Published version DOI: 10.1109/SMC53992.2023.10394087

Efficient Time-Delay System Optimization with Auto-Configured Metaheuristics

Roman Senkerik[1], Aleksander Byrski[3], Hubert Guzowski[3], Peter Janku[1], Tomas Kadavy[1], Zuzana Kominkova Oplatkova[1], Radek Matusu[2], Michal Pluhacek[1], Libor Pekar[2], Maciej Smolka[3] and Adam Viktorin[1]

***Abstract*— This paper presents an experimental study that compares the performance of four selected metaheuristic algorithms for optimizing a time delay system model. Time delay system models are complex and challenging to optimize due to their inherent characteristics, such as non-linearity, multimodality, and constraints. The study includes an explanation of the choice and core functionality of the selected algorithms, which are both baseline and state-of-the-art variants of self-organizing migrating algorithm (SOMA), state-of-the-art variant from the Success-History-based Adaptive Differential Evolution family of algorithms, with emphasis on diverse search (DISH algorithm), and Covariance Matrix Adaptation Evolution Strategy (CMA-ES) algorithm. The hyperparameters of the metaheuristic algorithms were set using the iRace automatic algorithm configuration framework. The paper emphasizes the importance of metaheuristic algorithms in control engineering for time-delay systems to develop more effective and efficient control strategies and precise model identifications. The experimental results highlight the effectiveness of the state-of-the-art algorithms with specific adaptive mechanisms like population organization process, diverse search and adaptation mechanisms ensuring a gradual transition from exploration to exploitation. Overall, this study contributes to understanding the challenges and advantages of using metaheuristic algorithms in control engineering for time delay systems. The results provide valuable insights into the performance of modern metaheuristic algorithms and can help guide the selection of appropriate adaptive mechanisms of metaheuristics.**

I. Introduction

In numerous research disciplines and industry sectors of today, a wide range of real-world challenges involves continuous single-objective optimization problems. Metaheuristic algorithms [1] have emerged as a promising tool for optimizing the complex problems due to their ability to handle high-dimensional optimization problems and their robustness to noise and uncertainty.

Control engineering is a critical field vital in ensuring the efficient operation of complex systems. However, various quantities do not act simultaneously. The latency between some action and its impact can appear [2]. Optimizing either the control or identification of a complex system with time delay is challenging due to numerous interdependent variables and constraints. A time delay in the system response further complicates the problem since the system is infinite-dimensional because of an infinite number of its modes [3].

In recent years, various metaheuristic algorithms, such as particle swarm optimization (PSO), artificial bee colony (ABC), genetic algorithms (GA), differential evolution (DE), and hybrid versions with other metaheuristic algorithms have been proposed for optimizing the control of a complex system with time delay [4]–[6]. Despite disadvantages associated with using these algorithms, like black-box nature, scalability, and the need for hyperparameters tuning, using metaheuristic algorithms in control engineering for time-delay systems offers a promising approach for developing more effective and efficient control strategies and precise model identifications. By leveraging the strengths of these algorithms, researchers and practitioners can develop more robust, scalable, and effective control systems.

The paper's organization is the following: After the research motivation and state-of-the-art section, selected variants of metaheuristic algorithms are introduced, followed by a description of the solved problem of optimization of time-delay system model identification. Experiments setup, a summary of the results, and a discussion follow afterward.

A. Motivation and Originality

Solving parametric optimization problems for time-delay systems usually brings several challenges. Wrongly selected fitting points (frequencies) can cause very low sensitivity to the changes in optimized model parameters and obtained results did not satisfy quality validation. Another major issue affecting the outcome is the nature of the time-delay system itself. If feedback loops inside the process include delays (i.e., the so-called state or internal delays), the system is infinite-dimensional because of an infinite number of its modes. Only a limited set of model parameters determines its properties driven by an infinite set of model-free response components. For most time-delay systems, only a subset

The research presented in this paper was partially supported by: NCN project no: 2020/39/I/ST7/02285, Polish Ministry of Education and Science funds assigned to AGH University of Science and Technology. It was also supported by Czech Science Foundation (GACR) project no: GF21-45465L, the Internal Grant Agency of the Tomas Bata University in Zlin, under project number IGA/CebiaTech/2023/004, and resources of A.I.Lab at the Faculty of Applied Informatics, Tomas Bata University in Zlin (ailab.fai.utb.cz).

[1]A.I.Lab, Department of Informatics and Artificial Intelligence, Faculty of Applied Informatics, Tomas Bata University in Zlin, Czech Republic {senkerik,oplatkova}@utb.cz

[2] Department of Automation and Control Engineering, Faculty of Applied Informatics, Tomas Bata University in Zlin, Czech Republic {pekar,rmatusu}@utb.cz

[3]Institute of Computer Science, AGH University of Science and Technology, Krakow, Poland {guzowski,smolka,olekb}@agh.edu.pl

of the so-called dominant modes has a decisive impact on system features in the time domain and the frequency domain [3]. This poses a challenge for metaheuristics, as there is a need to search a multimodal constrained space, and moreover, certain optimized parameters may be very close to the domain borders. All this imposes demands on the metaheuristic algorithm's selection, configuration, and core/adaptive internal mechanisms.

To better understand the effectiveness of metaheuristic algorithms in optimizing the parameters of a model with time delay clearly defines the motivation for this paper. This research paper investigates the performance comparison of selected metaheuristics to identify the most effective internal mechanisms influencing population behavior for a real-world constrained optimization problem, and then use the gained knowledge in further follow-up research.

B. Related works

Due to the rapid development of new algorithms and applications of not always correctly configured metaheuristics, criticism began to occur [7]–[9]. These studies have highlighted the need to move beyond creating new algorithms and focus on understanding the function and taxonomy of existing metaheuristic algorithms.

Following this trend, the previous research with the same optimization problem modified and supplemented the chosen metaheuristic algorithms with mechanisms supporting exploratory behavior and knowledge sharing [10], [11]. The results demonstrated effectiveness compared to classical evolutionary algorithms, such as genetic algorithms (GA). Another study with the same optimization problem [12] then attempted to identify more closely the problem of choosing the boundaries of the search space and the fact that some of the search parameters may lie very close to the domain borders. Another observation was that the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) algorithm [13] could handle the optimization problem better than the classical GA, and the Nelder-Mead [14] optimization method as a baseline technique. However, the study [12] focused on the near-domain borders search issue, the discrepancy between the seemingly low (suitable) fitness value, and the resulting inferior system identification and stability in the frequency domain. It did not focus on hyperparameters tuning and performance comparisons of several selected metaheuristic algorithms.

All these facts were behind the experiment setup, the selection of metaheuristic algorithms, and the search for their best configurations. We have selected the self-organizing migrating algorithm (SOMA) [15], [16] since it has gained renewed interest from the research community due to its adaptive nature, exploration capabilities, and ability to solve complex problems. SOMA is a population-based metaheuristic algorithm that modifies traditional crossover and mutation operations to simulate a social group of individuals. Recently, many powerful modern versions have been introduced [16]. We have chosen the original version as the baseline method and one state-of-the-art variant SOMA Team to Team Adaptive (T3A). The T3A variant was introduced in 2019 [17] and was tested in the CEC 2019 100 digits competition, where it achieved 4th place. Another algorithm belongs to the Success-History-based Adaptive Differential Evolution (L-SHADE) family [18]. We have selected the DISH [19] variant, which includes the distance-based approach, keeping the exploration ability and higher population diversity. The modern approaches considering the distance of solutions in the search space should lead to avoidance of premature convergence in higher dimensional objective spaces [20]. Finally, the last algorithm chosen was CMA-ES [13] as another baseline technique. CMA-ES is a widely-used, advanced optimization algorithm that excels in continuous, non-linear, and high-dimensional optimization problems. It has become popular among researchers and practitioners due to its robustness, efficiency, and adaptability.

II. Metaheuristic Algorithms

The selected variants of the metaheuristic algorithms are described in the following subsections. The hyperparameters of the algorithms were set using the iRace autoconfiguration framework [21] and are listed in the "experimental results" section IV. Why these algorithms were chosen is explained in the previous "related works" section I-B.

A. Generic SOMA Algorithm

The algorithm follows a simple process in the basic variant of SOMA, called All-To-One. At the start of each iteration, known as the migration loop, the fittest individual is selected as the leader. The remaining individuals then move towards the leader in the search space, taking jumps determined by the *step* parameter until they reach the final position given by the *pathLength* parameter. Each step is evaluated using the fitness function, and the best position, including the initial position of the individual, is selected as the new position of the individual in the next migration loop. The All-To-One variant of SOMA has been described in a recent book and survey [15], [16]. The exact position of each step is calculated according to (1).

$$x_{i,j}^{k+1} = x_{i,j}^{k} + (x_{L,j}^{k} - x_{i,j}^{k}) \cdot t \cdot PRTVector_j \qquad (1)$$

Where $x_{i,j}^{k+1}$ is the new position of *i*-th solution (for iteration *k+1*) for dimension *j*, and $x_{i,j}$ is the current position of the *i*-th solution. The $x_{L,j}$ represents a position of a leader (the leader selection depends on the used SOMA strategy). Parameter *t* represents steps from *i*-th solution to the leader. Solution *i* is migrating, by discrete steps, and the best-found solution on *t*-th position is propagated into a new iteration of the algorithm. The *t* parameter is generated in a range starting from 0 to *pathLength* with step size *step*.

The $PRTVector_j$ represents an important mechanism in SOMA. It is generated for each new *t* step. This vector determines which dimensions will be changed in a particular step *t*. In other words, in which dimensions the solution will "head" ("be perturbed") towards the *leader* position or not. Since the SOMA algorithm was developed in the

context of the control optimization challenge, this vector is called "perturbation". In contrast to other metaheuristics, its control must be understood as a certain threshold value not a mutation probability. The $PRTVector_j$ consists only of values 0 or 1. These values are generated based on the value of *PRT* parameter; the process is detailed in equation (2), where a *rand* is a pseudo-random number from a uniform distribution within the range of 0 to 1.

$$PRTVector_j = \begin{cases} if\ rand_j < PRT, & 1 \\ otherwise, & 0 \end{cases} \quad j = 1,\ ...,\ D \tag{2}$$

In addition to the All-To-One strategy, there are other basic strategies, namely All-To-Random, and All-To-All. As the name suggests, these strategies involve changing the choice of the Leader to be random and changing the organization of the movement.

B. SOMA T3A

This algorithm features several modifications compared to the original SOMA design, particularly emphasizing the population organization process [17]. This process consists of three repeated activities: organization, migration, and update.

The organization process involves two primary activities: the selection of individuals who will migrate (termed migrants) and the selection of a leader. Initially, *m* individuals are randomly selected from the population, and the *n* best individuals are chosen from these, where n≤m. These individuals become migrants. For leader selection, *k* individuals are randomly selected from the population, and the best individual from this set becomes the leader. The selected migrants then move towards the chosen leader, except when the leader is also one of the migrants (in which case, that individual skips the migration process).

In the migration process, the *PRT* parameter, to which SOMA exhibits significant sensitivity [22], has an adaptive function. The control of this parameter is based on the philosophy of transitioning from exploration to local search, gradually increasing from low values to 1, according to the following equation (3):

$$PRT = 0.05 + 0.90 \cdot \frac{FEs}{MaxFEs} \tag{3}$$

Another change is the gradual reduction of the step size (4), and the number of jumps is fixed and thus not bound to the *pathLength* parameter, as it is in the classic version of SOMA. This means that each individual moves toward the *leader* by a certain number of jumps (hence parameter *Njumps*). At these stepping points, the individual is evaluated by a fitness function.

$$step = 0.15 - 0.08 \cdot \frac{FEs}{MaxFEs} \tag{4}$$

Although this version of the algorithm removed the need to set the *pathLength* and *step* parameters, it added three new parameters for managing the organization process (*m*, *n*, and *k*).

C. DISH algorithm

The DISH algorithm represents the modified jSO [23], a DE-based algorithm from the L-SHADE family [18]. In the SHADE/L-SHADE core functionality, the mutation strategy is "current-to-*p*best/1" and uses four parent vectors – current *i*-th vector $\boldsymbol{x}_{i,G}$, vector $\boldsymbol{x}_{pbest,G}$ randomly selected from the $NP \times p$ best vectors (in terms of objective function value) from current generation *G*. The *p* value is randomly generated by uniform PRNG $U[p_{min},\ 0.2]$, where p_{min} = 2/*NP*. Third parent vector $\boldsymbol{x}_{r1,G}$ is randomly selected from the current generation and last parent vector $\boldsymbol{x}_{r2,G}$ is also randomly selected, but from the union of current generation *G* and external archive *A*. The mutated vector $\boldsymbol{v}_{i,G}$ is generated by (5).

$$\mathbf{v}_{i,G} = \mathbf{x}_{i,G} + F_i\,(\mathbf{x}_{pbest,G} - \mathbf{x}_{i,G}) + F_i\,(\mathbf{x}_{r1,G} - \mathbf{x}_{r2,G}) \tag{5}$$

L-SHADE algorithm uses a crossover scheme to create the trial vector $\boldsymbol{u}_{i,G}$, with a help of the current vector $\boldsymbol{x}_{i,G}$, and the mutated vector $\boldsymbol{v}_{i,G}$, similar as generic DE with the following differences. Control parameters scaling factor *F* and crossover rate *CR* are not static. Instead, the normal distribution is used for CR_i and the *i*-th scaling factor F_i is generated from a Cauchy distribution. In both cases, the historical memories with a size of *H* storing successful values of parameters *F* and *CR* are used.

Also, the selection process is almost identical to the original DE, with the addition of a historical archive. If the objective function value of the trial vector $\boldsymbol{u}_{i,G}$ is better than that of the current vector $\boldsymbol{x}_{i,G}$, the trial vector will become the new individual in new generation $\boldsymbol{x}_{i,G+1}$ and the original vector $\boldsymbol{x}_{i,G}$ will be moved to the external archive of inferior solutions *A*. Otherwise, the original vector remains in the population in the next generation, and the external archive remains unchanged.

Another operation in L-SHADE algorithm is the linear population decrease. The basic idea is to reduce the population size to promote exploitation in later phases of the evolution. Therefore, a new population size is calculated after each generation based on the available budget of fitness function evaluation *MAXFES*, user-predefined NP_{init} as the initial population size, and NP_f as the end population size.

Finally, the idea distinguishing DISH from the jSO/L-SHADE algorithm is that the original adaptation mechanism for parameters *F* and *CR* values uses weights based on the improvement of the objective function value, thus promoting exploitation over exploration. The DISH approach is based on the Euclidean distance between the trial and the original individual (please, see details in [19]).

Due to the limited space here, for detailed information about historical memory update processes for *F* and *CR* parameters, population linear decrease calculation, please refer to [18].

D. CMA-ES

CMA-ES is an evolutionary algorithm for optimizing complex, non-linear, continuous, and high-dimensional objective

functions. Introduced by N. Hansen in the mid-1990s [13], CMA-ES is highly effective when the optimization problem is characterized by a lack of gradients, non-separable variables, and noisy or ill-conditioned landscapes [24]. After initialization, the algorithm proceeds through several steps like selection, adaptation, and reproduction. The algorithm terminates after fulfilling the stopping criteria. CMA-ES automatically adapts the step-size and covariance matrix to suit the problem landscape, enabling efficient exploration and exploitation of the search space. We have used the well-documented Python implementation[1] with an accessible hyperparameter *initial_sigma* (step size). For further details, please refer to survey [13].

III. Optimized Model of Time-delay System

This section describes a model for optimizing the identification of a time-delay system. It also discusses the constraints and selection of sampling points or frequency based on the findings from a recent study [12].

A. General Description

The considered time-delay identification problem [25] is defined by a model transfer function $G_m : \mathbb{C} \to \mathbb{C}$ (6),

$$G_{m,\mathbf{p}}(s) = \frac{b_0 + b_{0,\tau} e^{-\tau_0 s}}{s^3 + a_2 s^2 + a_1 s + a_0 + a_{0,\theta} e^{-\theta s}} e^{-\tau s}. \quad (6)$$

Parameters of such a model form a 9-dimensional real vector

$$\mathbf{p} = [b_0, b_{0,\tau}, \tau_0, \tau, a_2, a_1, a_0, a_{0,\theta}, \theta]. \quad (7)$$

We assume that some of the parameters are related due to the static gain, i.e.

$$k = \frac{b_0 + b_{0,\tau}}{a_0 + a_{0,\theta}}, \quad (8)$$

where the value of k is well known (or estimated). We have used the value $k = 0.0322$.

The goal is to find such parameter values to satisfy (9):

$$G_{m,\mathbf{p}}(\mathbf{j}\,\omega_i) = A_i + \mathbf{j}\,B_i, \quad (9)$$

for some fitting points $\omega_1, \ldots, \omega_n$ and some measured values of $A_1, \ldots, A_n$ and $B_1, \ldots, B_n$, where $\mathbf{j}$ is the imaginary unit ($\mathbf{j}^2 = -1$). The fitting points are given in Table I.

We have used the reformulation using the classical least-square approach to solve (9) with metaheuristic algorithms. The final cost (or fitness) function has form (10),

$$\mathcal{C}(\mathbf{p}) = \sum_{i=1}^{n} \left[\left(\Re G_{m,\mathbf{p}}(\mathbf{j}\omega_i) - A_i\right)^2 + \left(\Im G_{m,\mathbf{p}}(\mathbf{j}\omega_i) - B_i\right)^2 \right] \quad (10)$$

This way, we obtain the final version of our optimization problem (11), which is to find such parameter values $\mathbf{p}^*$ that

$$\mathcal{C}(\mathbf{p}^*) = \min_{\mathbf{p}\in\mathcal{D}} \mathcal{C}(\mathbf{p}), \quad (11)$$

where $\mathcal{D}$ is the set of all $\mathbf{p} \in \mathbb{R}^9$ satisfying (8) and further defined constraints (12) – (15). The experiments were conducted in $\mathbb{R}^8$ since parameter b_0 was calculated using (8).

[1] https://github.com/CMA-ES/pycma

TABLE I
Observation data

i	ω_i	A_i	B_i
1	0.0002	0.03238	-0.00284
2	0.0003	0.03213	-0.00424
3	0.0005	0.03137	-0.00694
4	0.0008	0.02962	-0.01063
5	0.001	0.02813	-0.01278
6	0.0012	0.02645	-0.01465
7	0.0015	0.02371	-0.01692
8	0.0018	0.02087	-0.01857
9	0.002	0.01899	-0.01936
10	0.003	0.01063	-0.02054
11	0.005	0.00057	-0.01713
12	0.008	-0.00540	-0.01110
13	0.01	-0.00704	-0.00795
14	0.011	-0.00757	-0.00658
15	0.012	-0.00795	-0.00531
16	0.014	-0.00843	-0.00296
17	0.016	-0.00860	-0.00074
18	0.018	-0.00846	0.00147
19	0.02	-0.00795	0.00377
20	0.025	-0.00346	0.00982

B. Constrains

To achieve appropriate properties of solutions, we use the following mandatory feasibility constraints (12):

$$\tau_0 > 0, \tau > 0, \theta > 0 \quad (12)$$

Further, the stability conditions (13), which must be valid simultaneously:

$$\begin{aligned} &a_2 > 0, a_1 > 0, a_0 + a_{0,\theta} > 0, \\ &a_2 a_1 > a_0, \\ &a_2 a_1 > a_0 + a_{0,\theta}, \\ &\frac{a_{0,\theta}}{\sqrt{(a_0 - a_2\omega^2)^2 + \omega^2(a_1 - \omega^2)^2}} < 1, \quad \forall \omega > 0. \end{aligned} \quad (13)$$

Finally, the minimum-phase condition (14):

$$|b_0| > |b_{0,\tau}|, \quad (14)$$

and other natural conditions (15):

$$a_0 \neq 0, a_{0,\theta} \neq 0, b_{0,\tau} \neq 0. \quad (15)$$

IV. Experimental results

All experiments with the black box optimization problem instance were repeated 30 times with maximum fitness function evaluation (*MAXFES*) set to 400 000. The settings of the search space boundaries are defined in Table II. The hyperparameter settings for all algorithms were obtained from the automatic algorithm configuration tool iRace [21], and the settings are the following:

- SOMA All-To-One (ATO): *NP* = 72, *PRT* = 0.47, $pathLength$ = 2.3, *step* = 0.345,
- SOMA T3A: *NP* = 98, *Njump* = 26, *m* = 38, *n* = 13, $k = 36$,

TABLE II

PARAMETER SEARCH RANGES

Name	L limit	H limit
b_{0D}	0	100
τ_0	0	500
τ	0	1000
a_2	0	2500
a_1	0	2500
a_0	-250	250
a_{0D}	-250	250
θ	0	1000

TABLE III

PARAMETER SEARCH RANGES

Alg.	min	max	median	mean	std.dev.
ATO	1.08e-08	6.23e-05	9.37e-08	2.19e-05	2.93e-05
T3A	1.08e-08	6.23e-05	1.08e-08	1.19e-05	2.37e-05
DISH	**6.69e-13**	**2.17e-05**	**7.36e-13**	**7.21e-07**	**3.94e-06**
CMA-ES	1.08e-08	2.84e-03	8.70e-04	8.43e-04	6.28e-04

- DISH: NP_{init} = 86, NP_f = 14, H = 12,
- CMA-ES: *init_sigma* = 0.71.

The *NP* represents the number of individuals in the population (PopSize). The remaining parameters are explained in the section II "metaheuristic algorithms."

The nature of the optimization problem allows evaluating the fitness function even in the case of infeasible solutions (as long as they lie within bounds). Therefore, complex methods to avoid infeasible solutions on an algorithmic level were not implemented. Still, random re-initialization of infeasible solutions was chosen since the study [12] confirmed the higher efficiency of metaheuristics with population restarting.

The results are shown and organized in the following way: statistical data are presented in Table III, together with convergence plots in Fig. 1, and the box-whisker plots depicted in Fig. 2. The algorithm DISH (highlighted in bold) achieved the best feasible result. The parameter structure of this solution is: {2.4368e-11, 300.7509, 132.6753, 0.1772, 0.0090, 0.0001, -7.6526e-05, 143.0223}, confirming the problem of the near-domain bounds optimization problem (dimensions 1, 4, and 5). The stability of the best solution is also depicted as a Nyquist plot in Fig. 3.

V. CONCLUSIONS

This research aims to enhance our understanding of metaheuristic algorithms in control engineering and investigate their potential for optimizing the parameters of time delay system models. The results support the need to pay careful attention to the core functionality of the metaheuristic algorithm, the choice of its adaptive techniques, and the setting of its hyperparameters (configuration).

Focusing on the individual results, it can be seen from the statistical data, convergence plot (Fig. 1), and box-whisker plots (Fig. 2) that the DISH shows the best performance. This was also confirmed by the Mann-Whitney U test (Wilcoxon rank sum test), revealing the fact, DISH algorithm designed

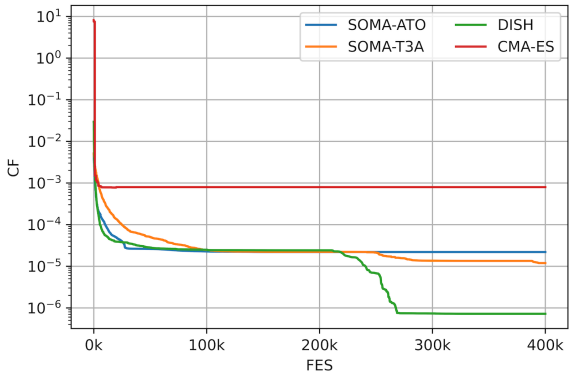

Fig. 1. Mean convergence plot for all algorithms and 30 runs.

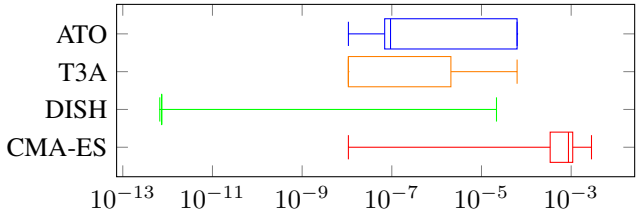

Fig. 2. Box and whisker plot, 30 repeated runs, where ID 1 is for SOMA ATO, 2: SOMA T3A, 3: DISH, and 4: CMA-ES

to support diverse search is significantly better than all other algorithms. Notable results were also achieved by SOMA T3A, which includes internal adaptive functionality for linear control of its own parameter and specific population organization that can support the exploratory capabilities of the algorithm and local search in the later stages of optimization. The CMA-ES algorithm provided the worst performance. The best-found solution was also checked regarding stability. Fig. 3 shows the so-called Nyquist plot, which is used for assessing the stability of a system with feedback. Our obtained model seems highly accurate and fitting for the whole range of frequencies. The presented experimental research has achieved significantly improved results, close to those obtained by theoretically demanding and time-consumptive mathematical-physical modeling requiring prior knowledge about the process.

It can be argued that for this type of real-world problem, it is advantageous to use an algorithm with an organizing process or sub-populations (even with partial restarting or injection of individuals) and to control both diverse search and the exploration-to-exploitation transition. The design of the DISH algorithm with an emphasis on population diversity seems to have aided in the continuation of the search process, as seen in the convergence plot (Fig. 1) in the region after 200,000 FES. Future Research Directions: The paper identifies potential areas for future research, suggesting that further exploration of bound constraints handling methods for metaheuristic algorithms and, of course, identifying the essential algorithm components for using autoconfiguration frameworks could lead to even more effective optimization.

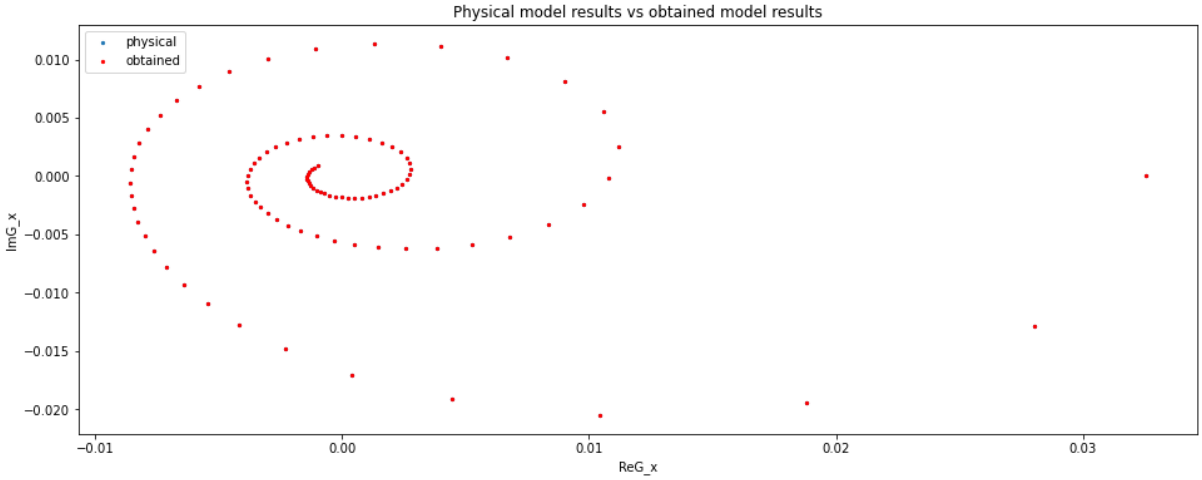

Fig. 3. Nyquist plot for the best individual solution (DISH algorithm)

REFERENCES

[1] A. E. Ezugwu, A. K. Shukla, R. Nath, A. A. Akinyelu, J. O. Agushaka, H. Chiroma, and P. K. Muhuri, "Metaheuristics: a comprehensive overview and classification along with bibliometric analysis," *Artificial Intelligence Review*, vol. 54, pp. 4237–4316, 2021.

[2] E. Fridman, *Introduction to Time-Delay Systems: Analysis and Control*, ser. Systems & Control: Foundations & Applications. Springer International Publishing, 2014.

[3] W. Michiels and S.-I. Niculescu, *Stability, control and computation or time-delay systems. An eigenvalue based approach*. SIAM, 01 2014.

[4] G.-H. Lin, J. Zhang, and Z.-H. Liu, "Hybrid particle swarm optimization with differential evolution for numerical and engineering optimization," *International Journal of Automation and Computing*, vol. 15, no. 1, pp. 103–114, 2018.

[5] W. Gu, Y. Yu, and W. Hu, "Artificial bee colony algorithmbased parameter estimation of fractional-order chaotic system with time delay," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 1, pp. 107–113, 2017.

[6] S. Ikeda and R. Ooka, "Metaheuristic optimization methods for a comprehensive operating schedule of battery, thermal energy storage, and heat source in a building energy system," *Applied energy*, vol. 151, pp. 192–205, 2015.

[7] K. Sörensen, "Metaheuristics—the metaphor exposed," *International Transactions in Operational Research*, vol. 22, no. 1, pp. 3–18, 2015, iSBN: 0969-6016 Publisher: Wiley Online Library.

[8] D. Molina, J. Poyatos, J. D. Ser, S. García, A. Hussain, and F. Herrera, "Comprehensive taxonomies of nature-and bio-inspired optimization: Inspiration versus algorithmic behavior, critical analysis recommendations," *Cognitive Computation*, vol. 12, pp. 897–939, 2020.

[9] J. de Armas, E. Lalla-Ruiz, S. L. Tilahun, and S. Voß, "Similarity in metaheuristics: a gentle step towards a comparison methodology," *Natural Computing*, vol. 21, no. 2, pp. 265–287, 2022.

[10] P. Kipiński, H. Guzowski, A. Urbańczyk, M. Smołka, M. Kisiel-Dorohinicki, A. Byrski, Z. K. Oplatkova, R. Senkerik, L. Pekar, R. Matusu *et al.*, "Socio-cognitive optimization of time-delay control problems using evolutionary metaheuristics," in *2022 IEEE 11th International Conference on Intelligent Systems (IS)*. IEEE, 2022, pp. 1–7.

[11] M. Nabywaniec, H. Guzowski, A. Urbańczyk, M. Smolka, M. Kisiel-Dorohinicki, A. Byrski, Z. K. Oplatkova, R. Senkerik, L. Pekar, R. Matusu *et al.*, "Socio-cognitive optimization of time-delay control problems using agent-based metaheuristics," in *2022 IEEE 11th International Conference on Intelligent Systems (IS)*. IEEE, 2022, pp. 1–7.

[12] H. Guzowski, M. Smołka, A. Byrski, L. Pekar, Z. K. Oplatkova, R. Senkerik, R. Matusu, and F. Gazdos, "Effective parametric optimization of heating-cooling process with optimum near the domain border," in *2022 IEEE 11th International Conference on Intelligent Systems (IS)*. IEEE, 2022, pp. 1–6.

[13] N. Hansen, "The cma evolution strategy: A tutorial," 2016. [Online]. Available: https://arxiv.org/abs/1604.00772

[14] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Computer Journal*, vol. 7, pp. 308–313, 1965.

[15] D. Davendra and I. Zelinka, "Self-organizing migrating algorithm," *New optimization techniques in engineering*, 2016, publisher: Springer.

[16] L. Skanderova, "Self-organizing migrating algorithm: review, improvements and comparison," *Artificial Intelligence Review*, vol. 56, no. 1, pp. 101–172, 2023.

[17] Q. B. Diep, I. Zelinka, S. Das, and R. Senkerik, "SOMA T3A for Solving the 100-Digit Challenge," in *Swarm, Evolutionary, and Memetic Computing and Fuzzy and Neural Computing*, ser. Communications in Computer and Information Science, A. Zamuda, S. Das, P. N. Suganthan, and B. K. Panigrahi, Eds. Cham: Springer International Publishing, 2020, pp. 155–165.

[18] R. Tanabe and A. S. Fukunaga, "Improving the search performance of shade using linear population size reduction," in *2014 IEEE congress on evolutionary computation (CEC)*. IEEE, 2014, pp. 1658–1665.

[19] A. Viktorin, R. Senkerik, M. Pluhacek, T. Kadavy, and A. Zamuda, "Distance based parameter adaptation for success-history based differential evolution," *Swarm and Evolutionary Computation*, vol. 50, p. 100462, 2019.

[20] A. Ghosh, S. Das, A. K. Das, R. Senkerik, A. Viktorin, I. Zelinka, and A. D. Masegosa, "Using spatial neighborhoods for parameter adaptation: An improved success history based differential evolution," *Swarm and Evolutionary Computation*, vol. 71, p. 101057, 2022.

[21] M. López-Ibáñez, J. Dubois-Lacoste, L. P. Cáceres, M. Birattari, and T. Stützle, "The irace package: Iterated racing for automatic algorithm configuration," *Operations Research Perspectives*, vol. 3, pp. 43–58, 2016.

[22] M. Pluhacek, A. Kazikova, T. Kadavy, A. Viktorin, and R. Senkerik, "Explaining soma: the relation of stochastic perturbation to population diversity and parameter space coverage," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2021, pp. 1944–1952.

[23] J. Brest, M. S. Maučec, and B. Bošković, "Single objective real-parameter optimization: Algorithm jso," in *2017 IEEE congress on evolutionary computation (CEC)*. IEEE, 2017, pp. 1311–1318.

[24] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es)," *Evolutionary computation*, vol. 11, no. 1, pp. 1–18, 2003.

[25] L. Pekař, M. Song, S. Padhee, P. Dostálek, and F. Zezulka, "Parameter identification of a delayed infinite-dimensional heat-exchanger process based on relay feedback and root loci analysis," *Scientific Reports*, vol. 12, no. 1, p. 9290, Jun 2022. [Online]. Available: https://doi.org/10.1038/s41598-022-13182-5