Robotica: (2019) volume 37, pp. 1595–1640. © Cambridge University Press 2019
doi:10.1017/S026357471900016X

CAM-ADX: A New Genetic Algorithm with Increased Intensification and Diversification for Design Optimization Problems with Real Variables

Edson Koiti Kudo Yasojima†*, Roberto Célio Limão de Oliveira†, Otávio Noura Teixeira‡ and Rodrigo Lisbôa Pereira†

† *Faculty of Computer Engineering, Federal University of Pará, Rua Augusto Corrêa 01, Caixa Postal 479, Guamá, Belém, Pará CEP:66075-100, Brazil*
E-mails: limao@ufpa.br; rod.lisboa@gmail.com
‡ *Faculty of Computer Engineering, Federal University of Pará, Campus Universitário de Tucuruí, Rua Itaipu, 36 - Vila Permanente, Tucuruí, Pará CEP:68464-000, Brazil*
E-mail: onoura@gmail.com

(Accepted January 19, 2019. First published online: March 1, 2019)

SUMMARY
This paper presents a modified genetic algorithm (GA) using a new crossover operator (ADX) and a novel statistic correlation mutation algorithm (CAM). Both ADX and CAM work with population information to improve existing individuals of the GA and increase the exploration potential via the correlation mutation. Solution-based methods offer better local improvement of already known solutions while lacking at exploring the whole search space; in contrast, evolutionary algorithms provide better global search in exchange of exploitation power. Hybrid methods are widely used for constrained optimization problems due to increased global and local search capabilities. The modified GA improves results of constrained problems by balancing the exploitation and exploration potential of the algorithm. The conducted tests present average performance for various CEC'2015 benchmark problems, while offering better reliability and superior results on path planning problem for redundant manipulator and most of the constrained engineering design problems tested compared with current works in the literature and classic optimization algorithms.

KEYWORDS: Genetic algorithm; Design optimization; Real variables; Engineering design.

1. Introduction

Problem optimization is the search for optimal parameter values for functions. These functions have certain numbers of parameterized variables to maximize or minimize the results according to the objective of the problem. In most real-coded problems, it is infeasible to use brute-force algorithms or similar techniques to search for optimal parameters; the implementation of a heuristic or refined stochastic process can help to achieve satisfactory results.[1] In evolutionary computing, there are various techniques and algorithms for problem optimization inspired by nature. Among the most famous heuristic methods are genetic algorithm (GA), particle swarm optimization (PSO), artificial bee colony algorithm (ABC), and harmonic balance (HB).[2] In addition, there

* Corresponding author. E-mail: koitiyasojima@gmail.com

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

are hybrid heuristic models that use combinations of different evolutionary techniques with the aim of increasing optimization power.[3]

This kind of functions are well known in the literature and tested in mathematical and metaheuristics optimization. Mathematical optimization often uses techniques such as dynamic programming, linear/non-linear programming, simulated annealing, etc. These methods area solution-based and use gradient information to improve already known solutions near their starting points. While these methods have a better exploitation potential, they lack on exploring the whole search space, thus losing some exploration potential compared with metaheuristics. Although pure evolutionary approaches offer better exploration on search spaces, they may not obtain satisfactory results in consequence of bad local optimization compared with solution-based algorithms.[4–6]

A lot of work using evolutionary computation have been put toward constrained optimization, specially combined with other solution-based techniques. These hybridized algorithms have balanced exploration and exploitation potential, providing result improvement for many well-known optimization problems. Garg[4] shows a hybrid technique using PSO and GA for solving constrained optimization problems. Patwal et al.[7] proposed a novel modified PSO using mutation strategies. Rani et al.[8] implemented an optimistic and pessimistic viewpoint in a fuzzy environment for numerical problem optimization; also Garg[9,10] proposed an evolutionary algorithm based on biogeography for reliability redundancy allocation problems, and the application of PSO in a fuzzy environment for multi-objective reliability redundancy problem optimization. In ref. [11], Garg created a novel hybridized technique using lambda-tau methodology and PSO, also analyzing the behavior and sensitivity of industrial system. All the works cited have one or more evolutionary programming methods implemented to optimize functions; they emphasize the importance of the exploration and local exploitation of optimization algorithms. By mixing global search and local optimization, superior numerical results are achieved for the problems tested, also increasing reliability and robustness of stochastic/evolutionary algorithms in real-life scenarios.

Exploration and exploitation are broad concepts in various areas of knowledge. Exploration is the capacity to explore or seek out objectives in unknown areas, while exploitation is the capacity to improve or optimize areas previously explored. In nature, the characterization of exploration/diversification is the search for food in unknown locations, while exploitation/intensification is analogous to returning to previously discovered locations to obtain more food. Evolutionary algorithms stand out for their power to identify solutions of two different types: global search (exploration) and local search (exploitation). Global search is characterized as the algorithm's capacity for finding potentially better (new) results within an unknown search space of the problem, while local search involves the capacity to improve on solutions to the problem that have previously been identified.[12] In terms of the power of exploration and exploitation of evolutionary algorithms, Crepinsek et al.[13] found/stated that the greater the power of exploration/exploitation, the better the results obtained within the algorithm.

The efficiency of GA depends on various factors and parameters, requiring adjustments according to the optimization problem. One of the factors that directly affects GA optimization is the diversity of the population. A population with very low diversity has high probability of converging very rapidly and stagnating at an unsatisfactory result, indicating that the solution became stuck at a local optimum in the search space.[14] For a GA to be effective, an appropriate level of exploration and exploitation is required. A low level of exploration implies that the algorithm has difficulty in exploring the search space in the direction of the best individual, while a low level of exploitation may prevent the improvement of the current best individual.[13,15,44]

The crossover operation of GAs is one of the most important phases in the algorithm; through the generation of new individuals based on already existing solutions (parents), the search power for an optimal solution is more effective. Among the most widespread types of crossover used with real-coded GAs are arithmetic crossover (blend crossover – BLX/ARX)[18] and binary crossover for real numbers (simulated binary crossover – SBX).[19,20] Many of these crossovers have difficulties in problems with real-coded values with constrained search spaces, since they may become stuck in local optima or end up generating individuals that are infeasible for the solution. In addition, the genes of the new individuals created are generally limited to values between the parent genes, making it difficult to identify other areas in the search space (exploration) when the population is highly concentrated in a specific area.[21–24]

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

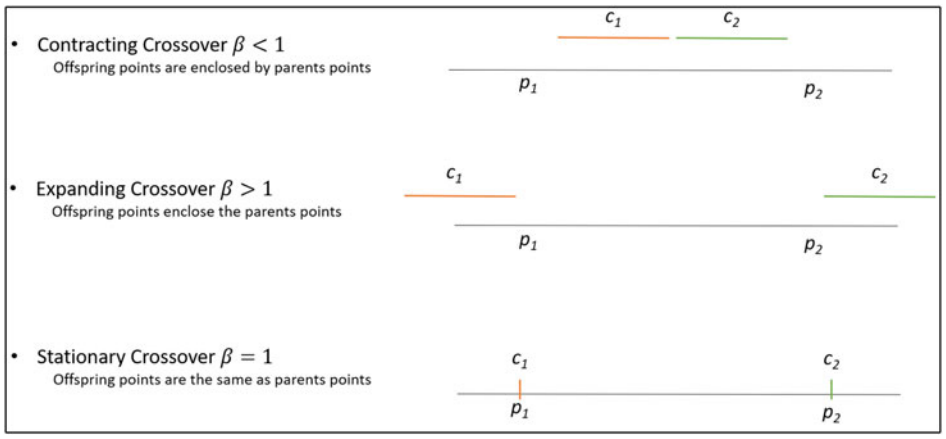


Fig. 1. Crossover types according to the spread factor.

Based on the above, this study proposes a new crossover method and correlation adjustment method to replace the mutation operator; the main objective of the proposed algorithm is to improve exploitation and exploration potential of real-coded GAs. Using ADX and CAM produces more feasible individuals than other crossover methods, thus increasing the diversity of the population and achieving better fitness values. The results using classic engineering problems and benchmark functions show superior performance and convergence for most of the optimizations. When analyzing the population diversity, the proposed method shows better performance with ADX+CAM versus compared works.

The remainder of this paper is organized as follows. Sections 2 and 3 present detailed operations of the new adjusted crossover (ADX) and the correlation adjustment mutation (CAM). Section 4 presents the computational cost analysis for the proposed methods. Section 5 compares the results obtained for the CEC'2015 problems with the top five algorithms of the competition. Section 6 applies the proposed GA to a path planning problem for a redundant manipulator. In Sections 7 and 2, we analyze the results obtained for constrained benchmark engineering problems by comparing the proposed method with other classic crossover operators and classic traditional optimization algorithms (particle swarm optimization and simulated annealing). Section 9 compares the engineering benchmark problems results with other works in the literature. The paper is concluded at Section 10.

2. Adjusted Crossover

In recent years, there has been an increase in research on crossover operators for real-coded GAs. Crossover operators are essential for GAs, and current crossover studies using real coding appear very promising. A comparative analysis between crossover operators using binary and real coding proved to be better than binary coding; however, there is still a need for operators with better levels of exploitation in the search space of the problem.[20,25–27]

Adjusted crossover (ADX) is the proposed crossover operator for real-coded GAs, using two individuals chosen by the selection method. It crosses each gene, subtracting gene values and multiplying by a certain weight, adding to or subtracting from the original parent gene. It provides increased exploitation potential for the GA, improving already known solutions.

The main objective of this crossover is to create offspring that are not too different from the parent genes, avoiding infeasibility by breaking restrictions and increasing the exploitation potential. According to a spread factor (β) denoted by (Eq. (1)):

$$\beta = \left| \frac{c_1 + c_2}{p_1 + p_2} \right| \tag{1}$$

where c is the children genes and p is the parent genes; this factor shows whether the offspring generated enclose the parent genes, the parent genes enclose the offspring gens, or if both genes are stationary Fig. 1. The proposed crossover creates offspring with contracting or expanding types, while SBX creates stationary types; and ARX, contracting types only.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

```
1. Create an ascending list (worst to best) of individuals selected for crossover to
the next generation
2. Create an integer array whose size is equal to the number of genes possessed by
each individual
3. Determine the direction for optimizing each gene and insert the corresponding
value into the array (1 for increasing value, -1 for decreasing value).
4. Choose two of the individuals to crossover, for each gene:
            a. adjustValue = parentGene1 - parentGene2
            b. Check the direction on the previously created array, is ascending:
                        i. newGene = parentGene + adjustValue * crossoverWeight
            c. If descending:
                        i. newGene = parentGene - adjustValue * crossoverWeight
            d. Set newGene to the new offspring
5. Repeat from step 4
```

Fig. 2. ADX algorithm.

The ADX operation is described next (Eq. (2)). First, the operator creates a gradient array $\nabla \overrightarrow{G}$ of the individuals chosen by the GA selection method in ascending order of fitness (from worst to best). Each position i of this array will contain whole values of 1 or -1, indicating whether the gene in position i tends to increase or decrease to optimize the problem, respectively.

Given a parent with genes $P_1 = \{x_1, x_2, \ldots, x_i\}$ and another parent with genes $P_2 = \{y_1, y_2, \ldots, y_i\}$, the descendents generated will be the result of the combination of the genes of each parent, creating two offspring: $X_1 = \{g(1), g(2), \ldots, g(i)\}$ and $X_2 = \{h(1), h(2), \ldots, h(i)\}$, where:

$$f(x_i, y_i) = (x_i - y_i) \times \alpha$$

$$g(i) = \begin{cases} x_i + f(x_i, y_i), \nabla\overrightarrow{G[i]} = 1 \\ x_i - f(x_i, y_i), \nabla\overrightarrow{G[i]} = -1 \end{cases}$$

$$h(i) = \begin{cases} y_i + f(x_i, y_i), \nabla\overrightarrow{G[i]} = 1 \\ y_i - f(x_i, y_i), \nabla\overrightarrow{G[i]} = -1 \end{cases} \quad (2)$$

The variable α is the weight defined in ADX and may assume continuous values between 0 and 1. This weight is the adjustment level of the value added to the original gene. The pseudocode of the proposed crossover is shown in Fig. 2:

Figure 3 shows an example of crossover using ADX. The ADX creates offspring with adjusted genes based on the gradient array, the parent genes, and the weight α. This method increases the exploitation potential of the optimization and can avoid breaking variable restrictions. By doing smooth adjustments, the ADX can create better feasible individuals.

3. Correlation Adjustment Mutation

After the crossover of the GA, the less fit individuals are discarded, preserving the more fit for the next generation (elitism process). The ADX increases the exploitation potential of the GA, improving existing solutions for the problem. To increase the exploration capabilities for searching new solution areas of the search space, this section proposes the CAM operator.

The proposed method adjusts all the population according to the results of the correlation tests between the fitness and the gene. This will spread the population through the search space providing increased exploration to the GA.

CAM is a correlation adjustment algorithm used in the mutation phase of a generation in the GA. It uses data from the entire population to determine coefficients of adjustment for the genes (variables). Through a simple random selection of a subgroup of individuals, CAM attempts to readjust the genes of each individual selected to improve its fitness (as in Fig. 5).

CAM uses Kendall's tau correlation coefficient test,[28,29] which assesses the degree of similarity between two datasets. The Kendall coefficients may vary with continuous values from [–1, 1]. A positive value indicates a positive association. A negative value indicates a negative association. The greater the magnitude, the stronger the association.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table I. A matrix that the CAM algorithm assembles using the less fit individuals from the population at the end of each generation.

	Gene 1	**Gene 2**	**...**	**Gene *m***	**Fitness *F*(*x*)**
Individual 1	*G*(*1,1*)	*G*(*1,2*)	...	*G*(*1,m*)	*F*(*g11,g12...g1m*)
Individual 2	*G*(*2,1*)	*G*(*2,2*)	...	*G*(*2,m*)	*F*(*g21,g22...g2m*)
...	...	...	...	...	...
Individual *n*	*G*(*n,1*)	*G*(*n,2*)	...	*G*(*n,m*)	*F*(*gn1,gn2...gnm*)

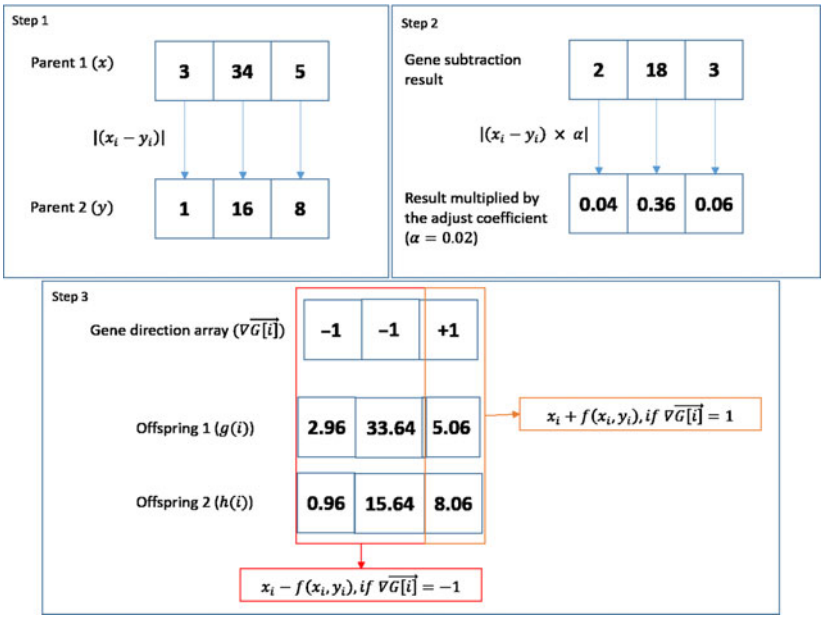


Fig. 3. Example of ADX crossover.

For example, consider data from a set n of pairs (x_1, y_1), $(x_2, y_2) \ldots (x_n, y_n)$. The Kendall tau coefficient equation is given by (Eq. 3):

$$\tau = \frac{(Z) - (Y)}{\frac{n(n-1)}{2}} \tag{3}$$

where:

- Z: number of concordant pairs (when $x_i > x_j$ and $y_i > y_j$ or $x_i < x_j$ and $y_i < y_j$)
- Y: number of discordant pairs (when $x_i > x_j$ and $y_i < y_j$ or $x_i < x_j$ and $y_i > y_j$)

After finalizing the crossover and mutation process, CAM will assemble a matrix $N \times (M + 1)$ of the individuals from the population used during the current generation, with N as rows and M as columns, as in Table I:

- N is the number of individuals
- M is the number of genes (variables) that comprise an individual
- The last column from the matrix will contain the fitness value corresponding to each individual in N

After creating the matrix, CAM will apply the Kendall correlation test between the Gene $\{1, 2, \ldots, m\}$ column in relation to data from the Fitness column. It is then possible to measure whether the variable will affect fitness in a concordant $(+)$ or discordant $(-)$ manner. The closer τ is to 1 or -1, respectively, the values will either have a strong positive or negative association.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

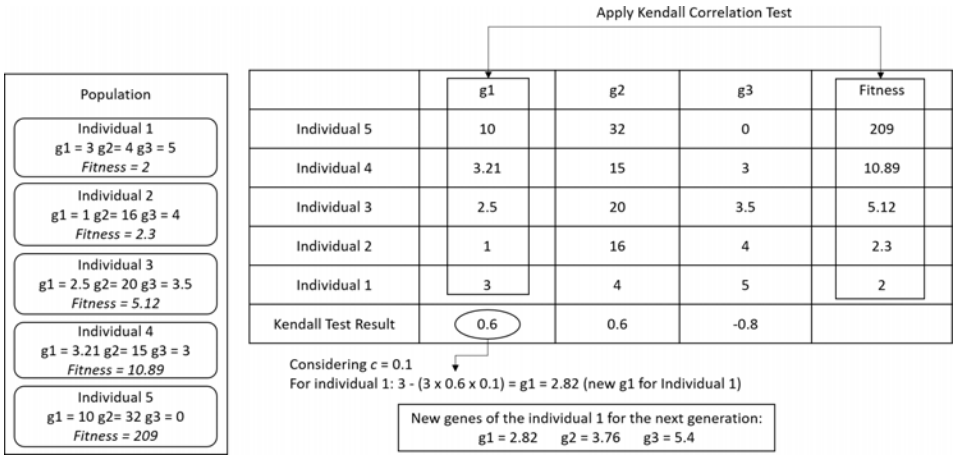


Fig. 4. Applying CAM on first gene (*g1*) of individual 1. The same process is applied on every gene (*g*) of each selected individual of the population before the next generation.



Fig. 5. After applying CAM, the dispersion of the population increased in the search space, and some individuals got closer to a possible optimal solution calculated based on the correlation matrix applied to the genes of the population. It is then possible to increase the level of diversity (search power) of a population without losing the capacity to intensify the results already obtained.

Consider that c (coefficient of precision) and k (number of iterations) are whole values manually selected in the algorithm. CAM will try to adjust the genes of an individual, such that for each individual i selected (Eq. 4):

$$\sum_{i=1}^{m} \sum_{0}^{k} G(i, m) = G(i, m) + G(i, m) \times \tau_m \times c \tag{4}$$

where:

- $G(i,m)$: cell from the matrix containing the value of gene m from individual i
- c: coefficient of precision
- m: gene index, according to Table I
- k: number of iterations on an individual's gene
- τ_m: results of the Kendall test for gene m in terms of fitness

Figure 4 illustrates the application of CAM with one iteration (k = 1) on the population before proceeding to the next generation of the GA (Figure 5).

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

4. ADX and CAM Computational Cost Analysis

The computational cost of the GA is given by time needed to create the population (*Tp*), evaluation of fitness (*Tf*), selection method (*Ts*), crossover operator (*Tc*), and mutation phase (*Tm*). Considering that *Tp* and *Ts* are the same for all the problems tested, *Tf* is problem-dependent; this section considers only *Tc* and *Tm* computational costs to analyze ADX and CAM effects.

Tc time relies on the crossover approach used. For SBX[19] and ARX,[18] both crossover operators use two parents to create two offspring. SBX swaps some genes between parents, and ARX creates a random value between the range of the parent gene values; we can call the swap/create phase of the crossovers *Tsc*. Each *Tsc* happens *n* times during *g* generations, where *Pn* is the number of each pair selected and *Gn* is the number of generations of the GA. The computational cost (*Cc*) of SBX and ARX can be given by (Eq. 5):

$$Cc = Gn \times Pn \times Tsc \tag{5}$$

The ADX crossover also creates values according to the parent gene (*Tsc*), but before the creation, the crossover creates and initializes a gradient array based on all selected individuals for the ADX operator; if we consider array initialization, the crossover adds an additional time *Ti* to the crossover phase. The ADX computation cost can be given by $Ti + Tsc$.

For each generation *Gn* of the algorithm, ADX uses $Gn \times Ti$ more time in comparison with other crossovers operators. The computational cost *Cc* of ADX can be given by (Eq. 6):

$$Cc = Gn[Ti + (Pn \times Tsc)] \tag{6}$$

The standard mutation phase verifies the chance of changing a random gene for everyone before going to the next generation. For each individual *i* there is a chance *r* that the gene will suffer a mutation (*Tm*), where *r* is generally a value between 0.01 and 0.05. The computational cost of mutation can be given by (Eq. 7):

$$Cc = Gn(i \times r \times Tm) \tag{7}$$

The CAM phase, unlike standard mutation, for each generation needs to build a correlation matrix $i \times j$ where *i* is the number of individuals selected for the next generation and *j* is the number of genes on an individual; this build phase is denoted by *Tcm*. After the matrix is built, the algorithm will change all the genes (*j*) of some *i* individuals of the population. If we consider *Tm* the time to change one gene, in the CAM phase we denote $j \times Tm$ the time to change all the genes of an individual. The computational cost *Cc* of the CAM phase in comparison with standard mutation can be given by (Eq. 8):

$$Cc = Gn\big[Tcm + (i \times j \times Tm)\big] \tag{8}$$

ADX and CAM add more computational cost for the overall performance of the GA, meaning that the algorithm will take more time and processing to finish execution compared with other crossovers (SBX and ARX) and standard mutation operators.

5. CEC'2015 Benchmark Problems

For benchmark problems, the GA tested the problems proposed in the Competition of Single-Objective Computationally Expensive Numerical Optimization of 2015 IEEE Congress on Evolutionary Computation (CEC'2015).[30] It consists of 15 optimization functions (F1–15) that are separated by the following category:

- Unimodal functions: F1, F2
- Simple multimodal functions: F3–F5
- Hybrid functions: F6–F8
- Composition functions: F9–F15

The proposed GA (ADX+CAM) results of the 15 optimization functions are presented in Tables II and III; according to ref. [30], the GA used 500 function evaluations (Fes) for 10D and 1500 for 30D, where D is the dimension of the solution matrix. Each value of the matrix is initially assigned

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table II. Results of CEC'2015 benchmark functions for 10D.

10D	Function	Best	Worst	Median	Mean	Std. Dev.
Unimodal	*F1*	1.16E+09	7.80E+09	4.78E+09	4.63E+09	1.98E+09
	F2	1.73E+04	1.48E+05	6.03E+04	7.19E+04	3.43E+04
Simple multimodal	*F3*	3.08E+02	3.13E+02	3.11E+02	3.11E+02	1.66E+00
	F4	1.37E+03	2.75E+03	1.95E+03	1.97E+03	3.70E+02
	F5	5.01E+02	5.04E+02	5.03E+02	5.03E+02	9.00E−01
	F6	6.01E+02	6.04E+02	6.03E+02	6.03E+02	7.57E−01
	F7	7.05E+02	7.38E+02	7.24E+02	7.23E+02	9.20E+00
	F8	8.81E+02	5.45E+04	4.34E+03	1.42E+04	1.74E+04
	F9	9.03E+02	9.05E+02	9.04E+02	9.04E+02	3.07E−01
Hybrid	*F10*	4.49E+04	1.71E+07	1.16E+06	2.21E+06	3.80E+06
	F11	1.11E+03	1.17E+03	1.12E+03	1.13E+03	1.90E+01
	F12	1.27E+03	2.76E+03	1.56E+03	1.60E+03	2.97E+02
Composition	*F13*	1.65E+03	2.25E+03	1.73E+03	1.83E+03	1.92E+02
	F14	1.60E+03	1.64E+03	1.61E+03	1.62E+03	1.16E+01
	F15	1.92E+03	2.17E+03	2.02E+03	2.02E+03	7.83E+01

Table III. Results of CEC'2015 benchmark functions for 30D.

30D	Function	Best	Worst	Median	Mean	Std. Dev.
Unimodal	F1	1.68E+10	3.28E+10	2.22E+10	2.31E+10	4.21E+09
	F2	7.02E+04	4.64E+05	1.90E+05	2.10E+05	9.65E+04
Simple multimodal	F3	3.31E+02	3.48E+02	3.39E+02	3.39E+02	3.93E+00
	F4	4.23E+03	6.60E+03	5.29E+03	5.40E+03	6.31E+02
	F5	5.02E+02	5.06E+02	5.04E+02	5.04E+02	9.67E−01
	F6	6.03E+02	6.04E+02	6.04E+02	6.04E+02	3.64E−01
	F7	7.24E+02	7.73E+02	7.41E+02	7.42E+02	1.22E+01
	F8	1.02E+04	2.14E+06	2.62E+05	3.84E+05	4.74E+05
	F9	9.13E+02	9.14E+02	9.14E+02	9.14E+02	2.17E−01
Hybrid	F10	4.54E+06	1.66E+08	2.00E+07	3.36E+07	4.22E+07
	F11	1.14E+03	1.37E+03	1.21E+03	1.22E+03	6.76E+01
	F12	1.81E+03	3.42E+03	2.40E+03	2.44E+03	4.58E+02
Composition	*F13*	1.65E+03	2.25E+03	1.73E+03	1.83E+03	1.92E+02
	F14	1.60E+03	1.64E+03	1.61E+03	1.62E+03	1.16E+01
	F15	1.92E+03	2.17E+03	2.02E+03	2.02E+03	7.83E+01

by a random number between [–100, 100]. For 10D tests, we used a population with five individuals through 100 generations; for 30D tests, the parameters were increased to six individuals in the population with 250 generations. Only one individual is passed through generations by elitism, and tournament operator was used for the selection method. The weights of ADX and CAM coefficient precision were empirically set to 0.01.

The following tables compare the mean results of best algorithms[30] submitted by participants of the CEC'2015 benchmark optimization competition. Best results are highlighted in bold.

Table IV shows the results for F1 and F2 unimodal functions; the proposed GA (ADX+CAM) achieved better results than humanCog on F1 (30D) and F2 (10D).

In the simple multimodal category (Table V), for the F4 problem, ADX+CAM obtained better performance than humanCog in 10D and achieved the best performance in 30D, falling behind only for MVMO. Also, the proposed algorithm got better result in F8 (30D) than the humanCog. For F3, F5, F6, F7, F8 (10D) and F9 problems, the ADX+CAM showed worse performance compared with other contestants.

For hybrid and composite functions in Tables VI and VII, the ADX+CAM only got better result in F10 (30D) than the humanCog. For all other functions, the proposed method failed to keep up with

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table IV. Results of unimodal functions means comparison.

	Unimodal functions			
	F1		**F2**	
	10D	**30D**	**10D**	**30D**
MVMO[31]	**1.93E+02**	**2.09E+03**	**1.68E−02**	**6.93E−03**
TunedCMAES[32]	1.17E+06	1.52E+06	4.78E+04	1.44E+05
CMAS-ES_QR[30]	4.43E+06	8.50E+05	2.58E+04	9.17E+04
iSRPSO[33]	7.40E+06	7.19E+08	3.19E+04	7.67E+04
humanCog[34]	3.27E+09	4.74E+10	7.80E+04	1.13E+05
ADX+CAM	4.63E+09	2.31E+10	7.19E+04	2.10E+05

Table V. Results of simple multimodal functions means comparison.

	Simple multimodal functions					
	F3		**F4**		**F5**	
	10D	**30D**	**10D**	**30D**	**10D**	**30D**
MVMO[31]	9.40E+00	3.79E+01	**4.65E+02**	**1.43E+03**	**1.13E+00**	**1.68E+00**
TunedCMAES[32]	7.62E+00	2.43E+01	1.34E+03	6.11E+03	2.77E+00	3.13E+00
CMAS-ES_QR[30]	**2.79E+00**	**1.15E+01**	1.73E+03	6.68E+03	3.20E+00	4.55E+00
iSRPSO[33]	6.60E+00	2.57E+01	9.25E+02	5.41E+03	2.46E+00	4.24E+00
humanCog[34]	1.12E+01	4.13E+01	2.09E+03	7.99E+03	2.82E+00	4.39E+00
ADX+CAM	3.11E+02	3.39E+02	1.97E+03	5.40E+03	5.03E+02	5.04E+02

	F6		**F7**		**F8**		**F9**	
	10D	**30D**	**10D**	**30D**	**10D**	**30D**	**10D**	**30D**
MVMO[31]	**3.26E−01**	**5.20E−01**	6.37E−01	**4.39E−01**	4.14E+01	4.03E+02	4.01E+00	**1.34E+01**
TunedCMAES[32]	6.00E−01	7.16E−01	6.31E−01	7.28E−01	3.68E+01	2.84E+01	4.17E+00	1.39E+01
CMAS−ES_QR[30]	4.17E−01	7.28E−01	**5.52E−01**	7.47E−01	**4.68E+00**	**1.74E+01**	3.96E+00	**1.34E+01**
iSRPSO[33]	5.29E−01	6.35E−01	5.71E−01	5.68E−01	5.03E+00	6.26E+02	**3.95E+00**	1.36E+01
humanCog[34]	3.63E+00	5.03E+00	2.74E+01	8.86E+01	7.77E+03	5.24E+06	4.16E+00	1.39E+01
ADX+CAM	6.03E+02	6.04E+02	7.23E+02	7.42E+02	1.42E+04	3.84E+05	9.04E+02	9.14E+02

Table VI. Results of hybrid functions means comparison.

	Hybrid functions					
	F10		**F11**		**F12**	
	10D	**30D**	**10D**	**30D**	**10D**	**30D**
MVMO[31]	**4.97E+02**	**9.29E+04**	1.17E+01	1.43E+02	2.00E+02	8.60E+02
TunedCMAES[32]	5.38E+05	4.89E+06	7.45E+00	**2.11E+01**	2.39E+02	7.66E+02
CMAS-ES_QR[30]	2.25E+05	3.25E+06	7.63E+00	2.46E+01	2.35E+02	**6.27E+02**
iSRPSO[33]	3.53E+05	6.83E+06	**7.26E+00**	5.09E+01	**1.82E+02**	7.36E+02
humanCog[34]	1.19E+06	5.60E+07	2.16E+01	2.76E+02	3.08E+02	1.60E+03
ADX+CAM	2.21E+06	3.36E+07	1.13E+03	1.22E+03	1.60E+03	2.44E+03

other algorithms; it is noticeable that in the composite functions, the algorithm stuck in local optima. That happened because of the very limited function evaluation limit, where CAM could not increase well the average diversity of the GA population.

To check the diversity effect of CAM on the GA population, we mapped population crowding based on the first two genes (*x1* and *x2*) of F1 (30D) and F14 (30D) during the generations of ADX

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table VII. Results of composite functions means comparison.

	Composite functions					
	F13		F14		F15	
	10D	30D	10D	30D	10D	30D
MVMO[31]	**3.16E+02**	**3.44E+02**	2.06E+02	2.76E+02	4.76E+02	1.19E+03
TunedCMAES[32]	3.47E+02	4.15E+02	2.05E+02	2.47E+02	4.42E+02	8.01E+02
CMAS-ES_QR[30]	3.26E+02	3.80E+02	**1.97E+02**	**2.35E+02**	3.79E+02	**4.90E+02**
iSRPSO[33]	3.31E+02	4.00E+02	2.01E+02	2.65E+02	**3.00E+02**	9.51E+02
humanCog[34]	4.33E+02	8.35E+02	2.15E+02	3.94E+02	4.74E+02	1.49E+03
ADX+CAM	1.83E+03	1.83E+03	1.62E+03	1.62E+03	2.02E+03	2.02E+03

and ARX crossovers. For both problems in Figs. 6 and 7, the scatter plots on top of the figures uses CAM; they present a more spread-out population than the bottom plots (without CAM) as the GA generations advance. It means that the GA was more successful in search space exploration when used with CAM. Comparing only the crossover operators without the use of CAM (bottom plots), ADX shows more exploration potential than ARX.

6. Robotic Inverse Kinematic Problem Analysis

Inverse kinematics (IK) have become a fundamental topic of research in robotic manipulators. Many of the IK problems have highly complex, nonlinear, coupled, and multiple solutions. There are many works on point-to-point trajectory generation of redundant and non-redundant manipulators, or continuous motion planning of redundant manipulators, whereas non-redundant manipulators with continuous path planning have received little attention among the research community.[35,36] To analyze the GA (ADX+CAM) proposed in this paper, we compare with the GA proposed by ref. [35], applying the algorithm on a continuous path planning of manipulators.

6.1. Path planning problem formulation

Consider a robot with M degrees of freedom and N task space coordinates, and a given desired path Pdc. The objective is to find a generated path Pgc where the accumulative deviation between Pdc and Pgc is minimum. Nk are the number of joints i (sampling points) that specify the accuracy of the desired path, and it is specified by the programmer. Pdc and Pgc are matrices of N by Nk where Pgc is calculated from a matrix P_Θ of M by Nk dimension, composed of feasible angles generated by the GA.

The deviation $E(i)$ at some point i is given by (Eq. 9):

$$E(i) = \sum_{k=1}^{N} |Pdc\,(k, i) - Pgc(k, i)| \tag{9}$$

The cumulative deviation (fitness function) between the two paths (desired and generated) is given by (Eq. 10):

$$E = \sum_{i=1}^{Nk} \sum_{k=1}^{N} |Pdc\,(k, i) - Pgc(k, i)| \tag{10}$$

The desired path (Pdc) for this problem with 20 joints ($Nk = 20$) is given by (Eq. 11) and shown in Fig. 8:

$$\begin{gathered} Pdc(1, i) = Xdc(i) = x_{\text{initial}} + \tfrac{x_{\text{final}} + x_{\text{initial}}}{Nk-1} \times (i-1) \\ Pdc(2, i) = Ydc(i) = 0.25 \\ x_{\text{initial}} = 0;\ \ x_{\text{final}} = 0.25 \end{gathered} \tag{11}$$

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

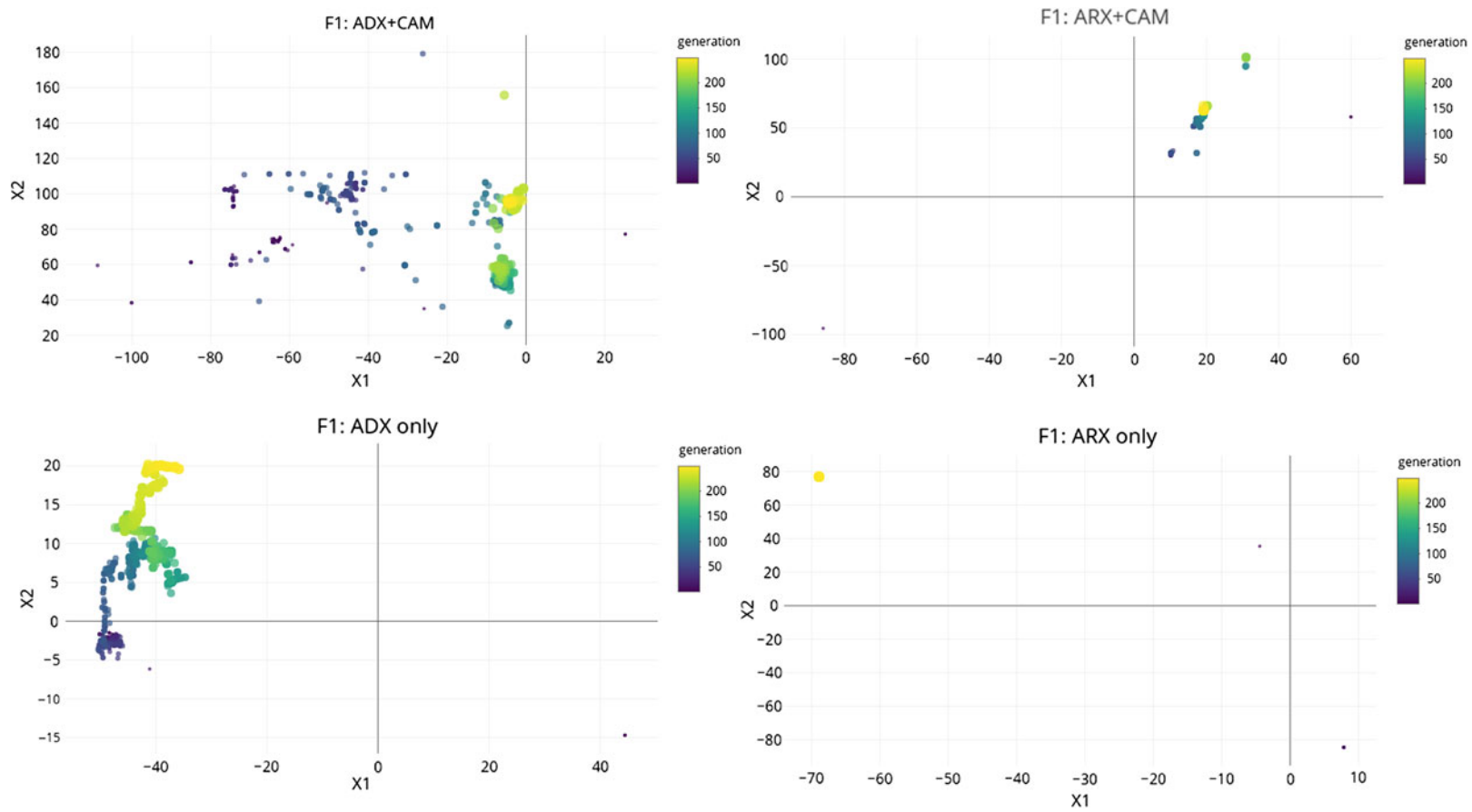


Fig. 6. F1 (30D) problem population crowding through GA generations; plots on the left are from ADX and ARX on the right side.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

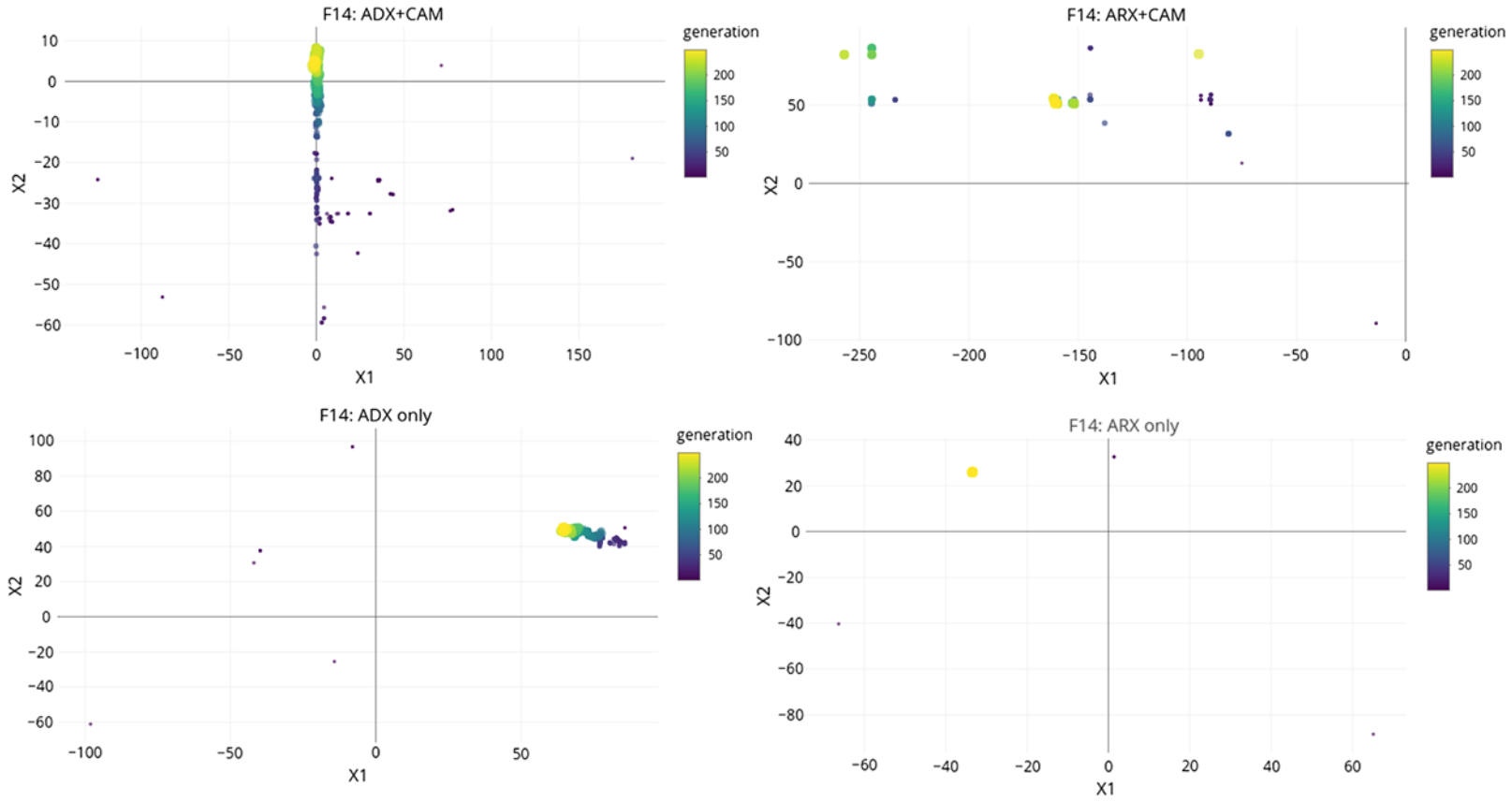


Fig. 7. F14 (30D) problem population crowding through GA generations; plots on the left are from ADX and ARX on the right side.

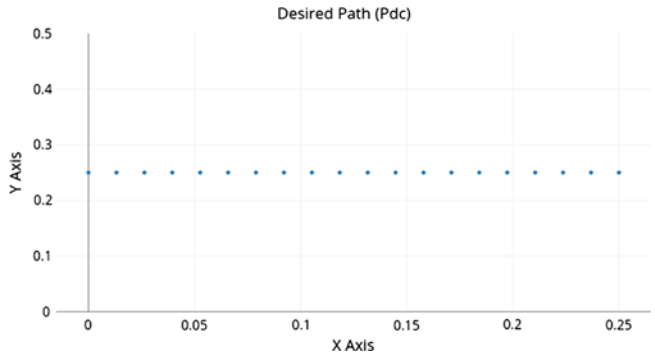


Fig. 8. Desired path (Pdc).

The kinematic models of the manipulator that will generate *Pgc* is given by (Eq. 12):

$$Xgc(i) = \sum_{j=1}^{M}\left(L_j \times \cos\left[\sum_{k=1}^{j}\theta_{k,i}\right]\right)$$
$$Ygc(i) = \sum\nolimits_{j=1}^{M}\left(L_j \times \sin\left[\sum\nolimits_{k=1}^{j}\theta_{k,i}\right]\right) \quad (12)$$

The parameters of 3R planar redundant manipulator are given as: $L1 = L2 = L3 = 0.5$; $N = 2$; $M = 3$; $-180 \leq \theta(h) \leq 180$ for $h = 1, 2, 3$; $Nk = 20$; $1 \leq i \leq Nk$.

6.2. 3R redundant manipulator problem results

We conducted 120 executions for this experiment, 20 for each crossover operator (ADX, SBX, and ARX) with CAM and 20 for each without CAM. For this IK problem, we applied separated GA subroutines on each column i of the P_Θ matrix, and the fitness function for each subroutine is the minimum deviation with the corresponding joint i. By splitting into smaller parts, the total population and generation was also divided between two joints, where all point i received 20 individuals and performed for 25 generations. Same parameters for population and generations are used from ref. [35].

The ADX crossover achieved best results compared with SBX and ARX; the cumulative deviation of ADX was <0.005, meaning that the generated path stayed on top or very close to the desired path (*Pdc*), shown in Fig. 9. CAM did not significantly improve the results for the GA using ADX, whereas compared with SBX configurations, *Pgc* was better when using CAM (Fig. 10), but the overall result for the SBX variation was worse than the other two crossovers. The ARX operator missed some joint points of *Pdc*, but the CAM helped creating a *Pgc* with better deviation than ARX only at some joint points (Fig. 11). Table VIII shows the best configuration found for this problem (found by ADX+CAM).

The fitness curves for all configurations achieved faster convergence when using CAM, especially for SBX and ARX; CAM improved the fitness values, providing fitness results closer to the optimum ($E = 0$) shown in Fig. 12.

IKs are well known in robotic optimization community; ADX+CAM proved to perform well for this problem compared with other crossovers, also reaching same result as ref. [35].

7. Engineering Optimization Problems

To perform the tests, five engineering problems were selected[3,4,37–40] due to their widespread use in the literature and for having characteristics of multi-objective functions with various constraints.

7.1. Design of pressure vessel (DPV)

DPV[3] concerns the problem of minimizing the total cost of function $f(x)$, including the costs of materials, forming, and welding of a cylindrical vessel capped at both extremities by hemispherical heads (Eq. 13). It has constraints in the variables: (T_s, x_1) is the thickness of the shell; (T_s, x_2) is the thickness of the head; (R, x_3) is the internal radius; (L, x_4) is the length of the cylindrical section of the vessel. The problem has four constraints g_{1-4}.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

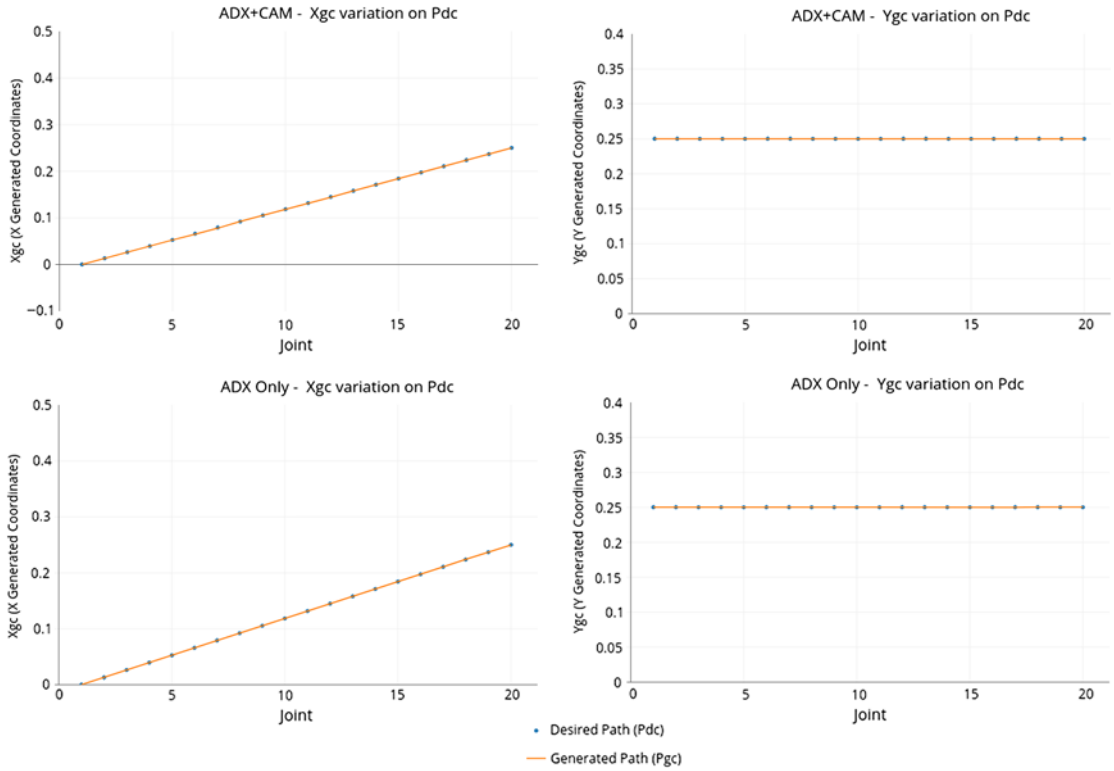


Fig. 9. ADX generated path (Pgc) compared with the desired path; ADX was able to find a path nearly identical to the desired path (Pdc).

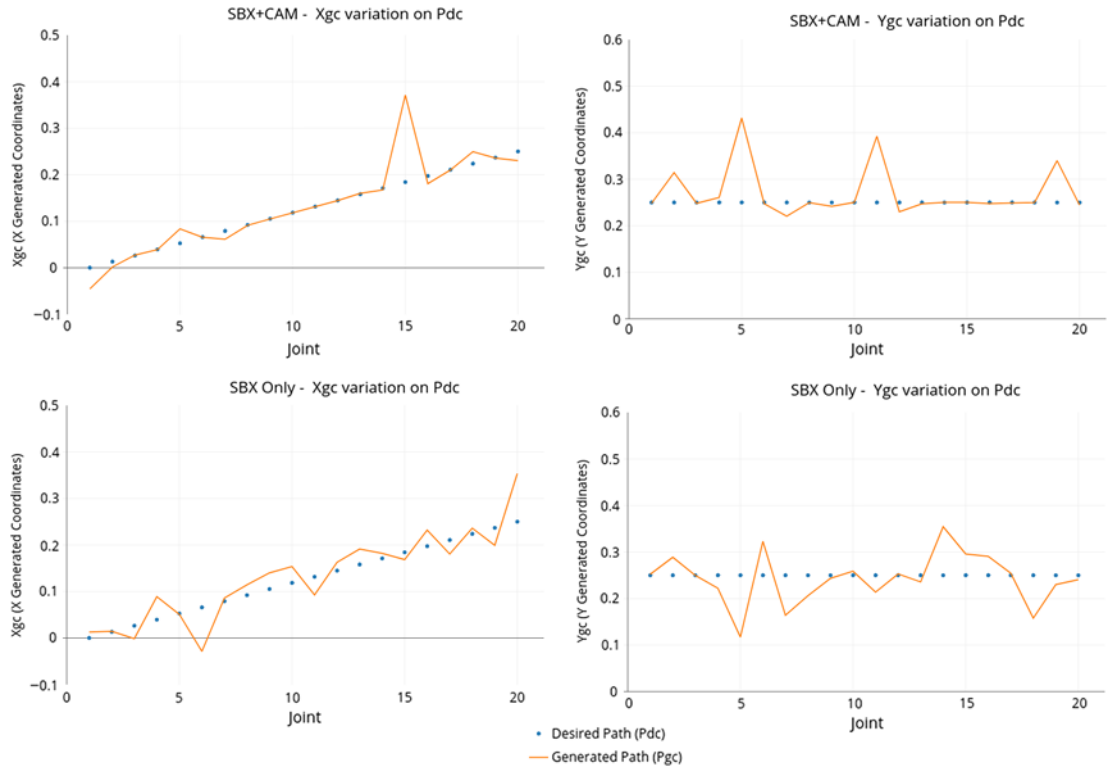


Fig. 10. SBX failed to find a better solution for the 3R redundant manipulator; the deviation was far from the optimal joint in some cases. The configurations with CAM (top side) achieved better results than SBX alone.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table VIII. Best solution to the 3R manipulator.

***Nk*(joint)**	***Pθ* (*M* = 3)**			***Pgc* (*N* = 2)**	
	ϑ_1	ϑ_2	ϑ_3	*Xgc*	*Ygc*
1	84.58663755	99.2398954	150.7892837	1.17*E*−05	0.250059139
2	85.61922796	103.086221	151.0746065	0.013138846	0.250049375
3	114.2616067	132.2130647	143.8781706	0.026435948	0.250059401
4	81.46650331	104.7391331	151.4081549	0.039442494	0.249992152
5	48.00699685	83.21534443	144.2344777	0.052576298	0.249958196
6	66.4017937	94.96775851	151.2184117	0.064721446	0.249799461
7	121.7339008	143.5690452	134.9242755	0.077813525	0.249839475
8	57.47721288	90.89468502	151.5295011	0.092337034	0.250348353
9	96.8534707	134.163299	145.5164409	0.105115727	0.250047758
10	8.398956245	80.9428497	130.9056913	0.118750851	0.249954193
11	64.44954803	105.4543423	154.8189469	0.131579235	0.249986724
12	58.51153224	100.1570313	155.5557187	0.144152013	0.249946299
13	177.5308716	155.7420472	91.77804148	0.157949669	0.250011596
14	48.37411046	90.83852409	156.5627239	0.170984385	0.250122902
15	101.0230329	149.9188693	136.6626152	0.184218357	0.24986417
16	95.30811879	149.5333462	139.3261017	0.197357989	0.249991647
17	89.47678319	148.8711804	142.2115573	0.210340186	0.249943925
18	73.14986278	139.8545406	152.0173438	0.223701118	0.249948287
19	69.90537288	139.8235808	153.524534	0.236789409	0.249991239
20	122.7452598	162.3170825	113.5803746	0.250010064	0.249951106

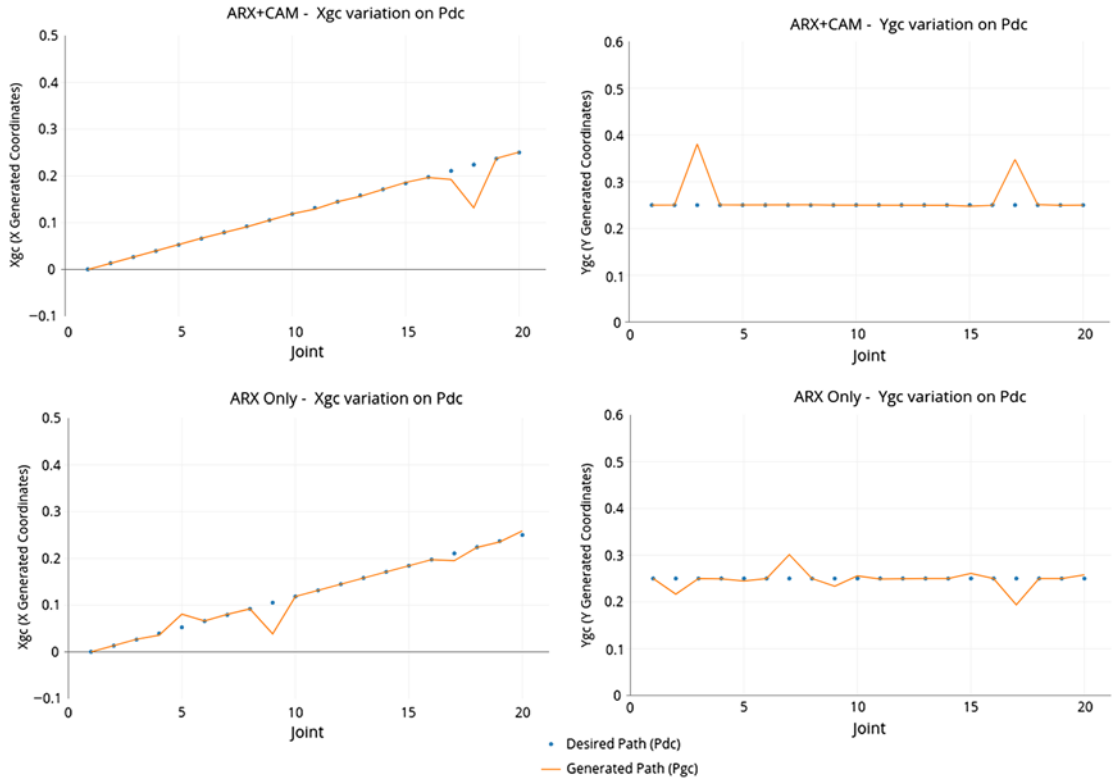


Fig. 11. ARX performed better than SBX, but worse than ADX. The crossover missed some joints. CAM was not able to improve the results on some cases (joints 18, 17 for Ygc and 3.17 for Xgc).

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

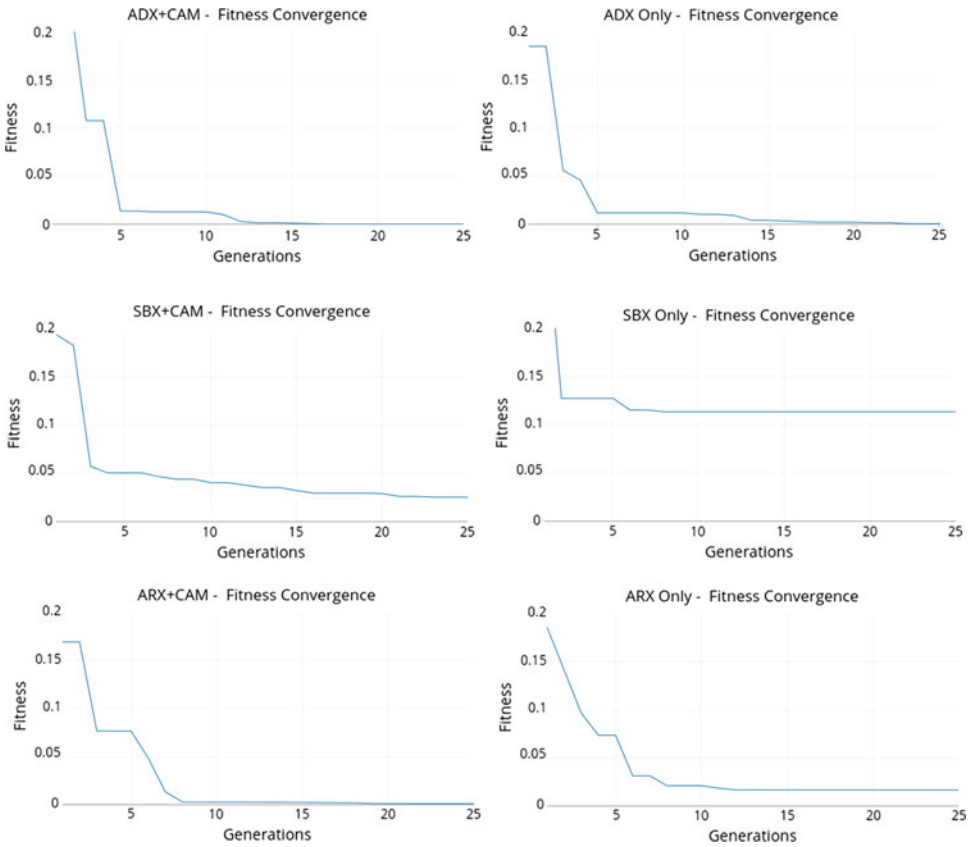


Fig. 12. Fitness curves for each crossover; the left side are configurations that also used CAM. SBX and ARX (middle and bottom rows, respectively) showed better optimum compared with configurations that did not use the correlation adjustment method (right side).

There are two variants for this problem. In the first one (DPV1),[3] x_4 goes from 10 to 240, and in the second one (DPV2),[4] assume x_4 between 10 and 200.

$$\begin{gathered}
\text{Minimize } f(x) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \\
s.t. g_1(x) = -x_1 + 0.0193x_3 \leq 0 \\
g_2(x) = -x_2 + 0.00954x_3 \leq 0 \\
g_3(x) = -\pi x_3^2 x_4 - \tfrac{4}{3}\pi x_3^3 + 1296000 \leq 0 \\
g_4(x) = x_4 - 240 \leq 0 \\
1 \times 0.0625 \leq x_1, x_2 \leq 99 \times 0.0625; \;\; 10 \leq x_3 \leq 200; \\
10 \leq x_4 \leq 240 \;\; (DPV1) \;\; or \; 200(DPV2)
\end{gathered} \tag{13}$$

7.2. Welded beam design (WBD)

For the WBD problem [3], the goal is to minimize function $f(x)$ with some constraints on the variables: (H, x_1) is the thickness of the weld; (t, x_2) is the width of the beam; (b, x_3) is the thickness of the beam; (l, x_4) is the length of the welded joint. The problem should obey the seven constraints g_{1-7} (Eq. 14).

The WBD problem also has two variants, WBD1 and WBD2, where WBD1[3] has seven restrictions and WBD2[4] has six.

$$\begin{gathered}
\text{Minimize } f(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2) \\
s.t. g_1(x) = \tau(x) - \tau_{\max} \leq 0 \\
g_2(x) = \sigma(x) - \sigma_{\max} \leq 0
\end{gathered}$$

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

$$g_3(x) = x_1 - x_4 \leq 0$$

$$g_4(x) = 0.125 - x_1 \leq 0$$

$$g_5(x) = \delta(x) - 0.25 \leq 0$$

$$g_6(x) = P - P_c(x) \leq 0$$

$$\tau(x) = \sqrt{\tau_1^2 + 2\tau_1\tau_2\left(\frac{x_2}{2R}\right) + \tau_2^2};\ \tau_1 = \frac{P}{\sqrt{2}x_1x_2};\ \tau_2 = \frac{MR}{J}$$

$$M = P\left(L + \frac{x_2}{2}\right);\ J(x) = 2\left\{\frac{x_1x_2}{\sqrt{2}}\left[\frac{x_2^2}{12} + \left(\frac{x_1+x_3}{2}\right)^2\right]\right\}$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1+x_3}{2}\right)^2};\ \sigma(x) = \frac{6PL}{x_4x_3^2}$$

$$\delta(x) = \frac{4PL^3}{Ex_3^3x_4};\ P_c(x) = \frac{4.013\sqrt{\frac{Ex_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)$$

$$G = 12 \times 10^6;\ E = 30 \times 10^6;\ P = 6000;\ L = 14$$

$$0.1 \leq x_1 \leq 2;\ 0.1 \leq x_2 \leq 10;\ 0.1 \leq x_3 \leq 10;\ 0.1 \leq x_4 \leq 2 \tag{14}$$

7.3. Minimize weight tension/compression on a spring (MWTCS)

In MWTCS,[3] the idea is to minimize the weight of tension/compression $f(x)$ on a spring, while obeying four constraints g_{1-4}, where (d, x_1) is the wire diameter, (D, x_2) is the mean coil diameter, and (P, x_3) is the number of active spools (Eq. 15).

$$\text{Minimize } f(x) = (x_3 + 2)x_2x_1^2$$

$$s.t. g_1(x) = 1 - \frac{x_2^3x_3}{71785x_1^4} \leq 0$$

$$g_2(x) = \frac{4x_2^2 - x_1x_2}{12566\left(x_2x_1^3 - x_1^4\right)} + \frac{1}{5108x_1^2} - 1 \leq 0$$

$$g_3(x) = 1 - \frac{140.45x_1}{x_2^2x_3} \leq 0$$

$$g_4(x) = \frac{(x_1+x_2)}{1.5} - 1 \leq 0$$

$$0.05 \leq x_1 \leq 2;\ 0.25 \leq x_2 \leq 1.3;\ 2 \leq x_3 \leq 15 \tag{15}$$

7.4. Speed reducer design (SRD)

The SRD[40] aims at minimizing $f(x)$ to design a speed reducer with 11 constraints g_{1-11}, where x_1 is the width of the face, x_2 is the tooth model, x_3 is the number of teeth in the pinion, x_4 is the length of the first shaft between the bearings, x_5 is the length of the second shaft between the bearings, x_6 is the diameter of first shaft, and x_7 is the diameter of second shaft (Eq. 16).

$$\begin{aligned}\text{Minimize } f(x) = {} & 0.7854x_1x_2^2\left(3.333x_3^2 + 14.9334x_3 - 43.0934\right) \\ & - 1.508x_1\left(x_6^2 + x_7^2\right) + 7.4777\left(x_6^2 + x_7^2\right) + 0.78054\left(x_4x_6^2 + x_5x_7^2\right)\end{aligned}$$

$$s.t. g_1(x) = \frac{27}{x_1x_2^2x_3} - 1 \leq 0$$

$$g_2(x) = \frac{397.5}{x_1x_2^2x_3} - 1 \leq 0$$

$$g_3(x) = \frac{1.93x_4^3}{x_2x_3x_6^3} - 1 \leq 0$$

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

$$g_4(x) = \frac{1.93x_5^3}{x_2x_3x_7^3} - 1 \leq 0$$

$$g_5(x) = \frac{1}{110x_6^3}\sqrt{\left(\frac{750x_4}{x_2x_3}\right)^2 + 16.9 \times 10^6} - 1 \leq 0$$

$$g_6(x) = \frac{1}{85x_7^3}\sqrt{\left(\frac{750x_5}{x_2x_3}\right)^2 + 157.5 \times 10^6} - 1 \leq 0$$

$$g_7(x) = \frac{x_2x_3}{40} - 1 \leq 0$$

$$g_8(x) = \frac{5x_2}{x_1} - 1 \leq 0$$

$$g_9(x) = \frac{x_1}{12x_2} - 1 \leq 0$$

$$g_{10}(x) = \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0$$

$$g_{11}(x) = \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0$$

$$2.6 \leq x_1 \leq 3.6;\ 0.7 \leq x_2 \leq 0.8;\ 17 \leq x_3 \leq 28;\ 7.3 \leq x_4 \leq 8.3$$

$$7.3 \leq x_5 \leq 8.3;\ 2.9 \leq x_6 \leq 3.9;\ 5 \leq x_7 \leq 5.5 \tag{16}$$

7.5. Three-bar truss (TBT)

The TBT[41] problem aims to minimize $f(x)$, with two continuous variables and three restrictions g_{1-3} as follows (Eq. 17):

$$\text{Minimize } f(x) = \left(2\sqrt{2}x_1 + x_2\right) \times l$$

$$s.t.\ g_1(x) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \leq 0$$

$$g_2(x) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \leq 0$$

$$g_3(x) = \frac{1}{x_1 + \sqrt{2}x_2}P - \sigma \leq 0$$

$$0 \leq x_1 \leq 1;\ 0 \leq x_2 \leq 1;\ l = 100;\ P = 2KN/cm^2;\ \sigma = 2KN/cm \tag{17}$$

7.6. Design of gear train (DGT)

This is a minimization unconstrained problem with four integer variables; it aims to minimize the gear ratio cost according to Eq. (18) [ref. 4].

$$\text{Minimize } f(x) = \left(\frac{1}{6.931} - \frac{x_1x_2}{x_3x_4}\right)^2$$

$$12 \leq x_1, x_2, x_3, x_4 \leq 60;\ x_i' \in \mathbb{Z}^+ \tag{18}$$

8. Analysis of Diversity and Results Obtained for Engineering Optimization Problems

Using CAM together with crossover ADX, we can obtain a population with a balanced level of diversity, enabling a broader sweep of the search space while performing local searches to enhance the results.

Diversity calculation, standard population diversity (SPD), describes the level of variation in a certain population. Through an analysis of the Euclidean distance of each individual in the GA in relation to a mean result of the population in the search space, it is possible to infer a certain diversity level. Greater diversity implies a greater variability of the population.[16,17]

First, the overall mean value for each gene comprising an individual is calculated. Next, the sum of the Euclidean distance of each individual to the general mean is calculated.

Consider a population with P individuals $(G_1, G_2, \ldots, G_p)$, given that each individual G has N genes, and thus $G_i = \left(G_{i,1}, G_{i,2}, \ldots, G_{i,N}\right)$. The general mean for each gene N is (Eq. 19):

$$G_N^{ave} = \frac{1}{P}\sum_{i=1}^{P} G_{i,N} \tag{19}$$

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Next, the sum of the Euclidean distance of each gene N from individual i in relation to the general mean G_N^{ave} is calculated by (Eq. 20):

$$SPD_i = \sqrt{\sum_{n=1}^{N} \left(G_{i,N} - G_N^{ave}\right)^2} \tag{20}$$

As SPD_i may have highly varied values depending on the type and size of the problem optimized by the GA, we apply normalization by calculating the standard deviation for each gene N in relation to population P in (Eq. 21):

$$\sigma\left(G_N^{ave}\right) = \sqrt{\frac{1}{P}\sum_{i=1}^{P} \left(G_{i,N} - G_N^{ave}\right)^2} \tag{21}$$

Lastly, we calculate the coefficient of variation C_v of the population in relation to the average individual G^{ave}. This coefficient indicates the variability of the population in each generation of the GA (Eq. 22):

$$SPD = C_v(G^{ave}) = \frac{1}{N}\sum_{j=1}^{N}\left(\frac{\sigma\left(G_j^{ave}\right)}{G_j^{ave}}\right) \tag{22}$$

To perform the tests, we applied three different types of crossover to the problems described above: one-cut point simulated binary crossover (SBX),[19] arithmetic crossover (ARX),[18] and adjusted crossover (ADX). For each crossover, the GA was executed 40 times, 20 with the CAM phase and 20 without.

The population used in each execution was randomly generated for the tests with and without CAM with 50 individuals with 500 generations. We chose the coefficient $\alpha = 0.05$ empirically for ADX and $c = 0.01$ for CAM. For the selection operator, we used the standard tournament selection and 5% of the population as elitism. The CAM phase replaced the mutation operator. For non-CAM tests, the mutation has a 1% chance to change the gene of some individuals by some random generated number between the gene value from the worst individual and the gene value from the best individual. Stopping criteria of the algorithm was set to 500 generations or the order of relative error equal to 10^{-6}. The maximum function evaluation on each execution is around $\sim 30{,}000$.

When an individual i breaks a restriction, the penalty function $P(i)$ adds value to the current individual fitness $Fit(i)$ [refs. 41,42]. The new fitness $NFit(i)$ will be calculated according to the following (Eq. 23):

$$NFit(i) = Fit(i) + P(i) \tag{23}$$

where $P(i)$ is denoted by (Eq. 24):

$$P(i) = bf \times nr \tag{24}$$

- bf is the fitness of the best individual of the generation
- nr is the number of restrictions that the current individual i broke

8.1. Design of pressure vessel

Figure 13 shows that in the DPV1 problem, all the configurations with CAM proved to have a stable high diversity level during all the executions of the algorithm, showing uniform distribution of the population in the search space for three different types of crossover. Without the CAM algorithm, the diversity of the population varied less, leading to worst fitness results for three different types of crossover (Table IX).

Values in Table IX show the mean SPD for each configuration. Configurations using CAM achieved better performance in search space exploration. High SPD mean values denote that the algorithm explored well the search space (exploration potential), avoiding local optima. It is important to be reminded that exploration potential alone does not mean that the algorithm achieved better fitness results.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table IX. Mean SPD levels for the configurations tested for DPV1.

Crossover	CAM	SPD mean
ADX	**Yes**	**0.046783466**
SBX	Yes	0.018913272
ARX	Yes	0.013920248
ADX	No	0.019719356
SBX	No	0.008308606
ARX	No	0.008320074

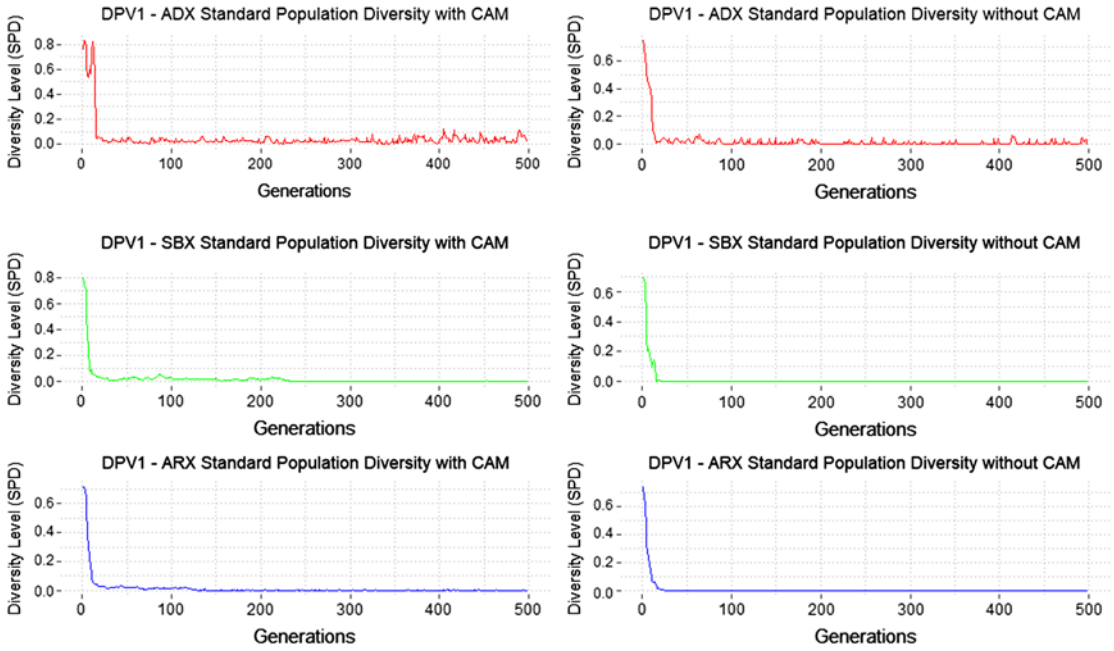


Fig. 13. Diversity plots with DPV1 configurations with (left plots) and without (right plots) CAM; with CAM algorithm, the diversity of the population varied better than without the CAM phase for all the crossovers tested.

Analyzing the best results for DPV1, ADX+CAM achieved the best and most reliable results of all the configurations, seen in Table X. Figure 14 shows the convergence plot of each configuration; the black dotted line shows the standard deviation limit in relation to mean fitness.

For the DPV2 variant, the diversity of ADX showed more variability in comparison with other crossover operators. SPD levels of ADX achieved 0.253794 with CAM and 0.016853 without. The population of SBX and ARX for both configurations had less variations, meaning that they probably have stuck in local optima as shown in Fig. 15. Table XI shows the SPD values of each configuration; ADX obtained better SPD values in comparison to other crossovers, showing better exploration potential.

Applying CAM, all crossovers obtained better results, but ADX+CAM achieved the best result of all the configurations as shown in Table XII and Fig. 16.

To confirm the results for both variants of the DPV problem, the p-values < 0.05[7,43] in Tables XIII and XIV show that one of the configurations in each pair of the table will always get better results, assuming the same number of population size and generations.

8.2. Welded beam design

In the WBD1 problem, the population of three configurations using CAM had better variability in all generations; ADX achieved higher SPD mean (0.111847917), followed by ARX = 0.0171753 and SBX = 0.014271732. In comparison with configurations without CAM, the obtained SPD mean with CAM are better. SBX and ARX had lower variability (SBX = 0.007802155 and

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table X. Mean results with standard deviations and best results obtained for three types of crossover for the DPV1 problem with and without CAM.

Crossover	CAM	Mean fitness	Std. Dev.	Best fitness	X1	X2	X3	X4
ADX	**Yes**	**5804.376773**	**0.001511001**	**5804.376257**	**0.727590953**	**0.359648586**	**37.69901308**	**239.9999799**
SBX	Yes	5820.066814	49.92834938	5804.376291	0.727590943	0.359648587	37.69901234	239.9999925
ARX	Yes	6074.118786	216.3093363	5804.376473	0.727590989	0.359648607	37.69901408	239.9999645
ADX	No	8895.457332	3795.00936	5831.165675	0.738372148	0.365271179	38.25116869	231.397417
SBX	No	13248.75369	3178.245363	7498.185704	1.125	0.5	51.40280072	96.46087191
ARX	No	16819.61528	3825.898879	9877.200681	1.031608158	0.988299592	46.41165417	154.1187151

ADX and SBX with CAM achieved the best and most reliable results of all the configurations without violating constraints. In the configuration without CAM, ADX achieved the best result.

Table XI. Mean SPD levels for the configurations tested for DPV2.

Crossover	CAM	SPD mean
ADX	**Yes**	**0.253794**
SBX	Yes	0.012102
ARX	Yes	0.012865
ADX	No	0.016853
SBX	No	0.015105
ARX	No	0.010446

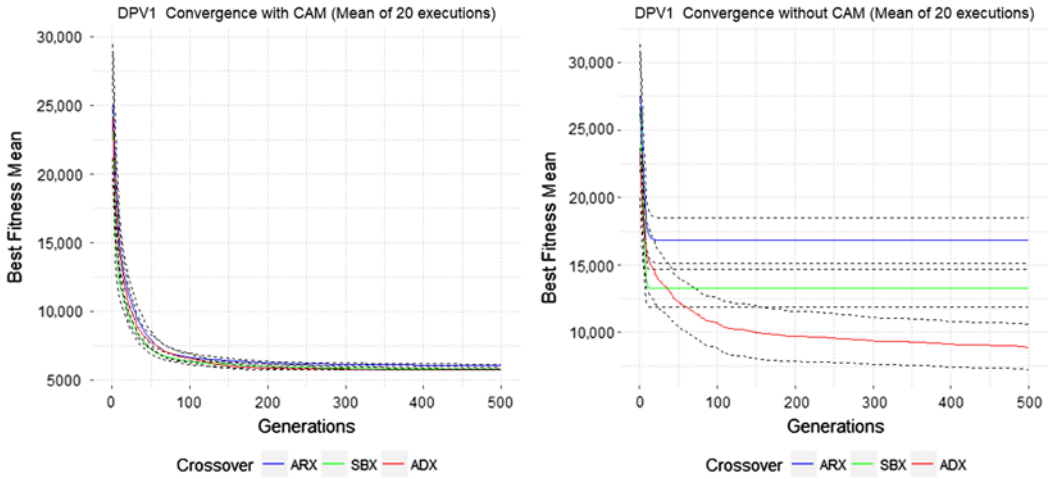


Fig. 14. Convergence graph obtained for the DPV1 problem with three types of crossover. The graph on the left used CAM, while the graph on the right did not. ADX obtained better results and faster convergence.

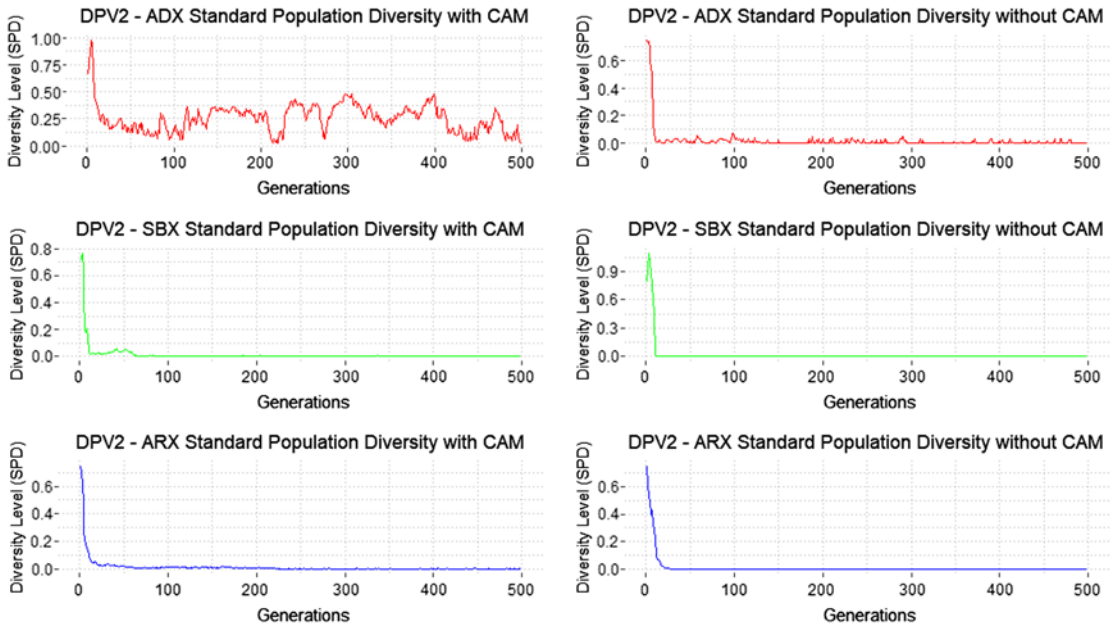


Fig. 15. Diversity plots with DPV2 configurations with (left plots) and without (right plots) CAM; with CAM algorithm, the diversity of the population varied better than without the CAM phase for all the crossovers tested.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table XII. Mean results with standard deviations and best results obtained for three types of crossover for the DPV2 problem with and without CAM.

Crossover	CAM	Mean fitness	Std. Dev.	Best fitness	X1	X2	X3	X4
ADX	**Yes**	**5885.333195**	**0.000136126**	**5885.332966**	**0.77816872**	**0.384649211**	**40.3196226**	**199.999946**
SBX	Yes	5912.699038	67.6716489	5885.332866	0.778168661	0.384649169	40.31961926	199.9999927
ARX	Yes	6240.630536	251.6894027	5885.333039	0.778168702	0.384649184	40.31962048	199.9999756
ADX	No	7148.721563	1345.430944	5899.012929	0.78587434	0.388584768	40.71887648	194.515903
SBX	No	12632.48067	3354.446078	7674.682181	0.928420318	0.674013419	41.66909929	182.0824132
ARX	No	14892.40824	3700.728735	7496.664921	1.05512907	0.418332934	43.5389206	159.5691149

ADX+ CAM obtained the best and most reliable results of all the configurations without violating constraints.

Table XIII. Friedman test (p-values) with all the configurations for DPV1.

	ADX w/ CAM	**SBX w/ CAM**	**ARX w/ CAM**	**ADX w/o CAM**	**SBX w/o CAM**
SBX w/ CAM	6.62E−06	NA	NA	NA	NA
ARX w/ CAM	1.99E−23	4.10E−13	NA	NA	NA
ADX w/o CAM	3.46E−32	8.39E−23	1.72E−05	NA	NA
SBX w/o CAM	1.35E−44	3.31E−37	1.72E−21	8.74E−12	NA
ARX w/o CAM	3.34E−50	1.07E−43	7.97E−30	7.27E−21	3.18E−05

Table XIV. Friedman test (p-values) with all the configurations for DPV2.

	ADX w/ CAM	**SBX w/ CAM**	**ARX w/ CAM**	**ADX w/o CAM**	**SBX w/o CAM**
SBX w/ CAM	0.032039571	NA	NA	NA	NA
ARX w/ CAM	1.33E−17	4.85E−13	NA	NA	NA
ADX w/o CAM	3.06E−26	5.79E−22	0.000101978	NA	NA
SBX w/o CAM	5.70E−40	1.31E−36	9.74E−21	8.08E−12	NA
ARX w/o CAM	8.17E−45	8.84E−42	2.18E−27	7.52E−19	0.001852786

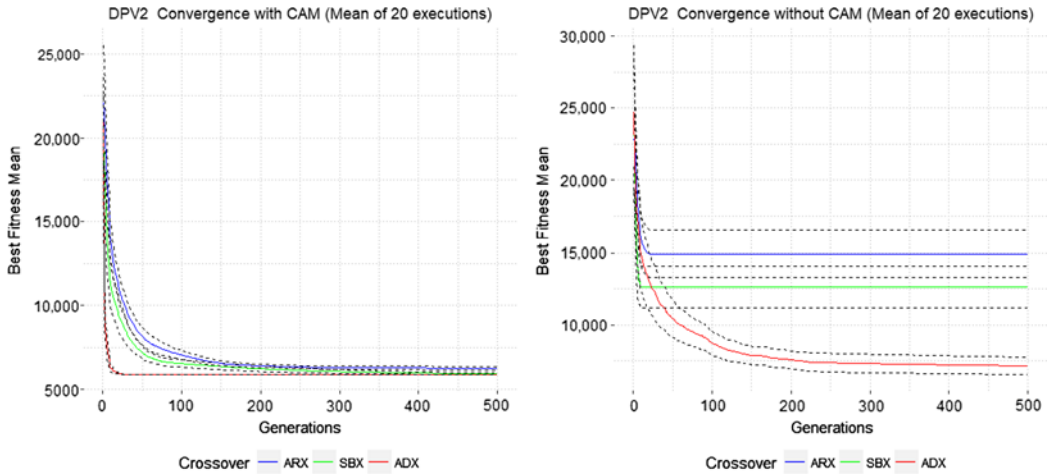


Fig. 16. Convergence graph obtained for the DPV2 problem with three types of crossover. The graph on the left used CAM, while the graph on the right did not. ADX obtained better results in relation to SBX and ARX.

ARX = 0.006849786), indicating high probability of trapping in local optima. Table XV shows higher SPD values with CAM for all configurations, indicating increased exploration potential, and Fig. 17 shows the variability of the population during the execution of GA.

For the WBD1 problem, CAM+ADX obtained the best result, with a major difference in relation to SBX and ARX shown in Table XVI and Fig. 18. Without CAM, ADX showed superior results compared with SBX and ARX.

The Friedman test for the WBD1 problem shows that ARX and SBX obtained similar behavior with and without CAM (p-value >0.05), while ADX showed better performance (Table XVII).

The diversity levels for the WBD2 problem show better variability using the CAM algorithm for ADX (Table XVIII). ADX without CAM achieved higher SPD (0.079875934) of all configurations. The variability of the population during the generations is shown in Fig. 19.

For the WBD2 variant, ADX+CAM obtained the best results, confidence interval, and faster convergence in comparison with other crossover operators (Table XIX) (Fig. 20). The CAM algorithm for SBX and ARX proved to be irrelevant according to $p = 0.197547421$ shown in Table XX.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table XV. Mean SPD levels for the configurations tested for WBD1.

Crossover	CAM	SPD mean
ADX	**Yes**	**0.111847917**
SBX	Yes	0.014271732
ARX	Yes	0.0171753
ADX	No	0.074437229
SBX	No	0.007802155
ARX	No	0.006849786

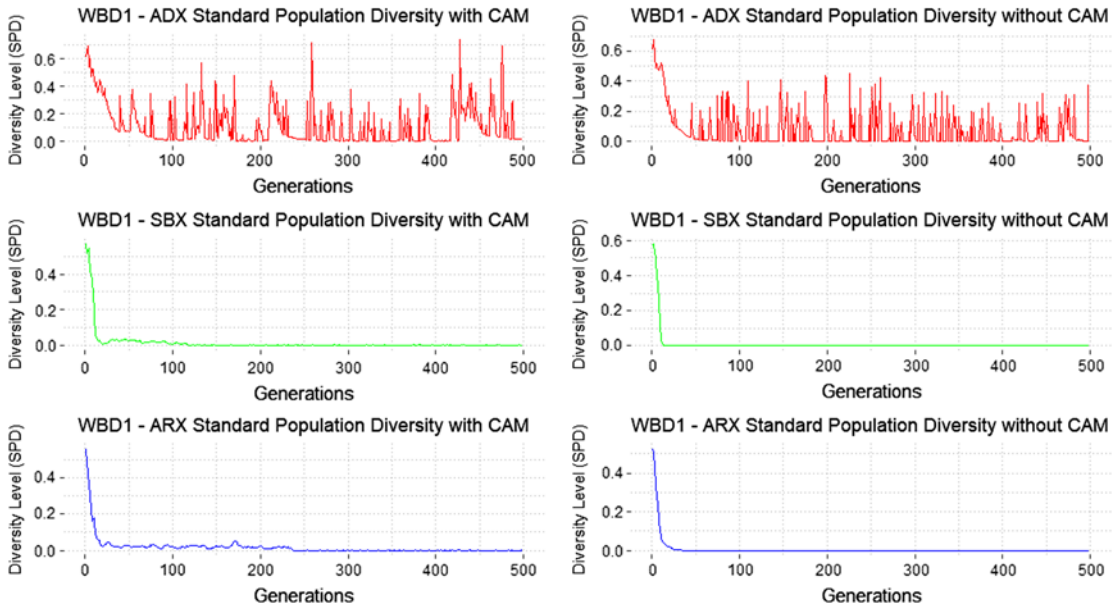


Fig. 17. Diversity plots with WBD1 configurations with (left plots) and without (right plots) CAM.

According to the Friedman test, ADX without CAM showed similar mean and results compared with SBX+CAM and ARX+CAM (p-value >0.05). Considering configurations with CAM, ARX had the same behavior as SBX also.

8.3. Minimize weight of tension/compression of a spring

In the MWTCS problem, all the crossovers obtained better SPD levels using CAM, achieving better variability in the population; ADX+CAM obtained higher SPD mean (1.667126838), followed by ARX+CAM (0.020345447) and SBX+CAM (0.008153023). ADX also achieved better SPD mean without the CAM algorithm as shown in Table XXI. Around generation 330, the SPD levels were higher on ADX+CAM, which happened because the population of the algorithm may have moderate number of infeasible individuals leading to very high fitness values due to the penalty function. The SPD variability throughout the generations can be observed in Fig. 21.

Comparing the best results obtained for MWTCS, as depicted in Table XXII, ADX+CAM obtained the best results (best fitness = 0.012665402) and better reliability (standard deviation = 2,06789E-07) in relation to SBX and ARX. All the crossovers achieved better results in less generations using CAM; the convergence curve of the MWTCS can be seen in Fig. 22.

The Friedman test shows that all the configurations were not similar to each other, thus confirming that ADX+CAM excels in the results (Table XXIII).

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table XVI. Mean results with standard deviations and the best results obtained for three types of crossover for the WBD problem with and without CAM.

Crossover	CAM	Mean fitness	Std. Dev.	Best fitness	X1	X2	X3	X4
ADX	**Yes**	**2.381194898**	**0.000385388**	**2.38095734**	**0.244369095**	**6.217518027**	**8.291469027**	**0.244369124**
SBX	Yes	2.573268148	0.189580495	2.390155897	0.243032341	6.078524799	8.491649257	0.243032779
ARX	Yes	2.540674969	0.229413567	2.381100828	0.244394728	6.216614302	8.291463407	0.24439473
ADX	No	2.477468347	0.169429462	2.380957299	0.244368865	6.217508198	8.291487671	0.244368865
SBX	No	4.233806903	0.867359629	2.996406072	0.337323901	5.667306346	7.026241366	0.343554324
ARX	No	4.241610503	0.741616166	3.226136684	0.351477463	4.753536657	6.876162573	0.415450468

Table XVII. Friedman test (p-values) with all the configurations for WBD1.

	ADX w/ CAM	**SBX w/ CAM**	**ARX w/ CAM**	**ADX w/o CAM**	**SBX w/o CAM**
SBX w/ CAM	3.12E−20	NA	NA	NA	NA
ARX w/ CAM	4.11E−16	**0.089105763**	NA	NA	NA
ADX w/o CAM	1.14E−06	7.72E−09	3.17E−05	NA	NA
SBX w/o CAM	1.45E−41	3.12E−20	2.79E−24	7.62E−33	NA
ARX w/o CAM	1.45E−41	3.12E−20	2.79E−24	7.62E−33	**1**

Table XVIII. Mean SPD levels for the configurations tested for WBD2.

Crossover	**CAM**	**SPD mean**
ADX	Yes	0.071542207
SBX	Yes	0.013876906
ARX	Yes	0.01669136
ADX	**No**	**0.079875934**
SBX	No	0.010430594
ARX	No	0.011155671

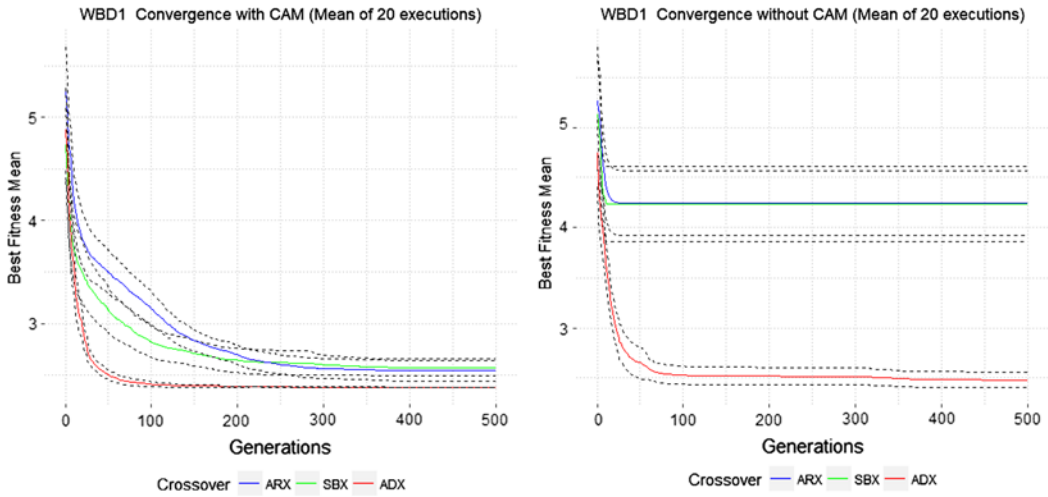


Fig. 18. Convergence graph obtained for the WBD1 problem with three types of crossover. The graph on the left used CAM, while the graph on the right did not. ADX obtained better results in relation to SBX and ARX.

8.4. Speed reducer design

For the SRD problem, Table XXIV shows that the ADX crossover obtained better SPD values using CAM, leading to a better chance of avoiding local optima. Both SBX and ARX did not achieve satisfactory results for SPD mean even when using CAM (nearly zero), showing that the population probably stuck in a local optimum. Figure 23 shows the SPD levels of the population during the execution of GA.

In the SRD problem, ADX+CAM achieved the best results (best fitness = 2894.382347), with better reliability (standard deviation = 0.00123656) as shown in Table XXV. Without the use of CAM, it was very likely that SBX and ARX became stuck in local optima as shown in the right plot (see Fig. 24).

The Friedman test in Table XXVI shows that SBX and ARX with CAM showed similar results (p-value >0.05).

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table XIX. Mean results with standard deviations and the best results obtained for three types of crossover for the WBD2 problem with and without CAM.

Crossover	CAM	Mean fitness	Std. Dev.	Best fitness	X1	X2	X3	X4
ADX	**Yes**	**1.695264346**	**5.26477E-05**	**1.695247207**	**0.205729633**	**3.253120167**	**9.036624068**	**0.205729641**
SBX	Yes	1.847086425	0.184058862	1.698384005	0.205575207	3.247453715	9.067656663	0.205575319
ARX	Yes	1.812662126	0.106905684	1.696550871	0.206090135	3.248859826	9.028716833	0.206090142
ADX	No	1.79672566	0.118298108	1.69525163	0.205730867	3.253105548	9.036596843	0.205730873
SBX	No	2.36868628	0.272667791	1.86184256	0.213648108	3.29484441	8.738486184	0.233217299
ARX	No	2.22187678	0.27141054	1.785544788	0.201140668	3.282486118	9.266044486	0.212715924

ADX+CAM achieved the best results without restriction violation.

Table XX. Friedman test (p-values) with all the configurations for WBD2.

	ADX w/ CAM	**SBX w/ CAM**	**ARX w/ CAM**	**ADX w/o CAM**	**SBX w/o CAM**
SBX w/ CAM	3.57E−20	NA	NA	NA	NA
ARX w/ CAM	1.90E−16	**0.197547421**	NA	NA	NA
ADX w/o CAM	4.94E−18	**0.580611184**	**0.580611184**	NA	NA
SBX w/o CAM	1.32E−41	3.57E−20	6.88E−24	2.55E−22	NA
ARX w/o CAM	3.54E−37	2.53E−14	4.94E−18	1.90E−16	0.017266559

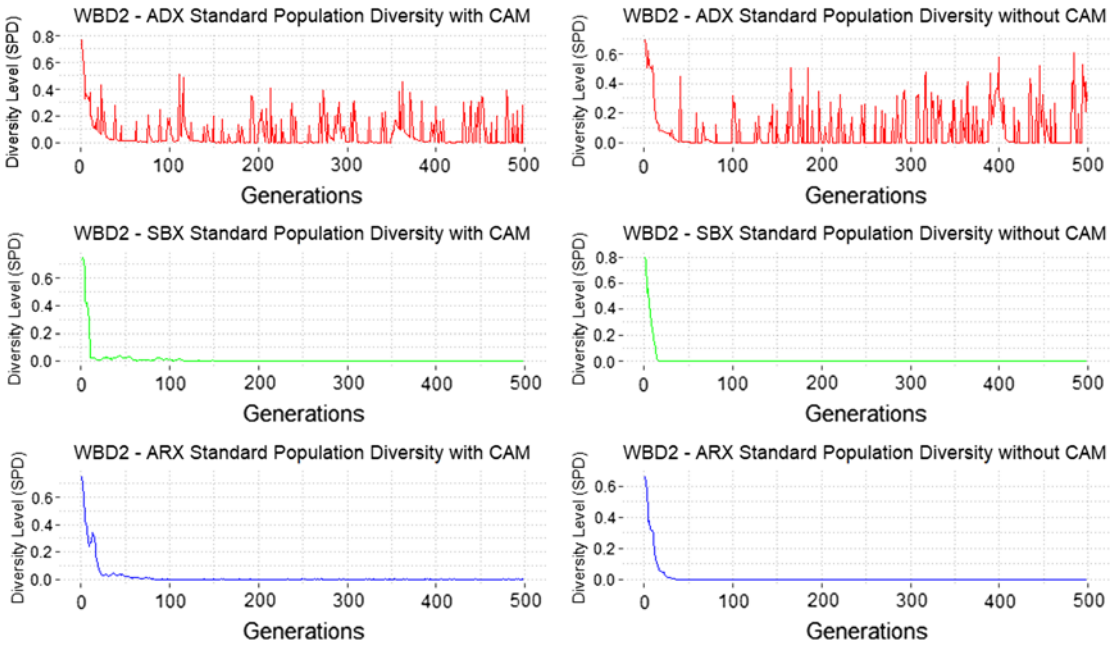


Fig. 19. Diversity plots with WBD2 configurations with (left plots) and without (right plots) CAM.

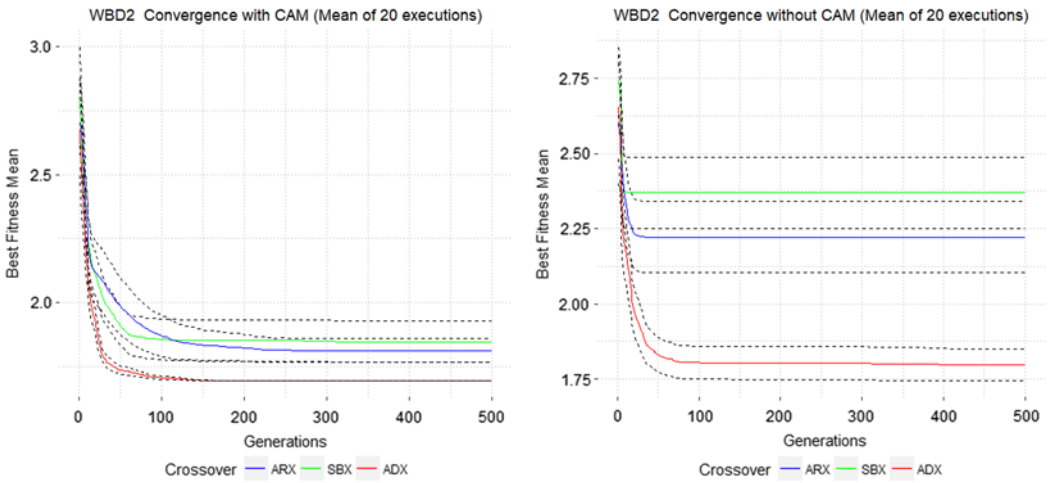


Fig. 20. Convergence graph obtained for the WBD2 problem with three types of crossover. The graph on the left used CAM, while the graph on the right did not. ADX obtained better results and confidence interval in relation to SBX and ARX.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table XXI. Mean SPD levels for the configurations tested for MWTCS.

Crossover	CAM	SPD mean
ADX	**Yes**	**1.667126838**
SBX	Yes	0.008153023
ARX	Yes	0.020345447
ADX	No	0.622638332
SBX	No	0.005587842
ARX	No	0.004934748

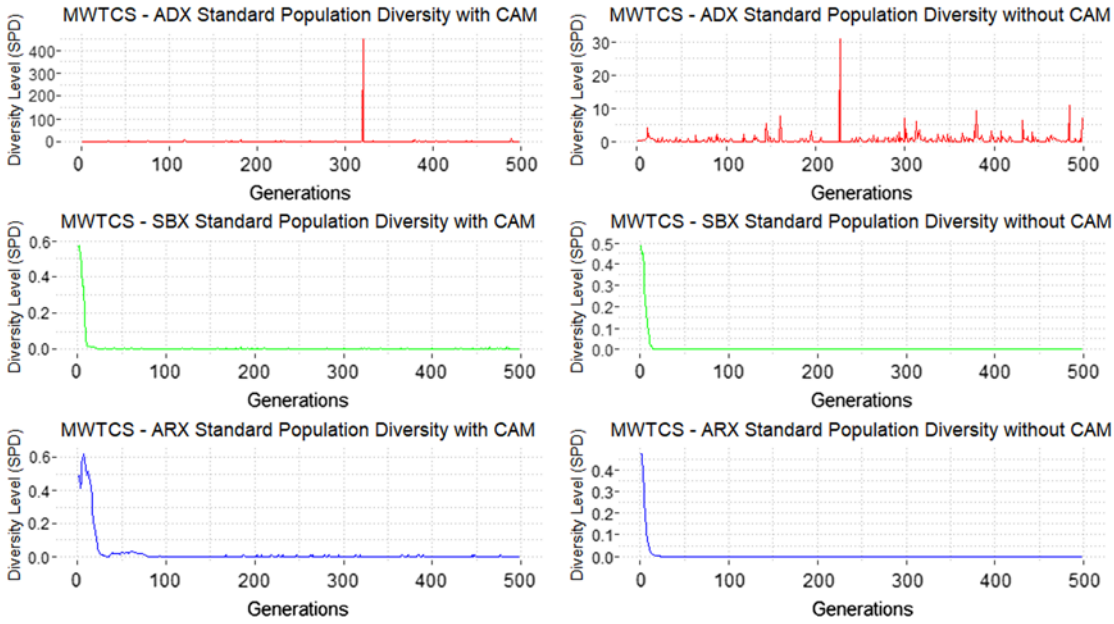


Fig. 21. SPD for each configuration used in MWTCS. The graphs in red (ADX) achieved the best SPDs in relation to other crossover methods (SBX and ARX).

8.5. Three-bar truss

For the population diversity of TBT, ADX obtained better SPD values with and without CAM (Table XXVII). SBX and ARX did not achieve better SPD levels, probably leading to local optima. The variability of SPD values for the population during the generation is shown in Fig. 25.

For the TBT problem, the ADX+CAM algorithm achieved best results compared with other crossovers as shown in Fig. 26 and Table XXVIII.

The Friedman test in Table XXIX shows that ARX without CAM had a p-value >0.05 compared with SBX+CAM, meaning they showed similar performance.

8.6. Design of gear train

ADX obtained better population diversity in comparison to SBX and ARX (Fig. 27); because of the integer nature of the variables, SBX and ARX failed to create a diverse population. The SPD results are shown in Table XXX.

The convergence of this problem was very similar to all configurations because of the integer variables (Fig. 28), but when comparing fitness results, ADX+CAM showed better results (Table XXXI).

Friedman test shows that ADX had the same performance with and without the CAM algorithm (p-value >0.05), meaning that CAM may not directly affect the results of the ADX operator executions (Table XXXII).

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table XXII. Mean results with standard deviations and the best results obtained for three types of crossover for the MWTCS problem with and without CAM.

Crossover	CAM	Mean fitness	Std. Dev.	Best fitness	X1	X2	X3
ADX	**Yes**	**0.012665402**	**2.06789E-07**	**0.012665234**	**0.051696686**	**0.356901202**	**11.27821854**
SBX	Yes	0.012700556	4.62379E-05	0.012665246	0.051663186	0.35609556	11.32553644
ARX	Yes	0.012702066	3.58667E-05	0.01266527	0.051644215	0.355639818	11.35244193
ADX	No	0.012872378	0.000220028	0.012678427	0.052545669	0.377677494	10.15823144
SBX	No	0.016561342	0.002225919	0.013559403	0.055650662	0.450778116	7.712635362
ARX	No	0.016425844	0.002798739	0.013403294	0.05593711	0.456646993	7.380606059

ADX+CAM achieved the best and most reliable results for all the configurations without violating constraints.

Table XXIII. Friedman test (p-values) for the configurations tested for MWTCS.

	ADX w/ CAM	**SBX w/ CAM**	**ARX w/ CAM**	**ADX w/o CAM**	**SBX w/o CAM**
SBX w/ CAM	1.79E−12	NA	NA	NA	NA
ARX w/ CAM	9.13E−21	0.000226533	NA	NA	NA
ADX w/o CAM	1.35E−31	3.05E−15	3.92E−07	NA	NA
SBX w/o CAM	1.46E−50	3.15E−39	9.87E−33	4.36E−22	NA
ARX w/o CAM	3.09E−47	5.64E−35	5.78E−28	1.35E−16	0.008551604

Table XXIV. Mean SPD levels for the configurations tested for SRD.

Crossover	**CAM**	**SPD mean**
ADX	**Yes**	**0.009205031**
SBX	Yes	0.000829004
ARX	Yes	0.000696936
ADX	No	0.005598273
SBX	No	0.000545229
ARX	No	0.000481946

The best mean SPD levels with and without CAM were for ADX.

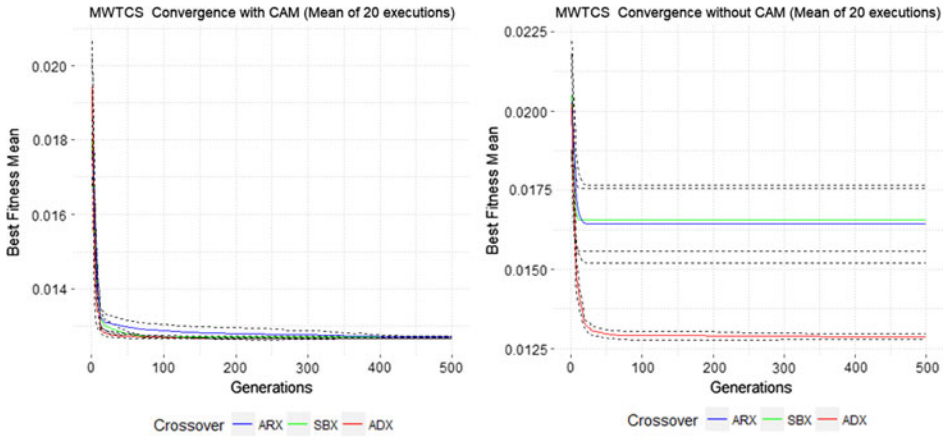


Fig. 22. Convergence graph obtained for the MWTCS problem with three types of crossover. The graph on the left used CAM, while the graph on the right did not. ADX+CAM obtained better results.

8.7. Conclusions of the diversity analysis and Friedman test

From analyses, it is possible to infer that population SPD of the GA is a very important factor in the global search for improved optimization. Configurations that had a high SPD were able to have better and more reliable results in relation to the simulated binary (SBX) and arithmetic (ARX) crossovers without CAM. ADX+CAM proved to have a better potential in the variation of diversity in global and local searches for optimizing the problems. According to Friedman test, the proposed work, compared with other SBX and ARX configurations, achieved a p-value <0.05 for all GA configurations, except for the DGT problem where we can assume that ADX+CAM had equal performance compared with ADX without CAM. Within the tests, the proposed work showed superior performance than other crossovers.

9. Analysis of Results Obtained for Other Classic Evolutionary Algorithms

To compare the proposed modified GA (ADX+CAM) with other classic optimization algorithms (PSO and simulated annealing), we set a population of 50 individuals with 500 generations and maximum of 30,000 function evaluations (FES); the same number of FES were set for a standard PSO and simulated annealing algorithms. The stopping criteria was 500 generations or relative error of order 10^{-6}.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table XXV. Mean results with standard deviations and the best results obtained for three types of crossover for the SRD problem with and without CAM.

Crossover	**CAM**	**Mean fitness**	**Std. Dev.**
ADX	**Yes**	**2894.382754**	**0.00123656**
SBX	Yes	2899.508371	17.28414998
ARX	Yes	2902.467239	17.57607257
ADX	No	2901.142232	15.37929146
SBX	No	3017.023008	56.76004703
ARX	No	3071.630338	38.70176863

Crossover	**CAM**	**Best fitness**	**X1**	**X2**	**X3**	**X4**	**X5**	**X6**	**X7**
ADX	**Yes**	**2894.382347**	**3.500000016**	**0.700000003**	**17.00000032**	**7.300000023**	**7.715320312**	**2.90000011**	**5.286654557**
SBX	Yes	2894.382284	3.500000031	0.700000004	17.0000003	7.30000035	7.715320026	2.900000049	5.286654469
ARX	Yes	2894.382304	3.500000075	0.70000001	17.00000014	7.300000352	7.715320183	2.900000028	5.286654472
ADX	No	2894.415945	3.500017471	0.700000334	17	7.300005315	7.715418416	2.9	5.286690856
SBX	No	2952.238809	3.564102727	0.701753597	17.00676682	7.465309122	7.845560158	2.94216127	5.303896827
ARX	No	3006.706731	3.555847898	0.702628428	17.23014312	7.799095756	8.048648956	2.974610109	5.305003096

Table XXVI. Friedman test (p-values) for the configurations tested for SRD.

	ADX w/ CAM	**SBX w/ CAM**	**ARX w/ CAM**	**ADX w/o CAM**	**SBX w/o CAM**
SBX w/ CAM	2.59E−05	NA	NA	NA	NA
ARX w/ CAM	0.005236215	**0.09508197**	NA	NA	NA
ADX w/o CAM	7.83E−19	5.98E−09	2.60E−12	NA	NA
SBX w/o CAM	1.12E−33	4.92E−25	3.16E−28	3.43E−11	NA
ARX w/o CAM	2.41E−40	9.58E−33	1.59E−35	5.87E−20	0.00017079

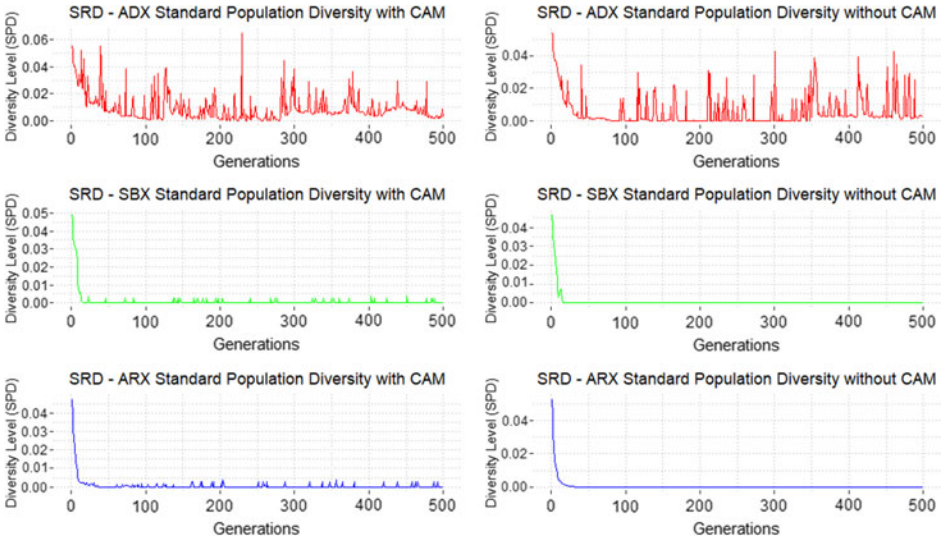


Fig. 23. SPD for each configuration used in SRD. The graphs in red (ADX) achieved the best SPD levels in relation to other crossover methods (SBX and ARX).

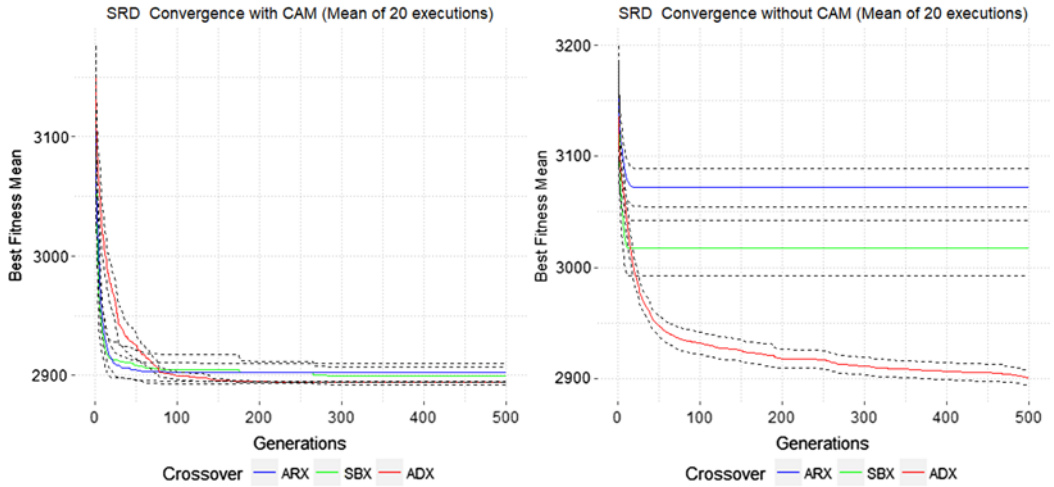


Fig. 24. Convergence graph obtained for the SRD problem with three types of crossover. The graph on the left used CAM, while the graph on the right did not. After applying CAM, ADX obtained better confidence interval and results in relation to SBX and ARX. Without the use of CAM, ADX also achieved better results and a better confidence interval after a few generations, while SBX and ARX stagnated in local optima of the problem.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table XXVII. Mean SPD levels for the configurations tested for TBT.

Crossover	CAM	SPD mean
ADX	**Yes**	**0.109022386**
SBX	Yes	0.011466528
ARX	Yes	0.008174441
ADX	No	0.102900506
SBX	No	0.005544339
ARX	No	0.007183454

The best mean SPD levels with and without CAM were for ADX.

Table XXVIII. Mean results with standard deviations and the best results obtained for three types of crossover for the TBT problem with and without CAM.

Crossover	CAM	Mean fitness	Std. Dev.	Best fitness	X1	X2
ADX	**Yes**	**263.8958436**	**3.77285E−07**	**263.8958434**	**0.788671319**	**0.408259083**
SBX	Yes	264.1438833	0.486676773	263.8958641	0.788840351	0.407781195
ARX	Yes	263.9961037	0.199284018	263.8958434	0.788675419	0.408247487
ADX	No	263.8968527	0.002043782	263.8958434	0.788676232	0.408245186
SBX	No	265.9423265	1.033586631	264.1378635	0.798778025	0.382093203
ARX	No	264.0825426	0.320391861	263.8959226	0.788347079	0.409176962

ADX+CAM achieved the best and most reliable results of all the configurations without restriction violation.

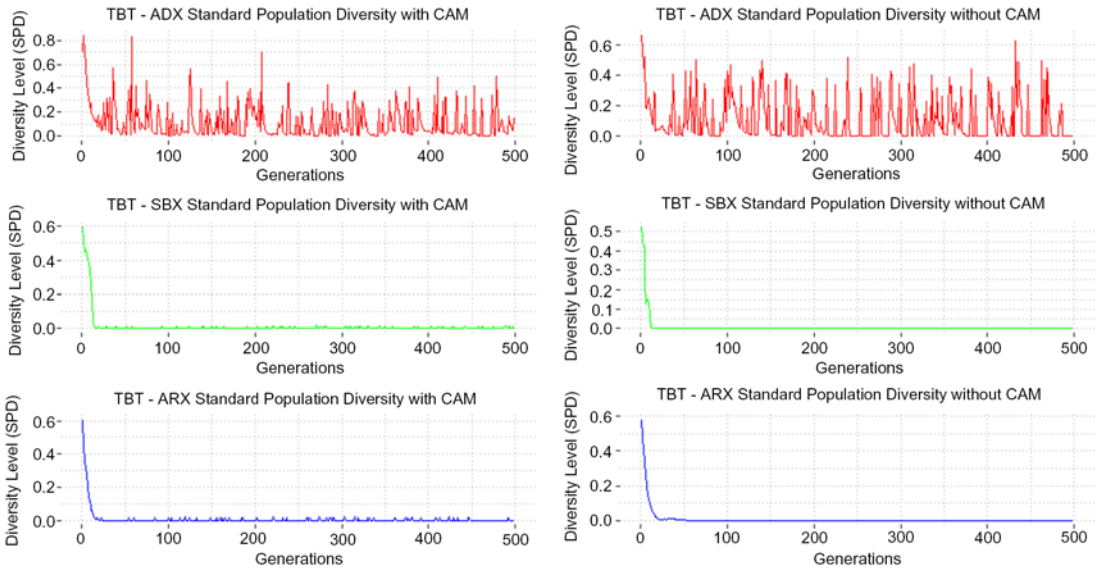


Fig. 25. SPD for each configuration used in TBT. The graphs in red (ADX) achieved the best SPD levels in relation to other crossover methods (SBX and ARX).

Each algorithm performed 20 executions; all the statistical results are shown in Table XXXIII. The modified GA shows superior performance compared with standard PSO and SA, except for the TBT problem where both PSO and GA showed similar results.

To confirm the results, a two-sample t-test with significance level of 0.05 was conducted within the GA (ADX+CAM), PSO, and SA as shown in Table XXXIV. When comparing the results, all the p-values obtained were <0.05, indicating that each type of mean differed from each other, thus confirming that GA had the best performance.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table XXIX. Friedman test (p-values) for the configurations tested for TBT.

	ADX w/ CAM	**SBX w/ CAM**	**ARX w/ CAM**	**ADX w/o CAM**	**SBX w/o CAM**
SBX w/ CAM	6.58E−28	NA	NA	NA	NA
ARX w/ CAM	3.49E−23	0.04635531	NA	NA	NA
ADX w/o CAM	3.66E−08	2.25E−15	2.73E−10	NA	NA
SBX w/o CAM	1.33E−44	1.96E−16	1.48E−21	3.90E−35	NA
ARX w/o CAM	6.58E−28	**1**	0.04635531	2.25E−15	1.96E−16

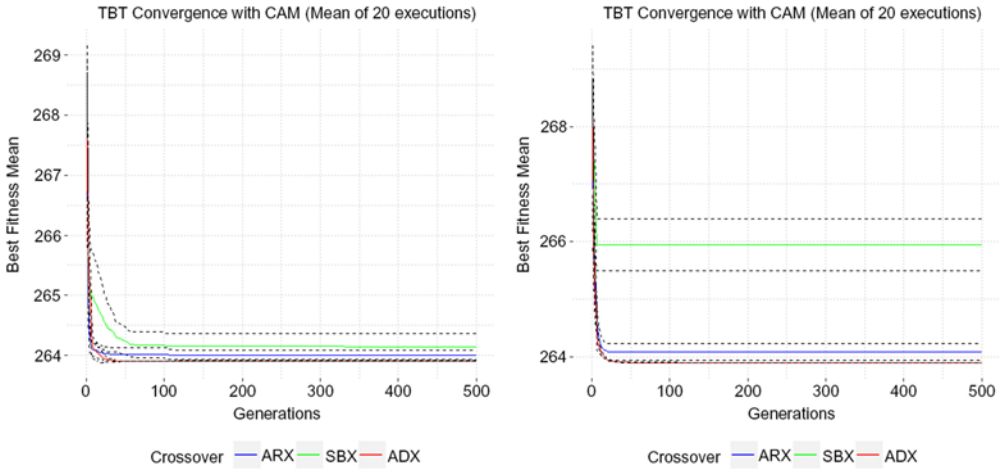


Fig. 26. Convergence graph obtained for the TBT problem with three types of crossover. The graph on the left used CAM, while the graph on the right did not. ADX obtained better results with CAM.

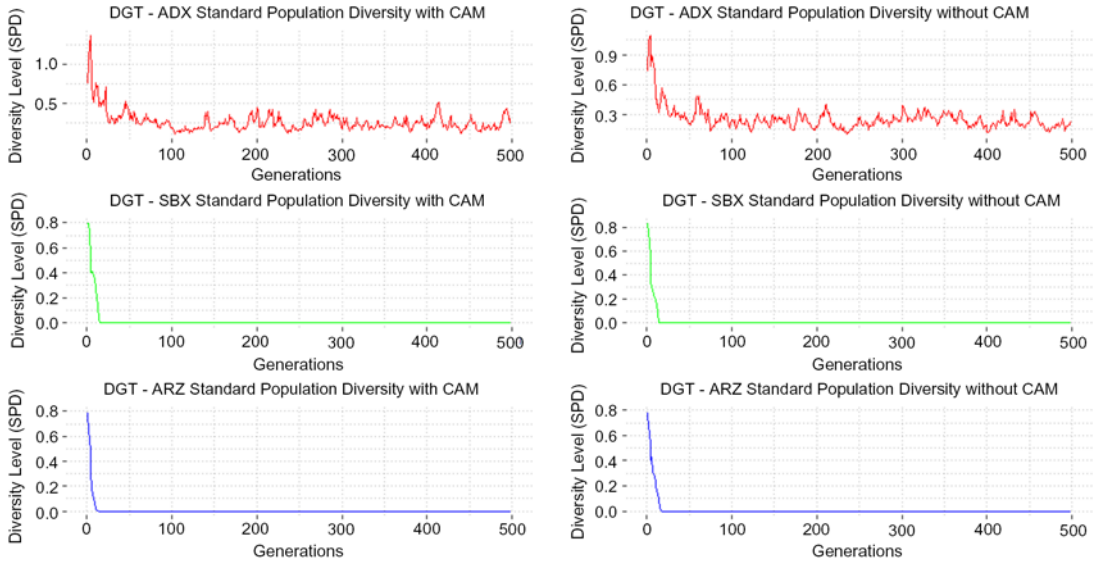


Fig. 27. SPD for each configuration used in DGT. The graphs in red (ADX) achieved the best SPD levels in relation to other crossover methods (SBX and ARX)

Also, the mean convergence curve of 20 executions is shown for each algorithm in Figs. 29 and 30. It is possible to see that the GA (blue line) converged faster than the PSO (red line) and SA (green line) for most of the problems, except for the DGT where the PSO and GA achieved faster convergence versus the SA.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table XXX. SPD mean results for the DGT problem.

Crossover	CAM	SPD mean
ADX	**Yes**	**0.260730253**
SBX	Yes	0.01137375
ARX	Yes	0.006671197
ADX	**No**	**0.250778074**
SBX	No	0.009769822
ARX	No	0.010337964

ADX shows superior performance at creating diverse populations.

Table XXXI. Mean results with standard deviations and the best results obtained for three types of crossover for the DGT problem with and without CAM.

Crossover	CAM	Mean fitness	Std. Dev.	Best fitness	X1	X2	X3	X4
ADX	**Yes**	**1.5085E−10**	**3.01889E−10**	**2.70086E−12**	**19**	**16**	**49**	**43**
SBX	Yes	3.06999E−06	6.848E−06	2.35764E−09	12	30	48	52
ARX	Yes	2.91529E−08	7.23451E−08	2.70086E−12	19	16	49	43
ADX	No	2.46004E−10	4.03594E−10	2.70086E−12	16	19	49	43
SBX	No	2.0689E−06	5.57602E−06	9.92158E−10	24	13	47	46
ARX	No	2.35903E−08	5.66741E−08	2.70086E−12	19	16	49	43

ADX+CAM achieved the best results of all the configurations without restriction violation.

Table XXXII. Friedman test (p-values) for the configurations tested for DGT.

	ADX w/ CAM	SBX w/ CAM	ARX w/ CAM	ADX w/o CAM	SBX w/o CAM
SBX w/ CAM	1.24E−32	NA	NA	NA	NA
ARX w/ CAM	3.31E−11	1.45E−17	NA	NA	NA
ADX w/o CAM	**0.59012357**	1.01E−31	3.54E−10	NA	NA
SBX w/o CAM	1.54E−35	0.246627372	3.76E−21	1.13E−34	NA
ARX w/o CAM	1.22E−13	5.02E−15	0.453916626	1.48E−12	1.16E−18

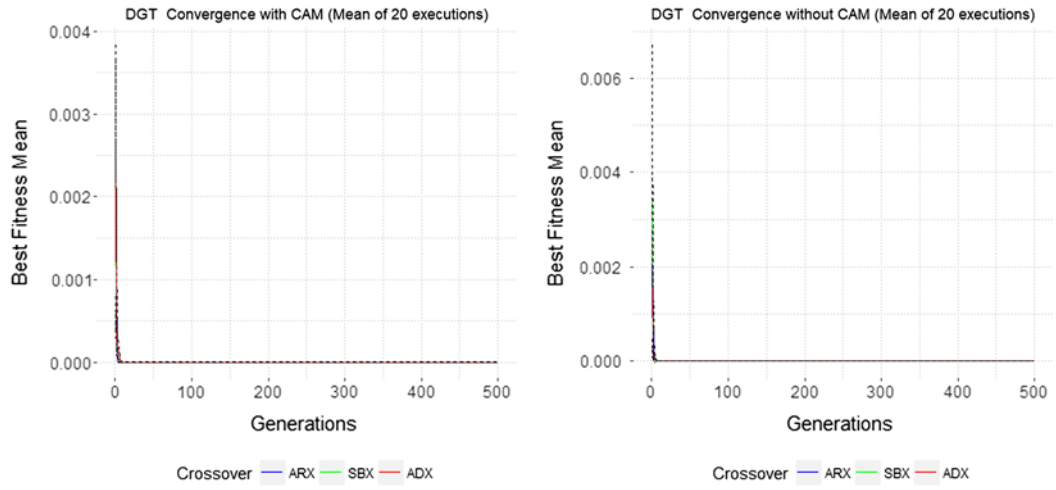


Fig. 28. Convergence plot for the DGT problem; results with CAM are shown on the right side.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table XXXIII. Statistical results for the problems tested.

Problem/algorithm		Best	Median	Mean	Worst	Std. dev.	Variance
DPV1	**GA**	**5804.3762565**	**5804.3764295**	**5804.3767727**	**5804.3831679**	**0.0015110**	**2.28E−06**
	PSO	5959.9438064	6573.7801216	6988.1912812	13756.7604985	1675.9313051	2808745.7395116
	SA	6853.3672792	8153.4648054	8207.8582150	9831.4218680	756.1727858	571797.2820582
DPV2	**GA**	**5885.3329659**	**5885.3331945**	**5885.3331955**	**5885.3334250**	**0.000136126**	**1.85E−08**
	PSO	5972.4022431	6780.3205959	6751.9703029	7524.3494835	369.9394966	136855.2311764
	SA	6922.0566095	8394.2598434	8406.9928462	9867.8898331	829.7893463	688550.3592198
WBD1	**GA**	**2.3809573**	**2.3810040**	**2.3811949**	**2.3822941**	**0.0003854**	**1.49E−07**
	PSO	2.4504108	2.5845788	2.7518595	4.0361274	0.3888061	0.1511702
	SA	2.5189233	2.7189795	2.7218511	2.8752662	0.0974808	0.0095025
WBD2	**GA**	**1.6952472**	**1.6952478**	**1.6952643**	**1.6954760**	**5.26E−05**	**2.77E−09**
	PSO	1.6983537	1.8166167	1.9595216	2.5739594	0.2889512	0.0834928
	SA	1.7916778	1.8814996	1.8795096	1.9565442	0.0409617	0.0016779
MWTCS	**GA**	**0.0126652**	**0.0126653**	**0.0126654**	**0.0126660**	**2.07E−07**	**4.28E−14**
	PSO	0.0126671	0.0136205	0.0136291	0.0159951	7.67E−04	5.89E−07
	SA	0.0128295	0.0131474	0.0131655	0.0138139	2.09E−04	4.37E−08
TBT	**GA**	**263.8958434**	**263.8958435**	**263.8958436**	**263.8958450**	**3.77E−07**	**1.42E−13**
	PSO	**263.8958434**	**263.8958434**	**263.8958437**	**263.8958466**	**9.09E−07**	**8.27E−13**
	SA	263.9150830	263.9679676	263.9857913	264.1543707	0.0676148	0.0045718
DGT	**GA**	**2.70086E−12**	**2.30782E−11**	**1.5085E−10**	**9.92158E−10**	**3.01889E−10**	**9.11368E−20**
	PSO	2.30782E−11	2.35764E−09	3.86926E−05	0.000773602	0.00017298	2.9922E−08
	SA	2.30782E−11	3.43047E−09	2.58858E−08	1.61223E−07	4.5131E−08	2.0368E−15
SRD	**GA**	**2894.3823473**	**2894.3824608**	**2894.3827541**	**2894.3879893**	**1.24E−03**	**1.53E−06**
	PSO	2914.1360462	3010.6239206	3023.0994604	3147.6440018	59.1938779	3503.9151795
	SA	2945.5810119	2966.0943175	2965.7906103	2985.4235394	11.3439139	128.6843820

Table XXXIV. Two-sample t-test assuming equal variances for ADX+CAM for each problem tested.

Problem	PSO	SA
DPV1	1.37E−15	2.34E−89
DPV2	1.64E−59	3.67E−142
WBD1	3.52E−88	1.87E−19
WBD2	1.98E−185	9.55E−35
MWTCS	1.09E−137	2.60E−07
TBT	1.98E−39	0.000220391
DGT	5.66E−11	0.005822203
SRD	0	1.49E−83

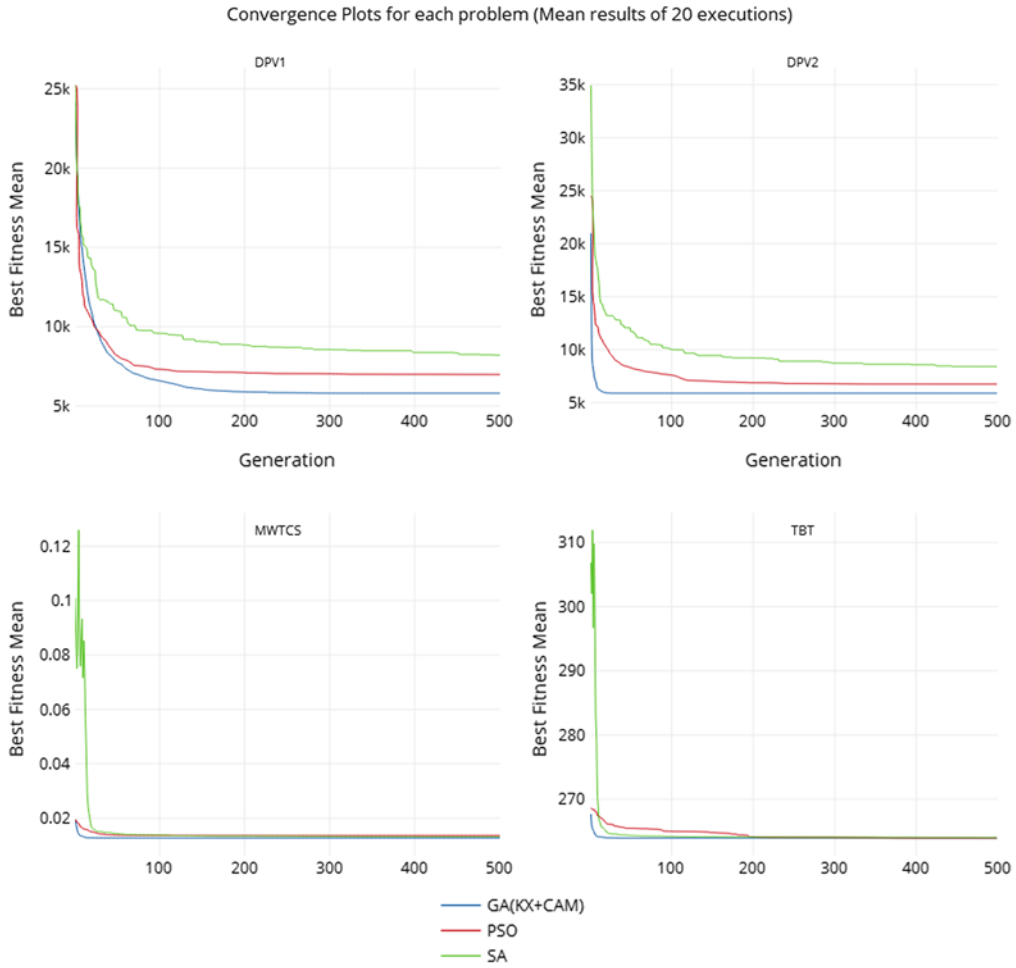


Fig. 29. Convergence plots for DPV1, DPV2, MWTCS, and TBT problems.

10. Engineering Problems Results Comparison

Comparing the best results of ADX+CAM with those of other studies, the proposed algorithm proved to be the best for most of the problems. For the DPV1, DPV2, WBD1, DGT, and TBT problems, ADX+CAM was able to achieve the best and most reliable results in relation to the compared studies as shown in Tables XXXV–XXXVIII and XLI. MWTCS[3] obtained the best result[73] and the best mean value. Given the variability of the results, one study[47] obtained the most reliable results with the lowest standard deviation, as shown in Table XXXIX. On WD2 problem,[4] have better result, but the proposed work shows the best mean and standard deviation (Table XLII). Some studies did not

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table XXXV. Results of DPV1 compared with other studies.

Method	Mean	Std. Dev.	Best Result
Dimmopoulos[45]	NA	NA	5850.3830
Mahdavi et al.[46]	NA	NA	5849.7617
Hedar & Fukushima[47]	6164.585867	257.473670	5868.764836
Gandomi et al.[48]	5937.33790	164.54747	5850.38306
Garg[3]	5805.473914033477	1.411462164114731	5804.448670820886
ADX+CAM	**5804.376773**	**0.001511001**	**5804.376257**

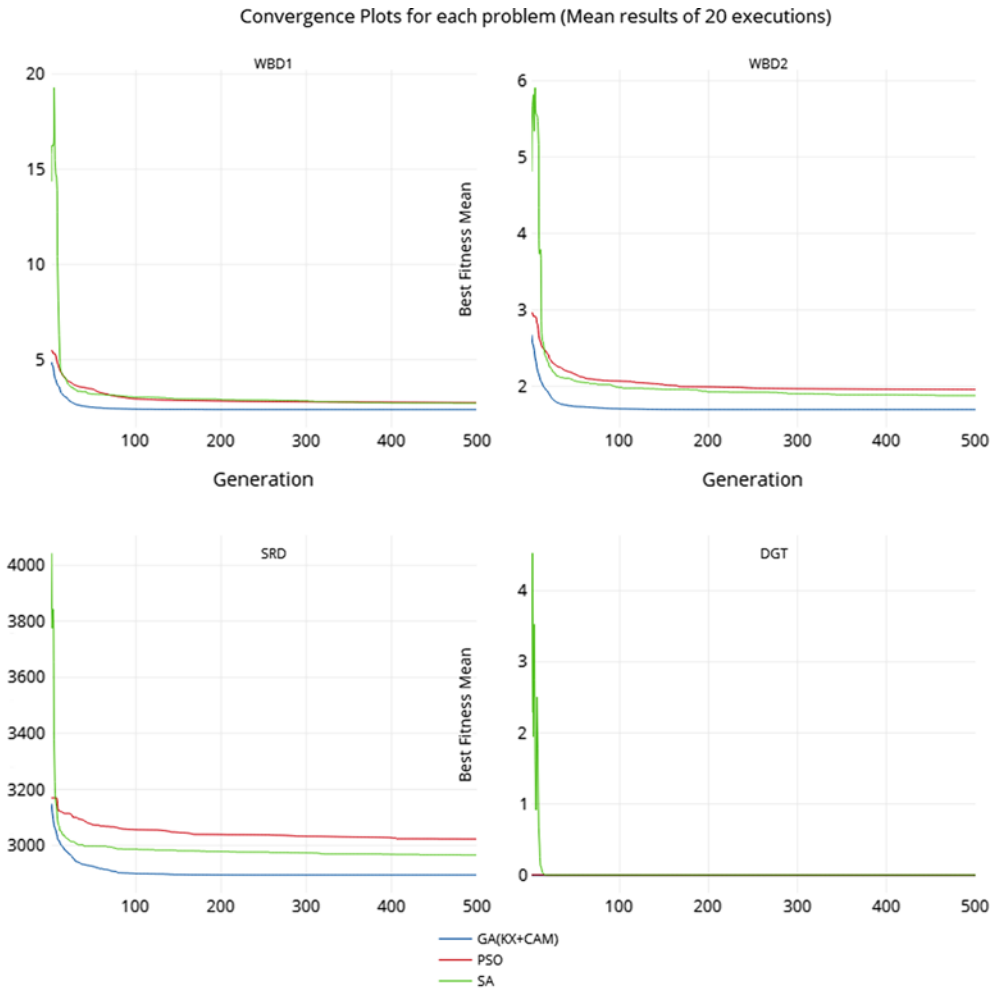


Fig. 30. Convergence plots for WBD1, WBD2, SRD, and DGT problems.

mention the number of function evaluations used to achieve the results; the SRD problem only presented the best results obtained. Considering the works compared, ADX+CAM was able to achieve the best results and had the best mean values for most of the problems. All the results are feasible (did not break any restrictions), except for the ones marked "**."

11. Final Considerations

Problem optimization through evolutionary heuristic algorithms is currently a widely explored field as it provides relatively fast solutions in relation to brute-force and deterministic methods. Of these techniques, GAs are an approach that is widespread in the literature. However, for these algorithms

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table XXXVI. Results of DPV2 compared with other studies.

Method	Mean	Std. Dev.	Best Result
Sandgren[37]	N/A	N/A	8129.1036
Kannan and Kramer[86]	N/A	N/A	7198.0428
Deb and Gene[89]	N/A	N/A	6410.3811
Coello[65]	6293.8432	7.4133	6288.7445
Coello and Montes[67]	6177.2533	130.9297	6059.9463
He and Wang[55]	6147.1332	86.4545	6061.0777
Montes and Coello[70]	6850.0049	426	6059.7456
Kaveh and Talatahari[72]	6081.7812	67.2418	6059.7258
Kaveh and Talatahari[90]	6075.2567	41.6825	6059.0925
Gandomi et al.[88]	6447.736	502.693	6059.714
Cagnina et al.[71]	6092.0498	12.1725	6059.714335
Coello[74]	6440.3786	448.4711	6059.7208
He et al.[69]	6289.92881	305.78	6059.7143
Akay and Karaboga[75]	6245.308144	205	6059.714339
Garg[24]	5887.557024	2.74529	5885.403282
Garg[4]	5885.382053	0.04908	**5885.332773**
ADX+CAM	**5885.333195**	**0.000136126**	5885.332966

Table XXXVII. Results of WBD1 compared with other studies.

Method	Mean	Std. Dev.	Best Result
Ragsdell & Phillips[58]	NA	NA	2.385937
Rao[59]	NA	NA	2.3860
Deb[38]	NA	NA	2.433116
Deb[60]	NA	NA	2.38119
Ray & Liew[61]	3.2551371	0.9590780	2.3854347
Hwang & He[62]**	2.26	NA	2.25
Mehta & Dasgupta[63]	2.3811786	NA	2.381134
Garg[3]	**2.38108932**	1.01227E-4	2.38099617
ADX+CAM	2.381194898	**0.000385388**	**2.38095734**

** = violated restrictions.

Table XXXVIII. Results of WBD2 compared with other studies.

Method	Mean	Std. Dev.	Best Result
Coello[65]	1.771973	0.01122	1.748309
Coello and Montes[67]	1.792654	0.07471	1.728226
Dimopoulos[45]	NA	NA	1.731186
He and Wang[55]	1.748831	0.012926	1.728024
Hedar and Fukushima[47]	1.7564428	0.0424175	1.7250022
Montes et al.[52]	1.725	1.00E−15	1.724852
Montes and Coello[70]	1.81329	0.0705	1.7373
Cagnina et al.[71]	2.0574	0.2154	1.724852
Kaveh and Talatahari[72]	1.729752	0.0092	1.724918
Kaveh and Talatahari[90]	1.727564	0.008254	1.724849
Gandomi et al.[48]	1.878656	0.2677989	1.7312065
Mehta and Dasgupta[63]	1.724865	NA	1.724855
Akay and Karaboga[75]	1.741913	0.031	1.724852
Garg[3]	1.69530842	2.84E−05	1.69526388
Garg4	**1.6952471**	**2.19E−09**	**1.6952471**
ADX+CAM	1.695264346	5.26477E−05	1.695247207

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table XXXIX. Results of MWTCS compared with other studies.

Method	Mean	Std. Dev.	Best Result
Belegundu[39]	NA	NA	0.0128334
Arora[64]	NA	NA	0.0127303
Coello[65]	0.01276920	3.9390E−06	0.01270478
Ray & Saini[66]	0.015526	NA	0.0130600
Coello & Montes[67]	0.012742	5.9000E−06	0.0126810
Ray & Liew[61]	0.012922669	5.92E−05	0.01266924934
Hu et al.[68]	0.012718975	6.446E−06	0.0126661409
He et al.[69]	0.01270233	4.12439E−06	0.0126652812
He & Wang[55]	0.012730	5.1985E−06	0.0126747
Zhang et al.[41]	0.012669366	1.25E−06	0.012665233
Hedar & Fukushima[47]	0.012665299	**2.2E−09**	0.012665285
Montes et al.[52]	0.012666	2.0E−07	0.012665
Montes & Coello[70]	0.013461	9.6600E−05	0.012698
Cagnina et al.[71]	0.0131	4.1E−05	0.012665
Kaveh & Talatahari[72]	0.012720	3.4888E−06	0.0126432
Omran & Salman[73]	**0.0126652642**	NA	0.0126652375
Coelho[74]	0.013524	0.001268	0.012665
Akay & Karaboga[75]	0.012709	0.012813	0.012665
Mohamed. A[49]	0.012668899	5.38E−06	0.012665232
Garg[3]	0.0126689724845	9.429426E−07	**0.0126652327883**
ADX+CAM	0.012665402	2.06789E−07	0.012665234

Table XL. Results of SRD compared with other studies.

Method	Mean	Std. Dev.	Best Result
Akhtar et al.[76]	NA	NA	3008.197440
Rao & Xiong[77]	NA	NA	3000.959715
Cagnina et al.[71]	NA	NA	2996.347849
Jaberipour & Khorram[78]	NA	NA	2994.477531
Li & Papalambros[79]	NA	NA	2994.553869
Tosserams et al.[80]	NA	NA	2996.645783
Lu & Kim[81]	NA	NA	3019.583365
Lin et al.[82]	NA	NA	2993.738921
Huang[83]	NA	NA	2990.124384
Lin et al.[84]	NA	NA	2994.471921
ADX+CAM	**2894.382754**	**0.00123656**	**2894.382347**

to perform well, they need to balance the diversity of the population and the capacity for finding solutions to the problem.[17]

This study presented ADX and CAM as techniques to increase optimization power and population diversity, respectively, preventing the algorithm from stagnating in local optima of the search space, significantly potentiating the results in relation to other known crossover methods. CAM allows for increasing population diversity by reallocating the individuals based on a correlation matrix calculated between all the genes of the population. The ADX increases the exploitation of already known solutions through adjusting the genes of two individuals based on a gradient vector during the crossover phase.

By testing ADX+CAM on the CEC'2015 problems and comparing the top five contestants, the proposed algorithm was able to provide better results for some problems than the humanCog.[34] The population had better exploration of the search space when using the CAM operator compared with the tests without the CAM phase. As for the engineering problems, the algorithm had better performance and acceptable population diversity levels and achieved better results in comparison to other works in the literature.

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

Table XLI. Results of TBT compared with other studies.

Method	Mean	Std. Dev.	Best Result
Mohamed. A[49]	263.895843	0.00E+00	263.8958434
Mohamed & Sabry[50]	263.895843	5.34E−13	263.8958434
Wang & Li[51]	263.895843	4.34E−14	263.8958434
Zhang et al.[41]	263.895844	9.70E−07	263.8958433
Wang et al.[85]	263.90041	1.10E−03	263.8958435
Melo & Carosio[53]	263.895843	2.58E−07	263.8958434
Gong et al.[54]	263.895843	0.00E+00	263.8958434
Sadollah et al.[56]	263.897996	3.93E−03	263.895852
Eskandar et al.[57]	263.895903	8.71E−05	263.895843
ADX+CAM	**263.8958436**	**3.77285E−07**	**263.8958434**

Table XLII. Results of DGT compared with other studies.

Method	Mean	Std. Dev.	Best Result
Sandgren[37]	NA	NA	5.712E−6
Kannan and Kramer[86]	NA	NA	2.146E−8
Deb and Goyal[87]	NA	NA	2.701E−12
Gandomi et al.[88]	1.9841E−09	3.5546E−09	2.701E−12
Garg[4]	1.2149E−09	8.7787E−10	**2.70085E−12**
ADX+CAM	**1.5085E−10**	**3.01889E−10**	2.70086E−12

Lastly, we performed an analysis comparing CAM and ADX with other types of evolutionary heuristics and algorithms. From our results, it is evident that CAM+ADX has a better potential for competing in problem optimization with other compared studies.

References

1. Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs* (Springer-Verlag, Berlin, 1994).
2. X. Yu and M. Gen, *Introduction to Evolutionary Algorithms* (Springer-Verlag, London, 2010).
3. H. Garg, "Solving structural engineering design optimization problems using an artificial bee colony algorithm," *J. Ind. Manage. Optim.* **10**(3), 777–794 (2014).
4. H. Garg, "A hybrid PSO-GA algorithm for constrained optimization problems," *Appl. Math. Comput.* **274**, 292–305 (2016).
5. A. Cheraghalipour, M. Hajiaghaei-Keshteli and M. Mahdi Paydar, "Tree Growth Algorithm (TGA): A novel approach for solving optimization problems," *Eng. Appl. Artif. Intell.* **72**, 393–414 (2018).
6. J. Zhang, M. Xiao, L. Gao and Q. Pan, "Queuing search algorithm: A novel metaheuristic algorithm for solving engineering optimization problems," *Appl. Math. Modell.* **63**, 464–490 (2018).
7. R. S. Patwal, N. Narang and H. Garg, "A novel TVAC-PSO based mutation strategies algorithm for generation scheduling of pumped storage hydrothermal system incorporating solar units," *Energy* **142**, 822–837 (2018).
8. D. Rani, T. R. Gulati and H. Garg, "Multi-objective non-linear programming problem in intuitionistic fuzzy environment: Optimistic and pessimistic view point," *Expert Syst. Appl.* **64**, 228–238 (2016).
9. H. Garg, "An efficient biogeography-based optimization algorithm for solving reliability optimization problems," *Swarm Evol. Comput.* **24**, 1–10 (2015).
10. H. Garg and S. P. Sharma, "Multi-objective reliability-redundancy allocation problem using particle swarm optimization," *Comput. Ind. Eng.* **64**(1), 247–255 (2013).
11. H. Garg, "Performance analysis of an industrial system using soft computing based hybridized technique," *J. Braz. Soc. Mech. Sci. Eng.* **39**, 1441–1451 (2016).
12. T. T. Hills, P. M. Todd, D. Lazer, A. D. Redish and I. D. Couzin, "Exploration versus exploitation in space, mind, and society," *Trends Cognit. Sci.* **19**(1), 46–54 (2014).
13. M. Crepinsek, S. H. Liu and M. Mernik, "Exploration and exploitation in evolutionary algorithms: A survey," *ACM Comput. Surv.* **45**(3), 33–35 (2013).
14. H. M. Pandeya, A. Chaudharyb and D. Mehrotrac, "A comparative review of approaches to prevent premature convergence in GA," *Appl. Soft Comput.* **24**, 1047–1077 (2014).
15. Y.-Y. Wong, K.-H. Lee, K.-S. Leung and C.-W. Ho, "A novel approach in parameter adaptation and diversity maintenance for genetic algorithms," *Soft Comput.* **7**, 506–515 (2003).

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

16. D. Curran, C. O'Riordan and H. Sorensen, "The effects of lifetime learning on the diversity and fitness of populations," *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, London, UK (2007) p. 337.
17. B. McGinley, J. Maher, C. O'Riordan and F. Morgan, "Maintaining healthy population diversity using adaptive crossover, mutation, and selection," *IEEE Trans. Evol. Comput.* **15**(5), 692–714 (2011).
18. L. J. Eshelman and J. D. Schaffer, "Real-coded genetic algorithms and interval-schemata," *Proceedings of the Workshop on Foundations of Genetic Algorithms*, Vail, Colorado, USA (1992) pp. 187–202.
19. K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Syst.* **9**, 115–148 (1995).
20. A. Cervantes-Castilho, E. Mezura-Montes and C. Coello, "An empirical comparison of two crossover operators in real-coded genetic algorithms for constrained numerical optimization problems," *Special Session on Evolutionary Computation (ROPEC)*, Ixtapa, Mexico (2014).
21. Y. C. Chuang, C. T. Chen and C. Hwang, "A real-coded genetic algorithm with direction-based crossover operator," *Inf. Sci.* **305**, 320–348 (2015).
22. M. Li, L. Qianting, M. Meiqiong and L. Sicong, "Optimization and application of single point crossover and multi offspring genetic algorithm," *Int. J. Hybrid Inf. Technol.* **9**(1), 1–8 (2016).
23. Q. Zhu, Q. Lin, Z. Du, Z. Liang, W. Wang, Z. Zhu, J. Chen, P. Huang and Z. Ming, "A novel adaptive hybrid crossover operator for multi objective evolutionary algorithm," *Inf. Sci.* **345**, 177–198 (2016).
24. Y.-C. Chuanga, C.-T. Chena and C. Hwangba, "A simple and efficient real-coded genetic algorithm for constrained optimization," *Appl. Soft Comput.* **38**, 87–105 (2016).
25. Z.-Q. Chen and Y.-F. Yin, "A new crossover operator for real-coded genetic algorithm with selective breeding based on difference between individuals," *8th International Conference on Natural Computation (ICNC)*, Chongqing, Sichuan, China (2012).
26. Y. Kaya, M. Uyar and R. Tekin, "A novel crossover for genetic algorithm: Ring crossover," *CoRR* http://arxiv.org/abs/1105.0355 (2011).
27. S. H. Ling, "Iterated function system-based crossover operation for real-coded genetic algorithm," *J. Intell. Learn. Syst. Appl.* **7**, 37–41 (2015).
28. M. G. Kendall, *Rank Correlation Methods* (Hafner Publishing Co., New York, 1955).
29. H. Abdi, "The Kendall rank correlation coefficient," In: *Encyclopedia of Measurement and Statistics* (N. Salkind, ed.) (SAGE Publications, Inc., Thousand Oaks, CA, USA, 2007).
30. Q. Chen, B. Liu, Q. Zhang, J. J. Liang, P. N. Suganthan and B. Y. Qu, "Problem definitions and evaluation criteria for CEC 2015 special session on bound constrained single-objective computationally expensive numerical optimization," *IEEE Conference on Evolutionary Computation*, Sendai, Japan (2015).
31. J. L. Rueda and I. Erlich, "MVMO for bound constrained single-objective computationally expensive numerical optimization," *2015 IEEE Congress on evolutionary Computation*, Sendai, Japan (2015) pp. 1011–1017.
32. M. Andersson, S. Bandaru, A. Ng and A. Syberfeldt, "Parameter tuned CMA-ES on the CEC'15 expensive problems," *2015 IEEE Congress on Evolutionary Computation*, Sendai, Japan (2015) pp. 1950–1957.
33. M. R. Tanweer, S. Suresh and N. Sundararajan, "Improved srpso algorithm for solving cec 2015 computationally expensive numerical optimization problems," *2015 IEEE Congress on Evolutionary Computation*, Sendai, Japan (2015) pp. 1943–1949.
34. A. Al-Dujaili, K. Subramanian and S. Suresh, "Humancog: A cognitive architecture for solving optimization problems," *2015 IEEE Congress on Evolutionary Computation*, Sendai, Japan (2015) pp. 3220–3227.
35. S. Momani, Z. Abo-Hammour and O. Alsmadi, "Solution of inverse kinematics problem using genetic algorithms," *Appl. Math. Inf. Sci.* **10**, 1–9 (2015).
36. Z. S. Abo-Hammour, O. M. K. Alsmadi, S. I. Bataineh, M. A. Al-Omari and N. Affach, "Continuous genetic algorithms for collision-free cartesian path planning of robot manipulators," *Int. J. Adv. Rob. Syst.* **8**(6), 14–36 (2011).
37. E. Sandgren, "Nonlinear integer and discrete programming in mechanical design," *Proceeding of the ASME Design Technology Conference*, Kissimmee, Florida (1988), pp. 95–105.
38. K. Deb, "Optimal design of a welded beam via genetic algorithms," *AIAA J.* **29**(11), 2013–2015 (1991).
39. A. D. Belegundu, *A Study of Mathematical Programming Methods for Structural Optimization* (Department of Civil and Environmental Engineering, University of Iowa, Iowa City, Iowa, 1982).
40. J. Golinski, "Optimal synthesis problems solved by means of nonlinear programming and random methods," *J. Mech.* **5**, 287–309 (1970).
41. M. Zhang, W. Luo and X. Wang, "Differential evolution with dynamic stochastic selection for constrained optimization," *Inf. Sci.* **178**, 3043–3074 (2008).
42. Z. Michalewicz and M. Schoenauer, "Evolutionary algorithms for constrained parameter optimization problems," *Evol. Comput.* **4**(1), 1–32 (1996).
43. S. A. Amanchi, *Applied Nonparametric Statistical Tests to Compare Evolutionary and Swarm Intelligence Approaches* (Master of Science, Department of Computer Science, North Dakota State University, Fargo, North Dakota, USA, 2014).
44. Y. Zhao, Y. Cai1 and D. Cheng, "A novel local exploitation scheme for conditionally breeding real-coded genetic algorithm," *Multimedia Tools Appl.* **76**, 17955–17969 (2017).
45. G. G. Dimopoulos, "Mixed-variable engineering optimization based on evolutionary and social metaphors," *Comput. Methods Appl. Mech. Eng.* **196**, 803–817 (2007).

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

46. M. Mahdavi, M. Fesanghary and E. Damangir, "An improved harmony search algorithm for solving optimization problems," *Appl. Math. Comput.* **188**, 1567–1579 (2007).
47. A. R. Hedar and M. Fukushima, "Derivative – free filter simulated annealing method for constrained continuous global optimization," *J. Global Optim.* **35**, 521–549 (2006).
48. A. H. Gandomi, X. S. Yang and A. H. Alavi, "Mixed variable structural optimization using firefly algorithm," *Comput. Struct.* **89**, 2325–2336 (2011).
49. A.W. Mohamed, "A novel differential evolution algorithm for solving constrained engineering optimization problems," *J. Intell. Manuf.* **29**(3), 659–692 (2017).
50. A.W. Mohamed and H. Z. Sabry, "Constrained optimization based on modified differential evolution algorithm," *Inf. Sci.* **194**, 171–208 (2012).
51. L. Wang and L. Li, "An effective differential evolution with level comparison for constrained engineering design," *Struct. Multi. Optim.* **41**, 947–963 (2010).
52. E. Mezura-Montes, C. A. C. Coello, J. Velázquez-Reyes, and L. MuñozDávila, "Multiple trial vectors in differential evolution for engineering design," *Eng. Optim.* **39**(5), 567–589 (2007).
53. V. V. Melo and G. L. Carosio, "Investigating multi-view differential evolution for solving constrained engineering design problems," *Expert Syst. Appl.* **40**(9), 3370–3377 (2013).
54. W. Gong, Z. Cai and D. Liang, "Engineering optimization by means of an improved constrained differential evolution," *Comput. Methods Appl. Mech. Eng.* **268**, 884–904 (2014).
55. Q. He and L. Wang, "An effective co-evolutionary particle swarm optimization for constrained engineering design problems," *Eng. Appl. Artif. Intell.* **20**, 89–99 (2007).
56. A. Sadollah, A. Bahreininejada, H. Eskandar, and M. Hamdi, "Mine blast algorithm: A new population-based algorithm for solving constrained engineering optimization problems," *Appl. Soft Comput.* **13**, 2592–2612 (2013).
57. H. Eskandar, A. Sadollah, A. Bahreininejad and M. Hamdi, "Water cycle algorithm – A novel meta-heuristic optimization method for solving constrained engineering optimization problems," *Comput. Struct.* **110–111**, 151–166 (2012).
58. K. M. Ragsdell and D. T. Phillips, "Optimal design of a class of welded structures using geometric programming," *ASME J. Eng. Ind.* **98**, 1021–1025 (1976).
59. S. S. Rao, *Engineering Optimization: Theory and Practice*, 3rd edition (John Wiley & Sons, Chichester, 1996).
60. K. Deb, "An efficient constraint handling method for genetic algorithms," *Comput. Methods Appl. Mech. Eng.* **186**, 311–338 (2000).
61. T. Ray and K. M. Liew, "Society and civilization: An optimization algorithm based on the simulation of social behavior," *IEEE Trans. Evol. Comput.* **7**, 386–396 (2003).
62. S. F. Hwang and R. S. He, "A hybrid real-parameter genetic algorithm for function optimization," *Adv. Eng. Inf.* **20**, 7–21 (2006).
63. V. K. Mehta and B. Dasgupta, "A constrained optimization algorithm based on the simplex search method," *Eng. Optim.* **44**, 537–550(2012).
64. J. S. Arora, *Introduction to Optimum Design* (McGraw-Hill, New York, 1989).
65. C. A. C. Coello, "Use of a self-adaptive penalty approach for engineering optimization problems," *Comput. Ind.* **41**, 113–127 (2000).
66. T. Ray and P. Saini, "Engineering design optimization using a swarm with an intelligent information sharing among individuals," *Eng. Optim.* **33**, 735–748 (2001).
67. C. A. C. Coello and E. M. Montes, "Constraint-handling in genetic algorithms through the use of dominance-based tournament selection," *Adv. Eng. Inf.* **16**, 193–203 (2002).
68. X. H. Hu, R. C. Eberhart and Y. H. Shi, "Engineering optimization with particle swarm," *Proceedings of the 2003 IEEE Swarm Intelligence Symposium*, Indianapolis, Indiana, USA (2003) pp. 53–57.
69. S. He, E. Prempain and Q. H. Wu, "An improved particle swarm optimizer for mechanical design optimization problems," *Eng. Optim.* **36**, 585–605 (2004).
70. E. M. Montes and C. A. C. Coello, "An empirical study about the usefulness of evolution strategies to solve constrained optimization problems," *Int. J. Gen. Syst.* **37**, 443–473 (2008).
71. L. C. Cagnina, S. C. Esquivel and C. A. C. Coello, "Solving engineering optimization problem with the simple constrained particle swarm optimizer," *Informatica* **32**, 319–326 (2008).
72. A. Kaveh and S. Talatahari, "An improved ant colony optimization for constrained engineering design problems," *Eng. Comput.* **27,** 155–182 (2010).
73. M. G. H. Omran and A. Salman, "Constrained optimization using CODEQ," *Chaos, Solitons Fractals* **42**, 662–668 (2009).
74. L. S. Coelho, "Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems," *Expert Syst. Appl.* **37**, 1676–1683 (2010).
75. B. Akay and D. Karaboga, "Artificial bee colony algorithm for large-scale problems and engineering design optimization," *J. Intell. Manuf.* **23**, 1001–1014 (2012).
76. S. Akhtar, K. Tai and T. Ray, "A socio-behavioral simulation model for engineering design optimization," *Eng. Optim.* **34**(4), 341–354 (2002).
77. S. S. Rao and Y. Xiong, "A hybrid genetic algorithm for mixed discrete design optimization," *J. Mech. Des.* **127**(6), 1100–1112 (2005).
78. M. Jaberipour and E. Khorram, "Two improved harmony search algorithms for solving engineering optimization problems," *Commun. Nonlinear Sci. Numer. Simul.* **15**(11), 3316–3331 (2010).

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press

79. H. L. Li and P. Papalambros, "A production system for use of global optimization knowledge," *J. Mech. Transm. Autom. Des.* **107**(2), 277–284 (1985).
80. S. Tosserams, L. F. P. Etman and J. E. Rooda, "An augmented Lagrangian decomposition method for quasi-separable problems in MDO," *Struct. Multidiscip. Optim.* **34**(3), 211–227 (2007).
81. S. Lu and H. M. Kim, "A regularized inexact penalty decomposition algorithm for multidisciplinary design optimization problems with complementarity constraints," *J. Mech. Des.* **132**(4), 12, Art. ID 041005 (2010).
82. M.-H. Lin, J.-F. Tsai and P.-C. Wang, "Solving engineering optimization problems by a deterministic global optimization approach," *Appl. Math. Inf. Sci.* **6**(3), 1101–1107 (2012).
83. C. H. Huang, "Engineering design by geometric programming," *Math. Prob. Eng.* **2013**, 8, Art. ID 568098 (2013).
84. M.-H. Lin, J.-F. Tsai, N.-Z. Hu and S.-C. Chang, "Design optimization of a speed reducer using deterministic techniques," In: *Mathematical Problems in Engineering* (Hindawi Publishing Corporation, London, UK, 2013).
85. Y. Wang, Z. Cai and Y. Zhou, "Accelerating adaptive trade-off model using shrinking space technique for constrained evolutionary optimization," *Int. Method Numer. Methods Eng.* **77**(11), 1501–1534 (2009).
86. B. K. Kannan and S. N. Kramer, "An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design," *Trans. ASME J. Mech. Des.* **116**, 318–320 (1994).
87. K. Deb and M. Goyal, "A combined genetic adaptive search (GeneAS) for engineering design," *Comput. Sci. Inf.* **26**(4), 30–45 (1996).
88. A. Gandomi, X. S. Yang and A. Alavi, "Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems," *Eng. Comput.* **29**(1), 17–35 (2013).
89. K. Deb and A. S. Gene, "A robust optimal design technique for mechanical component design," In: *Evolutionary Algorithms in Engineering Applications* (D. Dasgupta and Z. Michalewicz eds.) (Springer, Berlin, 1997) pp. 497–514.
90. A. Kaveh and S. Talatahari, "Engineering optimization with hybrid particle swarm and ant colony optimization," *Asian J. Civil Eng. (Build. Hous.)* **10**, 611–628 (2009).

https://doi.org/10.1017/S026357471900016X Published online by Cambridge University Press