to converge when compared with other architectures [1, 30], especially the monolithic architectures such as multi-disciplinary feasible or all-in-one. Additionally, according to the formulation of CO, increasing the number of design variables means that more copy variables are introduced, meaning that a similar optimization problem executed in a different MDO architecture would contain fewer design variables (and therefore quicker convergence) [1].

Multi-objective CO. The literature on multi-objective CO is fairly limited. Some of the multi-objective CO strategies rely on a priori methods such as linear physical programming [20, 23] or weighted sums [29]. However, a designer may prefer an a posteriori approach that allows them to choose a solution according to trade-offs or conflicts.

The collaborative optimization strategy for multi-objective systems (COSMOS) was proposed in 2007 [27]. This is a set-based approach where child solutions are generated from parent solutions, and the best solutions go forward to make up further generations. In COSMOS, the optimization is terminated when the number of supervisor iterations has met a predetermined number; in [27], this is 20.

Also based on the CO architecture is the Pareto Genetic Algorithm Collaborative optimization (PGACO) [10]. PGACO uses a similar approach to COSMOS by adopting an internal and an external cycle (analogous to the supervisor iterations in COSMOS) and using selection, crossover and mutation operators on the solutions following the completion of one external loop. As in COSMOS, the external loop termination criteria is based on a predetermined number of loops, and once this number is reached, the optimization is terminated.

3 PROPOSED TEST SUITE

The proposed test suite builds on a previously proposed multi-objective MDO test suite by the present authors, based on a monolithic architecture [16, 17]. However, monolithic architectures assume the existence of a single optimization problem, and may not be suitable for dealing with optimization problems that have been partitioned along disciplinary lines (or engineering groups), where each discipline has its own subproblem. The proposed test suite addresses this issue by adopting a distributed architecture based on the CO approach. There are other distributed architectures in the literature [21], but we have chosen CO because the disciplinary subproblems can be independent from each other, allowing teams to control the level of detail shared between them. Another contribution of the new test suite when compared with [16, 17] is the inclusion of constraints in the problem formulation—a very common property of many engineering problems. Given that the proposed problem formulation is based on the ZDT problems [32], we named this new test suite 'CO-ZDT'.

The CO architecture has a system subproblem where it is possible to define performance criteria for the entire system. Each discipline (or subsystem) contains a subproblem with a discipline analysis, and interdependencies can exist between the discipline analysis of the different subsystems. There are three types of design variables: global, local, and linking. Global variables are denoted by $\mathbf{z} = (z_1, \ldots, z_{n_z})^T$ and are accessible to both system and subsystem subproblems. Local variables are distributed across N subsystems, and are only accessible to their subproblems. Let $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,n_{x_i}})^T$ which contains n_{x_i} local variables at the ith subsystem where $i \in \{1, \ldots, N\}$. The linking variables are the output of an analysis conducted by each discipline, with the intention of mimicking the behaviour of a particular component in the system. There is a total of n_{y_i} output linking variable at the ith subsystem, given by $\mathbf{y}_i = (y_{i,1}, \ldots, y_{i,n_{y_i}})^T$. The disciplinary subproblems are made independent of each other by using copies of the design variables. These are denoted with a hat; for example $\hat{\mathbf{x}}_i$ is the copy of the vector of local variables $\mathbf{x}_i$. The "copies" of local and linking variables are treated as design variables at the system subproblem, while the "copies" of the global variables are treated as design variables by the subproblems of the subsystems. The system subproblem is given by:

$$\begin{aligned}
\min \quad & f_1(\mathbf{z}) = z_1 \\
\min \quad & f_2(\xi(\hat{\mathbf{x}}, \hat{\mathbf{y}}), \mathbf{z}) = g(\xi(\hat{\mathbf{x}}, \hat{\mathbf{y}}), \mathbf{z}) h(\mathbf{z}, \xi(\hat{\mathbf{x}}, \hat{\mathbf{y}})) \\
\text{where} \quad & g(\xi(\hat{\mathbf{x}}, \hat{\mathbf{y}}), \mathbf{z}) = 1 + \frac{9}{N_v - 1} \left(\sum_{i=1}^{N} \sum_{j=1}^{n_{x_i}} \tilde{\hat{x}}_{i,j} + \sum_{j=2}^{n_z} z_j \right) \\
& h(\mathbf{z}, \xi(\hat{\mathbf{x}}, \hat{\mathbf{y}})) = 1 - \sqrt{\frac{f_1(\mathbf{z})}{g(\mathbf{z}, \xi(\hat{\mathbf{x}}, \hat{\mathbf{y}}))}} \\
\text{w.r.t} \quad & \hat{\mathbf{x}}, \hat{\mathbf{y}}, \mathbf{z} \\
\text{s.t.} \quad & 0 \le \hat{\mathbf{x}}, \hat{\mathbf{y}}, \mathbf{z} \le 1 \\
& J^* = \sum_{i=1}^{n_z} (z_i - \hat{z}_{1,i})^2 + \sum_{i=1}^{n_z} (z_i - \hat{z}_{2,i})^2 + \\
& \sum_{i=1}^{N} \sum_{j=1}^{n_{x_i}} (\hat{x}_{i,j} - x_{i,j})^2 + \sum_{i=1}^{N} \sum_{j=1}^{n_{y_i}} (\hat{y}_{i,j} - y_{i,j})^2 \le \varepsilon
\end{aligned} \tag{1}$$

where the linking variables have been incorporated into the subproblem via the following function:

$$\xi(\hat{\mathbf{x}}_i, \hat{\mathbf{y}}_i) = \hat{\mathbf{x}}_i + ||\hat{\mathbf{y}}_i - \mathbf{y}_i^*||_1. \tag{2}$$

In Equation 2 the decision variables are penalised by the deviation between the linking variables and their optimal values ($\mathbf{y}_i^*$). The operator $|| \bullet ||_1$ is the L^1-norm. Let the output of Equation 2 be denoted by the vector $\tilde{\hat{x}}_i = (\tilde{\hat{x}}_{i,1}, \ldots, \tilde{\hat{x}}_{i,n_{x_i}})^T$. The function J^* is a consistency constraint that quantifies the difference between the values of the decision variables (and linking variables as well) that are kept by the system and subsystems subproblems. Given that the consistency constraint can be difficult to satisfy (implying that the deviation between system and subsystems is zero), we consider that the constraint is satisfied when the deviation is below a small number ε (bigger than zero). The subproblem at the kth subsystem where $k \in \{1, \ldots, N\}$ is given by:

$$\begin{aligned}
\min J_k(\mathbf{x_k}, \mathbf{y_k}, \hat{\mathbf{z}}_\mathbf{k}) = & \sum_{i=1}^{n_z} (z_i - \hat{z}_{k,i})^2 + \sum_{i=1}^{n_{x_k}} (\hat{x}_{k,i} - x_{k,i})^2 + \\
& \sum_{i=1}^{n_{y_k}} (\hat{y}_{k,i} - y_{k,i})^2
\end{aligned} \tag{3}$$

$$\text{w.r.t } \mathbf{x_k}, \mathbf{y_k}, \hat{\mathbf{z}}_\mathbf{k}$$

$$\text{s.t. } c_k \equiv -|\alpha_k(\hat{z}_1 - \beta_k)| + 1 \le 0 \text{ and } 0 \le \mathbf{x_k}, \mathbf{y_k}, \hat{\mathbf{z}}_\mathbf{k} \le 1$$

The objective function resembles the consistency constraint found in the system subproblem (Equation 2), and its aim is to reduce the inconsistency with respect to the kth subsystem. The constraint function c_k generates infeasible regions in f_1, where α determines the width of the regions and β specifies their lateral placement. Based on our experimental results we recommend setting α to 0.02. The value of β depends on the number of subsystems and should be set in a way that produces a number of evenly spaced feasible regions across the objective space. The values of the linking variables are determined at the subsystem level and require solving the following system of equations:

$$\boldsymbol{A}\mathbf{y} = -\boldsymbol{C}\hat{\hat{\mathbf{z}}} - \boldsymbol{D}\mathbf{x}, \tag{4}$$

where $\hat{\hat{\mathbf{z}}} = (\hat{\mathbf{z}}_2, \dots \hat{\mathbf{z}}_{n_z})^T$ excludes the first shared variable used in f_1, $\mathbf{y} = (\mathbf{y}_1, \dots, \mathbf{y}_N)^T$ contains all linking variables, and all decision variables are in $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_N)^T$. The matrices $\boldsymbol{A}$, $\boldsymbol{C}$ and $\boldsymbol{D}$ specify the couplings between the subsystems; more details about these matrices can be found in [17]. To solve the system of equations in Equation 4, a multidisciplinary analysis solver from the MDO literature can be used (e.g. Gauss–Seidel or Newton-based methods [21]).

4 PROPOSED METHOD AND TERMINATION CRITERIA

The method used in this paper builds upon the single-objective approach for CO, but is different from some of the existing multi-objective CO strategies such as COSMOS in that each member of the population is optimized separately.

(1) The variables ${\mathbf{x_i}}^{(0)}$ and $\hat{\mathbf{z}}^{(0)}$ are initialised using a Latin hypercube design of experiments. This is of size $nPop$, the size of the population. The matrices B_i, C_i and D_i, used in the subsystem analyses, are also initialised and row-normalised.
(2) The subsystem analyses are run once, using the values of ${\mathbf{x_i}}^{(0)}$ and $\hat{\mathbf{z}}^{(0)}$ established in Step (1). These return the values of the initial linking variables ${\mathbf{y_i}}^{(0)}$.
(3) The initial global, local and linking variables are sent to the consistency constraint J^* in the system-level optimization problem shown in 3. The system-level optimizer then uses its own design variables $\hat{\mathbf{x}}_\mathbf{i}$, $\hat{\mathbf{y}}_\mathbf{i}$ and $\mathbf{z}$ to find a set of solutions that satisfy the consistency constraint and are close to optimal in objective functions f_1 and f_2. At this stage in the optimization, ε is a large number, for example 10, to guarantee a set of solutions are found.
(4) The main optimization loop takes place:
    (a) The subsystems receive the variables $\hat{\mathbf{x}}_\mathbf{i}$, $\hat{\mathbf{y}}_\mathbf{i}$ and $\mathbf{z}$, obtained by the initial optimization in Step (3). They perform a single-objective optimization using the variables $\mathbf{x_i}$ and $\hat{\mathbf{z}}$ with the aim of minimising the consistency objective J_i and satisfying the subsystem constraints c_i. The linking variables $\mathbf{y_i}$ are obtained by running the subsystem analyses shown in Equation 3, and are used in the consistency objectives.
    (b) When optimization at the subsystem level has been terminated, the system-level problem receives the variables $\mathbf{x_i}$, $\mathbf{y_i}$ and $\hat{\mathbf{z}}$ from the subsystems. The system then runs the optimization again, similarly to that in Step (3), but ε is set to a much lower value – in this paper, between 10^{-1} and 10^{-7}. Subject to the consistency constraints, the system solves the problem with variables $\hat{\mathbf{x}}_\mathbf{i}$, $\hat{\mathbf{y}}_\mathbf{i}$ and $\mathbf{z}$.
    (c) The termination criteria, described in Section 4 are assessed. If the termination criteria are satisfied, or all the permitted supervisor iterations have been expended, the optimization terminates. If the termination criteria are not satisfied, and the number of completed supervisor iterations is lower than the permitted number, the optimization loop repeats from Step (4)(a).
(5) When the optimization process has been terminated, the results are stored for further analysis.

The above method is also described in the extended design structure matrix (XDSM) in Figure 1. See [19] for more information on XDSM diagrams.

There are three termination criteria:

(1) The constraint on the shape variable expressed in 3 is satisfied.
(2) The constraints on the linking variables expressed in 3 are satisfied to within 10^{-15}.
(3) The consistency constraint in the system-level problem is satisfied by all solutions (i.e. the maximum J^* must be equal to or less than ε).

There are a maximum of 10 permitted supervisor (system-subsystem) iterations for the sake of reducing experimental time while allowing the optimizers a suitable number of function evaluations to find a permissible result. When all of the above termination criteria are met, the optimization process is terminated. If all of the above criteria are not met within the 10 supervisor iterations, the optimization is terminated and the results are stored.

The code used to conduct the method described in this section are available in the project's GitHub repository[1].

5 EXPERIMENTAL SETUP

The following experiments were undertaken in Python, making use of the PyMoo package [2] for the system-level optimizer and SciPy [31] for the subsystem-level optimizers. The system-level optimizer is NSGA-II [7]. For each run of NSGA-II, a population of 50 is used with a number of generations that is varied between 2000 and 10,000 in the experiments. The subsystem-level optimizers are both SLSQP with a maximum of 200 iterations and a function tolerance of 10^{-5}. This is to prevent excessive optimization time. The optimal linking variable values y^* from Equation 3 is a vector of zeros of length n_{y_k} for each subsystem k. Each optimization is repeated 11 times for statistical significance. The hypervolumes of each of the repetitions at the final supervisor iteration are measured, and the median repetition is used for analysis.

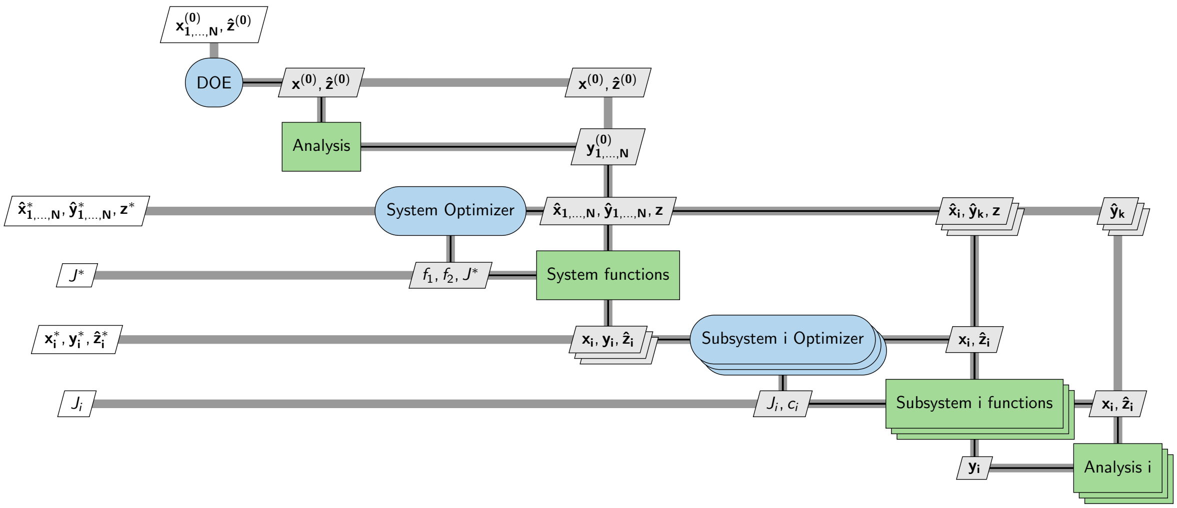
The constraints are spread equally between 0 and 1 in the z_1 variable. For example, for a 2-discipline system, there would be 2 equally spaced constraints, and in a 5-discipline system, there would be 5 equally spaced constraints.

The first experiment involves varying ε between 10^{-1} and 10^{-7}. Two subsystem problems will be used, and the number of generations will not be allowed to vary throughout. Each of the ZDT problems except 3 and 5 are used.

The second experiment looks at the evolution of the design variables and the consistency constraint at the system level for CO-ZDT1, 2, 4 and 6 for two, three, five and ten subsystems. ε is set to 10^{-7}.

[1] https://github.com/vj2Sheffield/CO_ZDT_benchmarks

Figure 1: The XDSM diagram for the multi-objective collaborative optimization strategy used in this paper.



The third experiment investigates the outcomes of the optimization when 10,000 generations are used in a problem with 2 subsystems. Once again, ε is varied between 10^{-1} and 10^{-7}.

The fourth study extends the approach in the first experiment to three subsystems. 2000 generations are used and ε is varied between 10^{-1} and 10^{-7}.

6 RESULTS

6.1 Convergence in a Single Supervisor Run of Multi-objective CO

The speed of convergence is of particular interest in collaborative optimization. In case study one, we set the number of allotted generations to 15,000 for one design point (15,000 function evaluations) to allow the optimizer to reach an acceptable value. This run takes place after the initial system-level optimization and a single subsystem-level optimization – in other words, this is after one supervisor iteration. The experiments were also undertaken with two, three, five and ten subsystems to demonstrate the effects of increasing the number of subsystems on convergence speed.

Figure 2 shows the value of the consistency constraint J^* over the 15,000 generations for CO-ZDT1, 2, 4 and 6 and two, three, five and ten subsystems. When the consistency constraint value crosses the red dashed line, it indicates that the constraint is satisfied, and in a run of the multiobjective CO method we outlined in Section 4, termination criterion (1) would be satisfied.

It can be seen that the only cases where the consistency constraint is satisfied is in CO-ZDT1, 2 and 6 for two subsystems. CO-ZDT1 and 2 show similar evolutions of constraint values, with each of the different subsystem numbers reaching similar final values. The ten-subsystem problems show the slowest reduction in the consistency constraint, settling at just below 1. The CO-ZDT4 and 6 problems show similar results, with 10 subsystems demonstrating the most shallow curve. Additionally in both the CO-ZDT4 and 6 problems, the final value of the consistency constraint is lowest in two subsystems, followed by three, five and ten subsystems.

Figure 2: The values of the consistency constraints J^* with a budget of 15,000 generations and for 2, 3, 5 and 10 subsystems. The red dashed lines indicate the placement of the constraint $\varepsilon = 10^{-7}$.

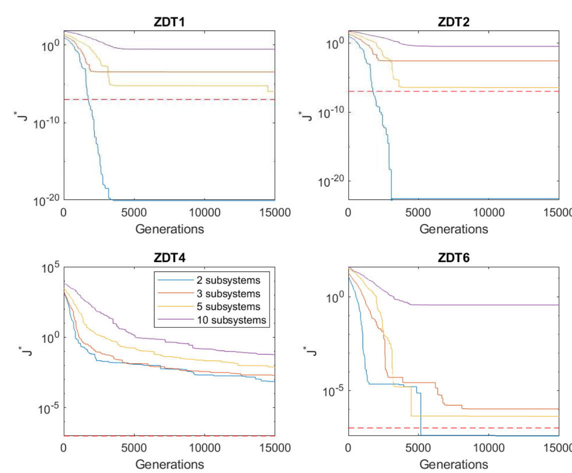


Figure 3 shows the convergence of the global (excluding z_1), copy local and linking variables for CO-ZDT1, 2, 4 and 6 problems with two subsystems. The variables shown in CO-ZDT4 are absolutes to demonstrate the movement of the variables at very small values. Like in Figure 2, ZDT1 and 2 demonstrate similar convergence patterns, reaching their final values before the 4000th generation. In CO-ZDT4, the variables decrease in the first 5000 generations, then stagnate between 10^{-4} and 10^{-2} until the 15,000th generation. In CO-ZDT6, the variables drop in the first 2000 generations, stall at approximately 10^{-13}, and then decrease again just before the 10,000th generation.

Figure 3: The convergence of the global, copy local and copy linking variables in one run of the optimizer with $\varepsilon = 10^{-7}$, excluding z_1, and a budget of 15,000 generations and two subsystems. The variables in CO-ZDT4 are absolutes to demonstrate the convergence movements at smaller values.

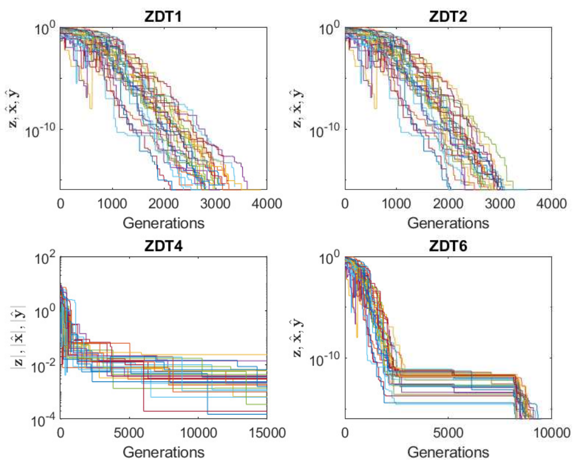


Figure 4: The convergence of the global, copy local and copy linking variables in one run of the optimizer with $\varepsilon = 10^{-7}$, excluding z_1, and a budget of 15,000 generations and ten subsystems.

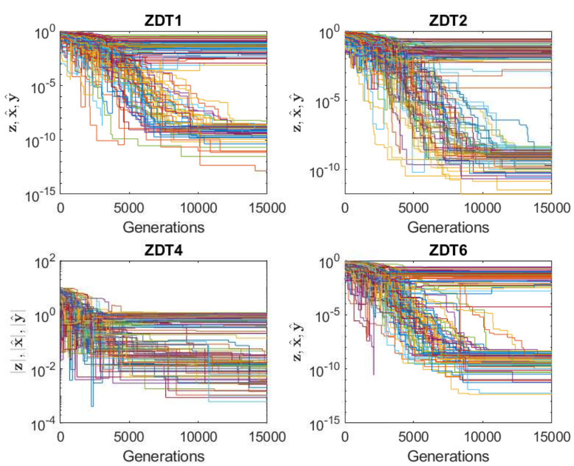


Figure 4 shows the convergence of all variables for the problem where there are ten subsystems. In these problems, there are 110 variables in total: 10 global variables, 50 local copy variables, and 50 linking copy variables. It can be seen that some of the variables decrease substantially, especially in CO-ZDT1, 2 and 6. However, some of them also remain between 10^{-1} and 10^{-5} and stagnate.

6.2 System Runs with 2000 Generations

In case study two, CO-ZDT1, 2, 4 and 6 were studied using the multi-objective CO method described in Section 4 using a set number of generations, population, and maximum supervisor iterations.

Figure 5 shows the mean number of supervisor iterations used before termination for each of the CO-ZDT problems and each ε value. It can be seen that, for all CO-ZDT except CO-ZDT4, the number of supervisor iterations at termination increase steadily until $\varepsilon = 10^{-4}$ and then jumps for smaller values in a shape resembling a sigmoid curve. In CO-ZDT4, the mean number of iterations starts at just under 6, then jumps to 10 for the remaining ε values.

Figure 5: The mean number of supervisor iterations completed at termination.

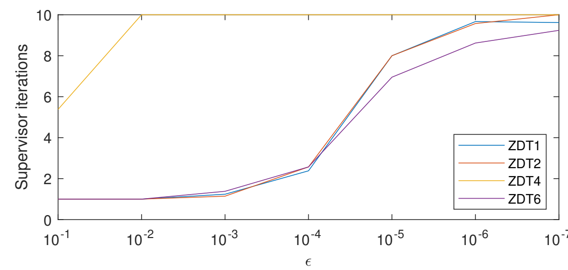


Figure 6 shows the nondominated solutions at termination for each of the CO-ZDT problems for $\varepsilon = 10^{-1}$ and 10^{-7}. Firstly, the solutions for CO-ZDT1 and 2 are easily recognisable as they adhere closely to the Pareto front, while CO-ZDT4 and 6 do not, with CO-ZDT4 displaying large f_2 values and CO-ZDT6 having solutions limited to the values of f_1 larger than 0.9.

In terms of constraint violation, the problems where ε is very low shows better constraint adherence than in those where ε is larger. Additionally, diversity along the nondominated solutions is greater in problems where ε is small than in those where it is large. Interestingly, a smaller ε appears to lead to f_2 values in CO-ZDT4 that are further away from the Pareto front than larger values - $\varepsilon = 10^{-1}$ tops out at just under 40, while $\varepsilon = 10^{-7}$ reaches over 150. The opposite effect is found in CO-ZDT6, where f_2 becomes smaller with a smaller ε; the largest f_2 solution in $\varepsilon = 10^{-1}$ is over 4, while the largest in $\varepsilon = 10^{-7}$ is just over 0.8.

6.3 System Runs with 10,000 Generations

In case study three, the number of generations was increased to 10,000, and the number of subsystems in the problem was maintained at two.

Figure 7 shows the nondominated solutions for CO-ZDT1 and 4 in a two-subsystem problem, $\varepsilon = 10^{-1}$ and 10^{-7} and 10,000 generations. It can be seen that the solutions at $\varepsilon = 10^{-1}$ are much less evenly spread across the objective space when compared with $\varepsilon = 10^{-7}$. In CO-ZDT4 at $\varepsilon = 10^{-1}$, the shape of the Pareto front is not recovered. However, at $\varepsilon = 10^{-7}$ for CO-ZDT4, the shape is recovered, and the nondominated points are much closer to the Pareto front in f_2.

Figure 6: Nondominated solutions at termination for $\varepsilon = 10^{-1}$ and 10^{-7} for CO-ZDT1, 2, 4 and 6, in a two-subsystem problem.

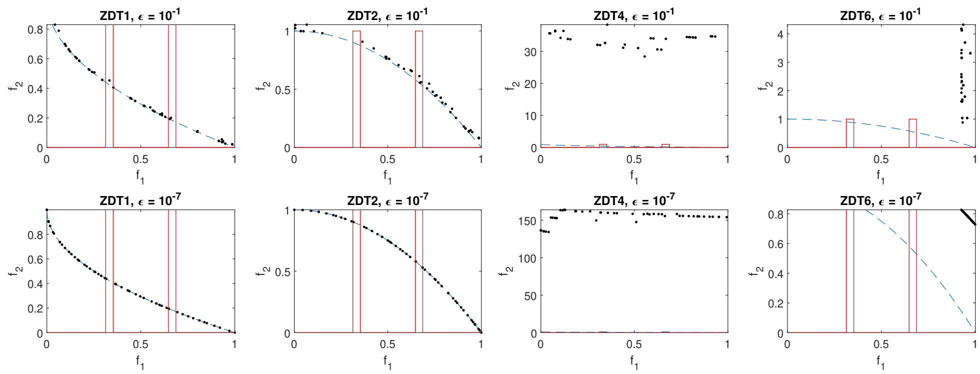
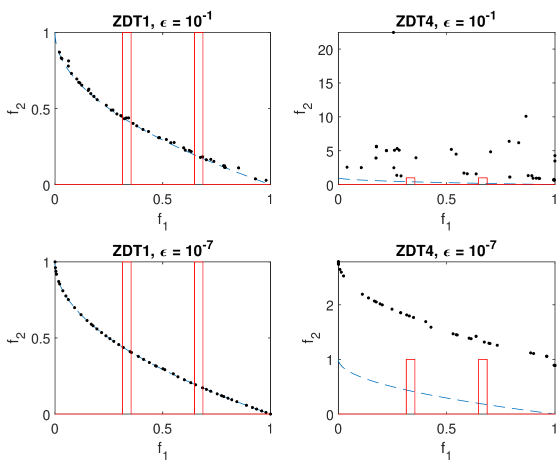


Figure 7: Nondominated solutions at termination for $\varepsilon = 10^{-1}$ and 10^{-7} for CO-ZDT1 and 4 in a two-subsystem problem with 10,000 generations at the system level.



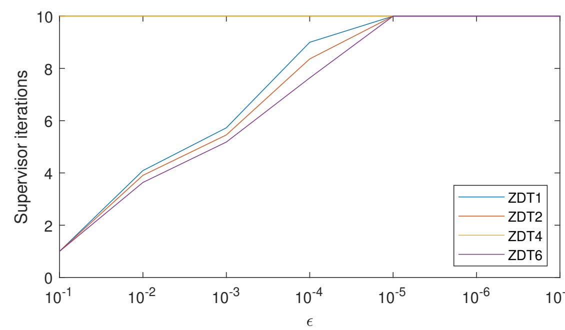
6.4 System Runs with 2000 Generations and 3 Subsystems

In case study four, the number of disciplines was increased to three and the number of generations was set at 2000, as in experiment one.

Figure 8 shows the mean number of supervisor iterations at termination for each of the CO-ZDT problems studied in this paper. CO-ZDT1, 2 and 6 trend upwards from $\varepsilon = 10^{-1}$ to 10^{-5}, and then reach 10 supervisor iterations. CO-ZDT4 begins with an average of 10 supervisor iterations, where it remains for all of the ε values. Like in case study one, the results indicate that for all the CO-ZDT problems studied in this paper, the number of supervisor iterations needed for termination increases with smaller values of ε.

Figure 9 shows the nondominated solutions for the three-discipline problem using the same format as in Figure 6. The constraints on z_1 are shown by the red lines while the black dots show the nondominated solutions. For CO-ZDT1, it can be seen that the nondominated

Figure 8: The mean number of supervisor iterations completed at termination for three subsystems.

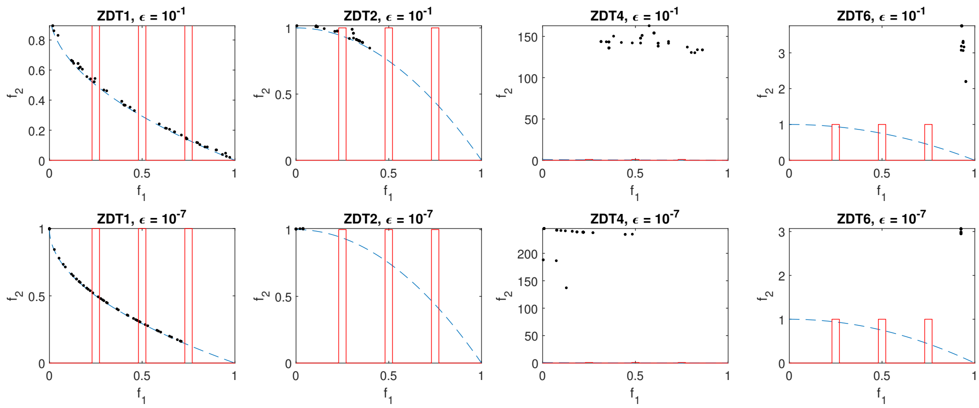


solutions for both $\varepsilon = 10^{-1}$ and 10^{-7} are close to the Pareto front, but in $\varepsilon = 10^{-1}$ there are greater constraint violations. Additionally, in $\varepsilon = 10^{-7}$, the nondominated solutions fail to cover the whole range of possible z_1, meaning the solutions do not extend past approximately 0.73. The spread gets worse with CO-ZDT2, especially at the smaller value of $\varepsilon = 10^{-7}$, where the solutions only extend to 0.04, and 0.4 for 10^{-1}. In CO-ZDT4, the solutions are more concentrated in the middle of f_1 for $\varepsilon = 10^{-1}$, but is less resemblant of the shape of the Pareto front than in 10^{-7}. Like in case study 1, the nondominated solutions are further away from the Pareto front in $\varepsilon = 10^{-7}$. Again, similarly to in case study 1, the solutions for CO-ZDT6 are limited to the larger values of f_1 in both $\varepsilon = 10^{-1}$ and 10^{-7}.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a set of four scalable benchmark problems for multi-objective collaborative optimization based off the ZDT test suite. We have investigated the results of running the optimization with the NSGA-II optimizer at the system level and the SLSQP optimizer at the subsystem level. We found that the problems tend to be slow to converge, and benefit from the use of a larger budget of function evaluations at the system level. We also found that the number of subsystems also tends to decrease the

Figure 9: Nondominated solutions at termination for $\epsilon = 10^{-1}$ and 10^{-7} for CO-ZDT1, 2, 4 and 6, in a three-subsystem problem.



speed of convergence, and that, in CO-ZDT4, the optimizers tended to find a solution that satisfied the consistency constraints much more slowly than in other CO-ZDT problems.

Establishing a set of constrained benchmarks in the area of MO-MDO opens up a wide range of research approaches for specialists in other fields, such as multi-criterion optimization, multidisciplinary design and classical optimization.

7.1 On the Speed of Convergence in CO

It is well-established that CO takes a long time to converge on a solution. Tedford and Martins show that, out of several monolithic and distributed architectures in single-objective problems, CO often demonstrates the slowest convergence [30]. Alexandrov and Lewis note the poor efficiency in CO due to the high autonomy afforded to the subsystems [1], especially in problems with large numbers of design variables (often caused by the structure of CO itself). Cormier et al. state that the slow convergence associated with CO had a detrimental effect on their design of a reusable launch vehicle [5].

Given the above, and the results in Section 6, we find that our results are in line with the existing literature on CO. This architecture produces more complex problems than 'vanilla' ZDT and therefore requires more computational resources.

7.2 Future Work

Multi-objective MDO is an emerging field and further work in this area could take many different directions. Firstly, and most importantly, it should be stated that problems closely related to the ZDT test suite do not have attributes that are desirable for a bi-objective BBOB problem—lacking scalability in objectives, relying on the leading variable to determine the shape of the nondominated solution front, no nonseperability in objective functions, and so on. The benchmark test set proposed in this paper inherits these problems. Therefore, future work should be focused on developing problems that are more realistic to real-world problems, and contain attributes such as more than two objectives, complex variable transformations, deception, degeneracy and so on.

Secondly, the number of supervisor iterations could be increased, with an analysis on how long it takes the optimization to reach a satisfactory solution.

Indicator-based constraints could be introduced as a requirement for termination, in addition to the termination criteria set out in Section 4. An example of this would be adding a termination criterion stating that a certain hypervolume must be achieved before the optimization process is allowed to terminate.

Additionally, future work should allow for some variation of the optimization problems by linking the subsystems together in different ways. In this work, each subsystem has contributed linking variables to and received linking variables from only one other discipline. In further work, subsystems should receive and contribute linking variables to and from multiple subsystems. This complicates the problem, and is more realistic to real-world problems.

ACKNOWLEDGMENTS

VJ was supported by the UK Engineering and Physical Sciences Research Council. JAD, VK and RCP were supported by SIPHER (MR/S037578/1), a UK Prevention Research Partnership funded by the UK Research and Innovation Councils, the Department of Health and Social Care (England) and the UK devolved administrations, and leading health research charities https://ukprp.org/.

REFERENCES

[1] Natalia Alexandrov and Robert Lewis. 2000. Comparative Properties Of Collaborative Optimization And Other Approaches To Mdo. *ASMO UK/ISSMO conference on engineering design optimization* (03 2000).
[2] J. Blank and K. Deb. 2020. pymoo: Multi-Objective Optimization in Python. *IEEE Access* 8 (2020), 89497–89509.
[3] Robert D. Braun, Peter Gage, I. M. Kroo, and I Sobiesiki. 1996. Implementation and Performance Issues in Collaborative Optimization *(6th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization)*. 295–305. https://doi.org/10.2514/6.1996-4017
[4] Dimo Brockhoff, Anne Auger, Nikolaus Hansen, and Tea Tušar. 2022. Using Well-Understood Single-Objective Functions in Multiobjective Black-Box Optimization Test Suites. *Evolutionary Computation* 30, 2 (06 2022), 165–193. https://doi.org/10.1162/evco_a_00298 arXiv:https://direct.mit.edu/evco/article-pdf/30/2/165/2025995/evco_a_00298.pdf
[5] Timothy Cormier, Andrew Scott, Laura Ledsinger, David McCormick, David Way, and John Olds. 2012. Comparison of collaborative optimization to conventional design techniques for a conceptual RLV. In *8th Symposium on Multidisciplinary Analysis and Optimization*. https://doi.org/10.2514/6.2000-4885

arXiv:https://arc.aiaa.org/doi/pdf/10.2514/6.2000-4885

[6] Evin J. Cramer, J. E. Dennis, Jr., Paul D. Frank, Robert Michael Lewis, and Gregory R. Shubin. 1994. Problem Formulation for Multidisciplinary Optimization. *SIAM Journal on Optimization* 4, 4 (1994), 754–776. https://doi.org/10.1137/0804044

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (2002), 182–197. https://doi.org/10.1109/4235.996017

[8] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. 2005. *Scalable Test Problems for Evolutionary Multiobjective Optimization.* Springer London, London, 105–145. https://doi.org/10.1007/1-84628-137-7_6

[9] Joao Duro, Yiming Yan, Robin Purshouse, and Peter Fleming. 2018. Collaborative Multi-Objective Optimization for Distributed Design of Complex Products. *GECCO '18: Proceedings of the Genetic and Evolutionary Computation Conference*, 625–632. https://doi.org/10.1145/3205455.3205579

[10] Yang Fan and Bouchlaghem Dino. 2010. Genetic Algorithm-Based Multiobjective Optimization for Building Design. *Architectural Engineering and Design Management* 6, 1 (2010), 68–82. https://doi.org/10.3763/aedm.2008.0077 arXiv:https://doi.org/10.3763/aedm.2008.0077

[11] Michael Farnsworth, Ashutosh Tiwari, Meiling Zhu, and Elhadj Benkhelifa. 2018. *A Multi-objective and Multidisciplinary Optimisation Algorithm for Microelectromechanical Systems.* Springer International Publishing, Cham, 205–238. https://doi.org/10.1007/978-3-319-64063-1_9

[12] S Gunawan, S Azarm, J Wu, and A Boyars. 2003. Quality-Assisted Multi-Objective Multidisciplinary Genetic Algorithms. *AIAA journal* 41, 9 (2003), 1752–1762.

[13] Chen-Hung Huang. 2003. *Development of Multi-Objective Concurrent Sub-space Optimization and Visualization Methods for Multidisciplinary Design.* Ph. D. Dissertation. Buffalo, NY: State University of New York at Buffalo.

[14] Chen-Hung Huang and Christina Bloebaum. 2004. Incorporation of Preferences in Multi-Objective Concurrent Subspace Optimization for Multidisciplinary Design. In *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference.* https://doi.org/10.2514/6.2004-4548 arXiv:https://arc.aiaa.org/doi/pdf/10.2514/6.2004-4548

[15] S. Huband, P. Hingston, L. Barone, and L. While. 2006. A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation* 10, 5 (2006), 477–506. https://doi.org/10.1109/TEVC.2005.861417

[16] V. Johnson, J. A. Duro, V. Kadirkamanathan, and R. C. Purhouse. 2022. Toward scalable benchmark problems for multi-objective multidisciplinary optimization. In *Proceedings of the 2022 IEEE Symposium Series On Computational Intelligence (IEEE SSCI).* 133–140.

[17] V. Johnson, J. A. Duro, V. Kadirkamanathan, and R. C. Purhouse. 2023. A scalable test suite for bi-objective multidisciplinary optimisation. In *Proceedings of the 2023 International Conference Series on Evolutionary Multi-Criterion Optimization (EMO).* 319–332.

[18] Kathrin Klamroth, Sanaz Mostaghim, Boris Naujoks, Silvia Poles, Robin Purshouse, Günter Rudolph, Stefan Ruzika, Serpil Sayın, Margaret M. Wiecek, and Xin Yao. 2017. Multiobjective optimization for interwoven systems. *Journal of Multi-Criteria Decision Analysis* 24, 1–2 (2017), 71–81. https://doi.org/10.1002/mcda.1598

[19] Andrew B. Lambe and Joaquim R. R. A. Martins. 2012. Extensions to the design structure matrix for the description of multidisciplinary design, analysis, and optimization processes. *Structural and Multidisciplinary Optimization* 46, 2 (01 Aug 2012), 273–284. https://doi.org/10.1007/s00158-012-0763-y

[20] Haiyan Li, Mingxu Ma, and Wenlei Zhang. 2016. Multi-objective collaborative optimization using linear physical programming with dynamic weight. *Journal of Mechanical Science and Technology* 30, 2 (01 Feb 2016), 763–770. https://doi.org/10.1007/s12206-016-0131-8

[21] Joaquim R. R. A. Martins and Andrew B. Lambe. 2013. Multidisciplinary Design Optimization: A Survey of Architectures. *AIAA Journal* 51, 9 (2013), 2049–2075. https://doi.org/10.2514/1.J051895

[22] Charles D. McAllister and Timothy W. Simpson. 2003. Multidisciplinary Robust Design Optimization of an Internal Combustion Engine . *Journal of Mechanical Design* 125, 1 (2003), 124–130. https://doi.org/10.1115/1.1543978

[23] C. D. McAllister, T. W. Simpson, K. Hacker, K. Lewis, and A. Messac. 2005. Integrating linear physical programming within collaborative optimization for multiobjective multidisciplinary design optimization. *Structural and Multidisciplinary Optimization* 29, 3 (01 Mar 2005), 178–189. https://doi.org/10.1007/s00158-004-0481-1

[24] Petra Meier, Robin Purshouse, Marion Bain, Clare Bambra, Richard Bentall, Mark Birkin, John Brazier, Alan Brennan, Mark Bryan, Julian Cox, et al. 2019. The SIPHER Consortium: Introducing the new UK hub for systems science in public health and health economic research. *Wellcome Open Research* 4, 174 (2019). https://doi.org/10.12688/wellcomeopenres.15534.1

[25] Sharon Padula, Natalia Alex, Lawrence Green, and Natalia Alexandrov. 1996. MDO Test Suite At Nasa Langley Research Center. (Sep 1996). https://doi.org/10.2514/6.1996-4028

[26] Sumeet Parashar and Christina Bloebaum. 2006. Multi-Objective Genetic Algorithm Concurrent Subspace Optimization (MOGACSSO) for Multidisciplinary Design. *Collection of Technical Papers - AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference* 8. https://doi.org/10.2514/6.2006-2047

[27] Sébastien Rabeau, Philippe Dépincé, and Fouad Bennis. 2007. Collaborative optimization of complex systems: a multidisciplinary approach. *International Journal on Interactive Design and Manufacturing (IJIDeM)* 1, 4 (01 Nov 2007), 209–218. https://doi.org/10.1007/s12008-007-0025-1

[28] R. Sellar, S. Batill, and J. Renaud. 1996. Response surface based, concurrent subspace optimization for multidisciplinary system design. In *34th Aerospace Sciences Meeting and Exhibit.* 96–0714. https://doi.org/10.2514/6.1996-714

[29] Ravindra V. Tappeta and John E. Renaud. 1997. Multiobjective Collaborative Optimization *(International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Vol. Volume 2: 23rd Design Automation Conference).* 403–411. https://doi.org/10.1115/DETC97/DAC-3772 arXiv:https://asmedigitalcollection.asme.org/IDETC-CIE/proceedings-pdf/DETC97/80449/V002T29A017/6640325/v002t29a017-detc97-dac-3772.pdf V002T29A017.

[30] Nathan P. Tedford and Joaquim R. R. A. Martins. 2010. Benchmarking multidisciplinary design optimization algorithms. *Optimization and Engineering* 11 (2010), 159–183. https://doi.org/10.1007/s11081-009-9082-6

[31] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17 (2020), 261–272. https://doi.org/10.1038/s41592-019-0686-2

[32] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. 2000. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation* 8, 2 (2000), 173–195. https://doi.org/10.1162/106365600568202