Parameter Sensitivity Analysis of the SOMA T3 Algorithm and its Historical Best Modification

Daniel Valenta[1,*,†], Radka Poláková[1,†]

[1]Silesian University in Opava, Faculty of Philosophy and Science, Institute of Computer Science, 746 01 Opava, Czech Republic

Abstract

The Self-Organizing Migrating Algorithm Team-to-Team (SOMA T3) is a population-based metaheuristic that has been shown to be one of the best-performing SOMA variants in recent comparative studies. This paper continues earlier work on improving SOMA T3 by addressing two open questions: how sensitive to its five control parameters the algorithm is, and whether proposed Historical Best memory component can further improve solution quality. A Latin Hypercube Sampling (LHS) study with 30 parameter combinations on the CEC 2020 benchmark suite ($D = 10$ and $D = 30$) shows that the number of jump steps N_{jump} is the most influential parameter, and that values around 30 consistently outperform the default of 45. The size of pool of leader candidates k has a moderate effect; the parameters m, n, and NP are relatively insensitive.

Then, the Historical Best modification (SOMA T3-Hbest) is introduced and evaluated at $D = 10$ and $D = 30$. It consistently improves hybrid functions (F5, F6, F7) but degrades performance on composition and multimodal functions at both tested dimensions. In its current form, it is not a universal improvement, but may serve as a targeted strategy for problems of hybrid character.

Keywords

SOMA, optimization, self-organizing migrating algorithm, parameter sensitivity, latin hypercube sampling, historical best, CEC 2020

1. Introduction

Metaheuristic algorithms are widely used for continuous black-box optimization problems where gradients are unavailable or unreliable. According to the No Free Lunch theorem [9], no single algorithm performs best on all problems. However, within a given problem class, careful algorithm design and parameter tuning can make a large practical difference.

The Self-Organizing Migrating Algorithm (SOMA) [1, 2] is a population-based optimizer inspired by cooperative migration behaviour of social animals. Since its introduction, many variants have been proposed [4, 6, 7, 8], each targeting different weaknesses of the original design. A recent comprehensive review [5] found that the Team-to-Team variant (SOMA T3) [3] remains one of the strongest performers, which makes it a worthwhile base for further study.

Our previous work [15] explored three modifications of SOMA T3: random migration vector length (SOMA T3-RL), a population diversity mechanism (SOMA T3-DIV), and their combination (SOMA T3-RL-DIV). The results showed that diversity maintenance (DIV) consistently improved performance, while the random length modification had little effect on its own. The paper concluded that further improvements remain possible and called for additional modifications and hybrid strategies.

The present paper follows up on that work in two directions. First, we ask whether the default control parameters of SOMA T3 are actually well-chosen, by conducting a systematic parameter sensitivity analysis using Latin Hypercube Sampling (LHS) [12]. LHS provides good coverage of the parameters' space with far fewer experiments than a full-factorial grid. Second, we test a Historical Best (Hbest) modification inspired by Particle Swarm Optimization, where each individual retains its historically best position and uses it to guide migration early in the run. [16]

ITAT'26: Information Technologies – Applications and Theory, September 25–29, 2026, Biele Karpaty, Slovakia

*Corresponding author.

†These authors contributed equally.

daniel.valenta@fpf.slu.cz (D. Valenta); radka.polakova@fpf.slu.cz (R. Poláková)

0009-0005-0781-7755 (D. Valenta); 0000-0002-0782-5668 (R. Poláková)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

published 2026-11-09

CEUR-WS.org/Vol-4323/paper27.pdf

The experiments use the CEC 2020 benchmark suite [10] and the open-source `opfunu` library [11] for function evaluation.

This paper makes three contributions:

- A systematic LHS-based sensitivity study of SOMA T3 control parameters on CEC 2020.
- Identification of parameters' setting that improves over the published defaults, together with settings that should be avoided.
- A proposition and evaluation of a Historical Best modification and analysis of where it helps and where it does not.

2. SOMA T3 Algorithm

In this chapter, we will explain the basics of SOMA T3.

2.1. Basic Mechanism

SOMA maintains a population of NP candidate solutions in the search space $[\text{VarMin}, \text{VarMax}]^D$. In each migration step, a subset of individuals moves toward a selected leader by generating N_{jump} trial solutions along the path between them. For migrant $\mathbf{x}_{\text{mig}}$ and leader $\mathbf{x}_{\text{lead}}$, the t-th trial solution is:

$$\mathbf{x}^{(t)} = \mathbf{x}_{\text{mig}} + \left(\mathbf{x}_{\text{lead}} - \mathbf{x}_{\text{mig}}\right) \cdot s^{(t)} \cdot \mathbf{p}^{(t)}, \tag{1}$$

where $s^{(t)} = t \cdot \Delta s$, Δs is step size, $s^{(t)}$ is size of jump for t-th trial solution and $\mathbf{p}^{(t)} \in \{0, 1\}^D$ is a binary perturbation mask whose components are independently set to 1 with probability PRT (perturbation rate). Both Δs and PRT adapt linearly over the run:

$$\text{PRT} = 0.05 + 0.90 \, \frac{FEs}{FEs_{\max}}, \tag{2}$$

$$\Delta s = 0.15 - 0.08 \, \frac{FEs}{FEs_{\max}}, \tag{3}$$

where FEs is the current number of function evaluations and $FEs_{\max}$ is the total budget. This schedule promotes broad exploration early and fine-grained exploitation late in the run.

2.2. Team-to-Team Selection

In each migration, m candidate migrants are drawn uniformly at random from the population. The n individuals with the best (lowest) fitness among them become active migrants. For each active migrant, k candidate leaders are drawn uniformly at random, and the one with the best fitness becomes the leader (it must differ from the migrant). Each active migrant is replaced by its best offspring if the offspring improves upon the parent's fitness. The five control parameters are therefore m, n, k, N_{jump}, and NP.

3. Parameter sensitivity experiment

In this chapter, we will describe the experiment we conducted. The implementation of SOMA T3 follows the reference Python implementation of SOMA T3A published by the author of original proposition of the algorithm [14].

The sensitivity of SOMA T3 performance to its control parameters was assessed using a simple screening approach. A set of 30 parameter combinations was generated via Latin Hypercube Sampling [12] to ensure representative coverage of the parameter space with a limited number of experiments. For each combination, SOMA T3 was run 21 times per benchmark function and the mean excess above

the optimum $\bar{e}_i$ was computed as the aggregate performance score. The Spearman rank correlation coefficient ρ_s between each parameter value and $\bar{e}_i$ across the 30 combinations was then used to quantify the influence of each parameter on solution quality. Since the data in almost all cases did not follow a normal distribution according to the Shapiro–Wilk test, Spearman's rank correlation coefficient was used to quantify sensitivity.

3.1. Latin Hypercube Sampling

LHS partitions each parameter dimension into $N = 30$ equal intervals and draws exactly one sample per interval [12]. This guarantees that all marginal distributions are well represented even with a small number of combinations, making it suitable for sensitivity screening.

We used the LHS implementation in `scipy.stats.qmc` [13] with $N_{\text{samples}} = 30$ combinations. The parameter ranges are listed in Table 1; the published defaults fall roughly in the centre of each range. After sampling, every combination was clipped to satisfy the following structural constraints of SOMA T3. The parameter ranges were selected based on the following considerations. For m, the range $\langle 5, 15\rangle$ spans ±5 around the default of 10; values below 5 would result in a tournament too small to provide meaningful selection pressure, while values above 15 would frequently violate the structural constraint $m \leq \lfloor NP/4 \rfloor$ for smaller populations. For n, the minimum of 2 was chosen because a single active migrant would make migration impractically slow; the maximum of 8 was chosen to ensure that the number of migrations within the fixed evaluation budget remains reasonable; larger values of n would consume too many evaluations per migration and leave too few migrations for the population to evolve. For k, the range $\langle 5, 20\rangle$ spans roughly ±5 around the default of 15. The lower bound of 5 was set to avoid near-random leader selection – at $k \leq 4$ the tournament provides almost no selection pressure. The upper bound of 20 was chosen to prevent leader selection from becoming near-deterministic: at large k the leader is almost always the global best individual, which changes the stochastic character of SOMA T3 and increases computational cost; values above 20 were therefore not considered. For N_{jump}, the range $\langle 20, 60\rangle$ spans roughly ±15 around the default of 45. The lower bound of 20 ensures that enough offspring are generated along each migration path for meaningful exploration. Finally, for NP, the lower bound of 70 guards against premature convergence and ensures the structural constraints on m and k remain satisfiable across all combinations; the upper bound of 150 was set to keep computational costs manageable given the total number of runs ($30 \times 10 \times 21 = 6{,}300$ per dimension).

$$m \leq \lfloor NP/4 \rfloor, \tag{4}$$

$$2 \leq n \leq m - 1, \tag{5}$$

$$n + 1 \leq k \leq \lfloor NP/3 \rfloor. \tag{6}$$

Table 1
Published defaults [3] and parameter ranges.

Parameter	Symbol	Min	Max	Default
Migration candidates	m	5	15	10
Active migrants	n	2	8	5
Leader candidates	k	5	20	15
Jump steps	N_{jump}	20	60	45
Population size	NP	70	150	100

3.2. Benchmark and Metrics

The complete list of the 30 LHS combinations after applying the structural constraints is given in Table 2.

All experiments use the CEC 2020 benchmark suite [10]: F1 (unimodal), F2–F4 (basic multimodal), F5–F7 (hybrid), and F8–F10 (composition). The evaluation budget is $FEs_{\max} = D \times 10^5$. Each combination

Table 2
30 LHS parameter combinations after applying structural constraints (same set used for both $D = 10$ and $D = 30$).

#	m	n	k	N_{jump}	NP
1	13	2	11	58	115
2	8	4	15	29	143
3	8	5	9	55	71
4	14	4	19	30	148
5	6	5	12	33	103
6	13	7	10	50	74
7	14	6	18	38	106
8	6	5	13	40	111
9	12	3	16	48	86
10	8	4	14	48	134
11	14	7	9	20	145
12	11	3	18	46	91
13	12	6	11	26	92
14	11	2	6	37	125
15	12	3	19	28	129
16	10	5	7	33	76
17	7	4	17	36	86
18	7	5	13	25	122
19	9	8	20	42	120
20	10	5	6	53	97
21	10	3	8	56	95
22	5	4	14	24	137
23	8	7	12	54	131
24	12	4	16	51	116
25	9	5	6	22	137
26	5	4	17	41	79
27	6	5	15	32	82
28	15	3	8	43	142
29	9	6	7	59	101
30	13	4	10	45	110

was evaluated with 21 independent runs per function on both $D = 10$ ($30 \times 10 \times 21 = 6{,}300$ runs, wall-clock ≈ 14.9 hours) and $D = 30$ ($30 \times 10 \times 21 = 6{,}300$ runs, wall-clock ≈ 39.6 hours), distributed across 11 CPU cores (Apple MacBook Pro 14", CPU: M3 Pro, 2023). The same 30 parameter combinations (Table 2) were used for both dimensions to allow direct comparison.

Solution quality $\tilde{y}_{i,f}$ is the median fitness over 21 runs. For cross-function comparison we use the excess above the known optimum f_f^*:

$$e_{i,f} = \tilde{y}_{i,f} - f_f^*, \quad 1 \le i \le 30, \quad 1 \le f \le 10, \tag{7}$$

and the mean excess across all ten functions:

$$\bar{e}_i = \frac{1}{10} \sum_{f=1}^{10} e_{i,f}, \quad 1 \le i \le 30. \tag{8}$$

4. Parameter sensitivity results

In this chapter, we will evaluate the results of the experiment.

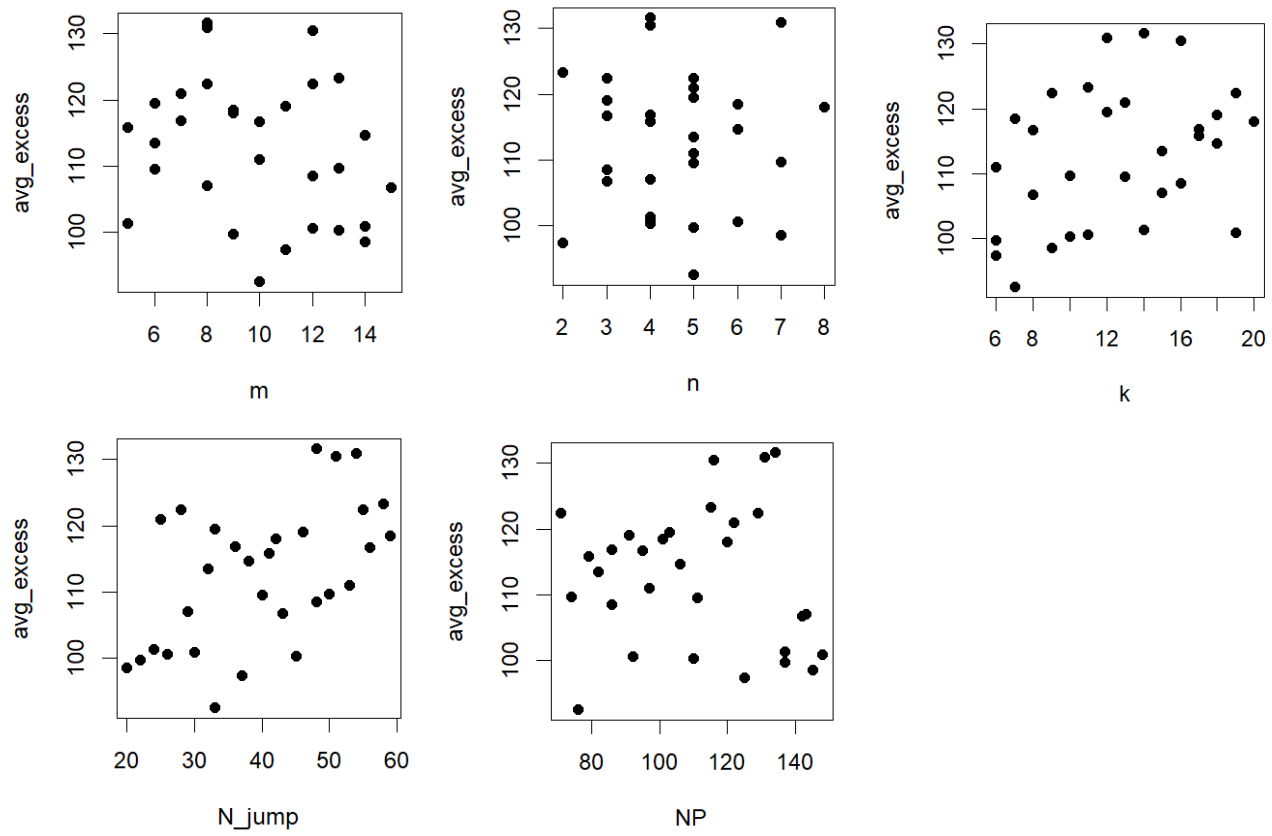


Figure 1: Parameter sensitivity for $D = 10$ (21 runs per combination per function).

4.1. Results in $D = 10$

All data has not normal distribution (tested by Shapiro-Wilk normality test), so sensitivity is quantified by the Spearman rank correlation coefficient between each of parameters (m, n, k, N_{jump}, NP) and $\bar{e}_i$ across the 30 combinations. Figure 1 shows scatter plots for $D = 10$ (21 runs).

The Spearman correlation coefficients for $D = 10$ are $r_{s(m)} \doteq -0.26$, $r_{s(n)} \doteq -0.03$, $r_{s(k)} \doteq 0.36$, $r_{s(N_{\text{jump}})} \doteq 0.51$, $r_{s(NP)} \doteq -0.16$.

We tested if their theoretical equivalents are equal to zero or not. According result of the statistical tests, only ρ_s for dependence of k and $\bar{e}$ and N_{jump} and $\bar{e}$ is not zero at significance level of 0.05.

N_{jump} is the very influential parameter ($r_s \doteq 0.51$): larger values lead to worse results. k shows a positive effect ($r_s \doteq 0.36$), suggesting that larger leader candidate pools are mildly harmful, possibly because they increase the chance of selecting a distant leader.

The best overall combination for $D = 10$ is $m = 10$, $n = 5$, $k = 7$, $N_{\text{jump}} = 33$, $NP = 76$ ($\bar{e} = 92.47$), and the worst is $m = 8$, $n = 4$, $k = 14$, $N_{\text{jump}} = 48$, $NP = 134$ ($\bar{e} = 131.64$) – a 42% difference.

Table 3 lists the best combination per function.

4.2. Results in $D = 30$

Figure 2 shows scatter plots for $D = 30$.

Here, only mean excess has normal distribution and other data have not the distribution (tested by Shapiro-Wilk normality test), so that is why we again compute Spearman correlation coefficients.

The correlation coefficients for $D = 30$ are $r_{s(m)} \doteq 0.01$, $r_{s(n)} \doteq 0.05$, $r_{s(k)} \doteq -0.58$,

Table 3
Best parameter combination per function, (m, n, k, N_{jump}, NP) is i-th combination of parameters, $D = 10$ (21 runs).

F	$\tilde{y}_i$	m	n	k	N_{jump}	NP
F1	100.00	13	2	11	58	115
F2	1175.04	7	5	13	25	122
F3	711.20	5	4	14	24	137
F4	1900.35	10	5	7	33	76
F5	1700.00	13	7	10	50	74
F6	1601.64	9	5	6	22	137
F7	2118.04	10	5	7	33	76
F8	2300.00	13	2	11	58	115
F9	2600.00	13	2	11	58	115
F10	2961.63	12	3	19	28	129

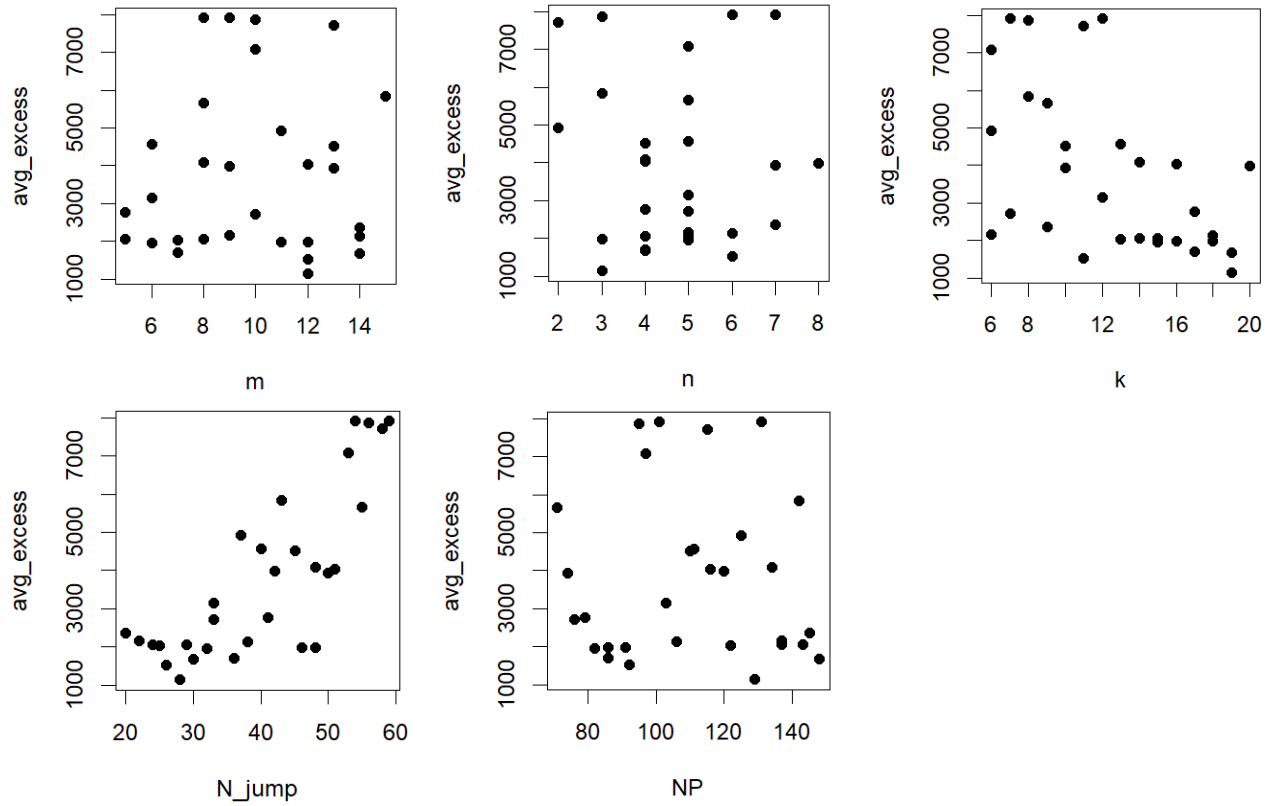


Figure 2: Parameter sensitivity for $D = 30$ (21 runs per combination per function).

$r_{s(N_{\text{jump}})} \doteq 0.73$, $r_{s(NP)} \doteq -0.003$.

Also in this case, we tested if the theoretical correlation coefficients are or are not equal to zero. According to the results of the statistical tests done again at significance level of 0.05, ρ_s, similarly as for $D = 10$, is not equal to zero for k (and $\bar{e}$) and N_{jump} (and $\bar{e}$).

At higher dimension ($D = 30$), the dependence changes substantially. N_{jump} becomes even more dominant ($r_s = 0.73$): large jump counts are strongly harmful in $D = 30$. The mean excess ranges from 1,143 to 7,919 – a nearly 7-fold difference between the best and worst combination, compared to only 42% at $D = 10$. This shows that parameter choice becomes far more critical as dimension grows.

The k reverses sign compared to $D = 10$ ($r_{s_{D=30}} = -0.58$ vs $r_{s_{D=10}} = +0.36$): at $D = 30$ a larger leader

candidate pool is now *beneficial*, likely because the larger search space makes it more important to select a high-quality leader rather than a nearby one.

The best overall combination for $D = 30$ is $m = 12$, $n = 3$, $k = 19$, $N_{\text{jump}} = 28$, $NP = 129$ ($\bar{e} = 1{,}143$). This combination sets both high-impact parameters in the direction favoured by the sensitivity analysis: a small $N_{\text{jump}} = 28$ (consistent with $r_s = 0.73$) and a large $k = 19$ (consistent with $r_s = -0.58$), which serves as an internal validation of the sensitivity findings.

Table 4 lists the best combination per function for $D = 30$.

Table 4
Best parameter combination per function, (m, n, k, N_{jump}, NP) is i-th combination of parameters, $D = 30$ (21 runs).

F	$\tilde{y}_i$	m	n	k	N_{jump}	NP
F1	100.00	13	2	11	58	115
F2	4502.73	14	7	9	20	145
F3	751.66	5	4	14	24	137
F4	1902.48	9	5	6	22	137
F5	7174.93	12	3	19	28	129
F6	1623.22	14	7	9	20	145
F7	3363.32	12	3	19	28	129
F8	2401.70	14	7	9	20	145
F9	2715.25	8	7	12	54	131
F10	2864.06	12	3	19	28	129

4.3. Comparison of $D = 10$ and $D = 30$

Table 5 summarises the values of Spearman correlation coefficients and also informs us if they equal or do not equal to zero for both dimensions side by side.

Table 5
Values of Spearman correlation coefficients between each parameter and mean excess $\bar{e}$ and information about their equality to zero (result of statistical test at significance level 0.05) for $D = 10$ and $D = 30$.

Parameter	r_s ($D = 10$)	$\neq 0$	r_s ($D = 30$)	$\neq 0$
m	−0.26		0.01	
n	−0.03		0.05	
k	0.36	*	−0.58	*
N_{jump}	0.51	*	0.73	*
NP	−0.16		−0.003	

The main finding that holds across both dimensions is the strong positive effect of N_{jump}: keeping it low (around 20–35) is the single most important tuning decision regardless of dimension. The role of k reverses with dimension, and m, n, NP remain relatively unimportant throughout.

As a safe starting point across both dimensions, we suggest $N_{\text{jump}} \approx 30$, $k \approx 15$, and default values for m, n, and NP as were determined in [3].

5. Historical Best Modification

In addition to the parameter sensitivity study, this paper explores a modification of the SOMA T3 migration mechanism. Our previous work [15] concluded that further improvements remain possible and called for additional modifications and hybrid strategies. In this section, we introduce and evaluate one such modification – the Historical Best (Hbest) component, in which each individual retains its historically best position and uses it to guide migration.

5.1. Motivation and Design

In standard SOMA T3, each migrant moves purely toward the current leader. We added a Historical Best (Hbest) component: each individual stores its historically best position and the migration target blends the leader with that position using an adaptive weight:

$$\mathbf{x}_{\text{eff}} = \alpha \cdot \mathbf{x}_{\text{lead}} + (1 - \alpha) \cdot \mathbf{x}_{\text{Hbest}}, \quad \alpha = \frac{FEs}{FEs_{\max}}. \tag{9}$$

Early in the run ($\alpha \approx 0$), the migrant is pulled mainly toward its own best-known position, encouraging local refinement. As the budget is consumed ($\alpha \to 1$), the target shifts fully toward the global leader, increasing convergence pressure. The historical best of each individual is updated whenever an offspring improves upon the individual's current record.

5.2. Results and Analysis

SOMA T3-Hbest was compared against the original SOMA T3 on all ten CEC 2020 functions using the default parameter configuration and 21 independent runs per function at both $D = 10$ and $D = 30$.

The experiment was performed with the default SOMA T3 parameters for both variants. The Wilcoxon signed-rank test confirmed that there are significant differences between results of both tested algorithms in all cases except function F1 in both tested dimensions.

Results are shown in Tables 6 and 7.

Table 6
Median and standard deviation over 21 runs, $D = 10$. **Bold**: better median.

F	SOMA T3		SOMA T3-Hbest	
	Median	Std	Median	Std
F1	**100.00**	0.00	**100.00**	0.00
F2	1257.29	105.43	**1165.11**	60.93
F3	**711.69**	2.03	712.47	3.08
F4	1900.51	0.16	**1900.45**	0.10
F5	**1700.21**	0.19	**1700.21**	0.89
F6	1612.97	108.29	**1601.43**	46.00
F7	2238.14	149.84	**2100.97**	133.47
F8	**2300.00**	0.82	2303.03	34.10
F9	**2600.00**	53.01	2704.02	53.07
F10	**2961.64**	17.45	2970.84	6.99

Table 7
Median and standard deviation over 21 runs, $D = 30$. **Bold**: better median.

F	SOMA T3		SOMA T3-Hbest	
	Median	Std	Median	Std
F1	**100.00**	0.00	**100.00**	0.00
F2	**4874.85**	416.17	6968.39	710.50
F3	**761.68**	9.20	995.67	81.57
F4	**1902.79**	0.87	1903.01	1.05
F5	29307.73	32499.20	**6852.46**	2951.49
F6	2066.96	312.70	**2046.19**	377.04
F7	9345.15	17287.42	**3736.74**	1187.34
F8	**2405.87**	3.61	2413.52	6.29
F9	**2730.30**	51.89	2785.22	45.17
F10	**2864.79**	1.19	2867.79	2.12

At $D = 10$, SOMA T3-Hbest improves on four functions (F2, F4, F6, F7) and ties on two (F1, F5). The most notable gains are on F6 (-7.0%) and F7 (-6.1%), both hybrid functions, where historical memory helps the migrant stay near a good region and avoid being drawn to a poor leader. However, the modification degrades F8, F9, and F10, and F8 in particular sees its standard deviation jump from 0.82 to 34.10 – indicating high instability on this composition function.

At $D = 30$ the picture reverses sharply. Hbest improves only F5, F6, and F7, while degrading six of the remaining functions; F1 is solved at the optimum by both variants.
The mean excess across all functions drops from 4,006 (original) to 1,437 (Hbest) – driven almost entirely by the dramatic improvement on F5 (from 29,308 to 6,852) and F7 (from 9,345 to 3,737), where the original SOMA T3 is highly unstable (std $> 17{,}000$). On the remaining functions Hbest is consistently worse, suggesting that historical memory accumulated early in a 30-dimensional space is often misleading rather than helpful.

6. Discussion

The sensitivity study give a consistent picture of SOMA T3 behaviour across two tested dimensions.

The most robust finding is that N_{jump} is the dominant parameter at both $D = 10$ ($r_s \doteq 0.51$) and $D = 30$ ($r_s \doteq 0.73$). Reducing it from the default 45 to roughly 20–35 consistently improves results. Shorter migration paths force more frequent re-evaluation of the population structure and prevent migrants from overshooting good regions, which becomes increasingly important in higher dimensions.

The role of k reverses with dimension. At $D = 10$, a smaller leader pool is slightly better ($r_s \doteq +0.36$); a larger pool is clearly better ($r_s \doteq -0.58$) at $D = 30$. In high-dimensional spaces, the population is more spread out and selecting from more candidates helps find a truly good leader. A practical implication is that k should be increased when scaling to higher dimensions.

According our foundings, the setting of parameters m, n, and NP is relatively unimportant at both dimensions, statistical tests confirmed that corresponding correlation coefficients should be considered as zero. Users can keep the original defaults for these and focus tuning on N_{jump} and, at higher dimensions, k.

Parameter choice is far more critical at $D = 30$. The mean excess ranges from 1,143 to 7,919 across the 30 combinations – a nearly 7-fold difference between the best and worst setting. At $D = 10$, the same range was only 42% (92 to 132). This suggests that poorly chosen parameters are merely suboptimal in low dimensions but can be catastrophic in high dimensions.

Hybrid functions F5 and F7 are particularly challenging at $D = 30$. The best median found for F5 was 7,175 against an optimum of 1,700, and for F7 it was 3,363 against 2,100. No parameter combination in our study came close to the optimum on these functions at $D = 30$, whereas at $D = 10$ both were solved reliably. This points to a structural limitation of the SOMA T3 migration mechanism on hybrid landscapes in higher dimensions, independent of parameter settings, and warrants dedicated future investigation.

The Hbest modification (Section 5) shows mixed results and in its current form is not a general improvement over SOMA T3. It consistently helps on hybrid functions (F5, F6, F7), where historical memory prevents migrants from being drawn to poor leaders in a structured landscape. On composition and multimodal functions it is harmful, as early accumulated memory misleads rather than guides. The modification may therefore be useful as a targeted strategy when the problem is known to be of hybrid type, but should not be applied blindly across problem classes.

7. Conclusion

We presented a parameter sensitivity study of SOMA T3 using Latin Hypercube Sampling on the CEC 2020 benchmark suite at both $D = 10$ and $D = 30$, using 30 parameter combinations and 21 independent runs each.

The key finding is that N_{jump} is the very influential parameter at both dimensions ($r_s \doteq 0.51$ at $D = 10$, $r_s \doteq 0.73$ at $D = 30$). Values around 20–35 consistently outperform the default of 45. The role of k reverses with dimension: a small leader pool is slightly better at $D = 10$ ($r_s \doteq +0.36$) but a large pool is clearly better at $D = 30$ ($r_s \doteq -0.58$). The parameters m, n, and NP are relatively insensitive at both dimensions.

Based on these findings we recommend:

- $N_{\text{jump}} \approx 25$–$35$ for both tested dimensions.
- $k \approx 7$–10 for $D = 10$; $k \approx 15$–20 for $D = 30$.
- m, n, NP: original defaults are adequate.

The best combinations found were $m = 10$, $n = 5$, $k = 7$, $N_{\text{jump}} = 33$, $NP = 76$ ($\bar{e} = 92.47$) for $D = 10$, and $m = 12$, $n = 3$, $k = 19$, $N_{\text{jump}} = 28$, $NP = 129$ ($\bar{e} = 1{,}143$) for $D = 30$.

In Section 5, Historical Best modification (SOMA T3-Hbest) was evaluated at both $D = 10$ and $D = 30$ with 21 runs per function. The results show that Hbest consistently improves hybrid functions ($F5$, $F6$, $F7$) at both dimensions, but degrades performance on composition and multimodal functions. In its current form, Hbest is not a viable path toward a universal improvement of SOMA T3; it is rather a targeted tool that may be useful when the problem type is known in advance to be of hybrid character.

SOMA T3 remains one of the strongest SOMA variants according to recent comparisons [5], and the present results confirm that well-chosen parameters and targeted modifications can further improve its performance.

Future work will extend the parameter sensitivity analysis to higher dimensions ($D \in \{50, 100\}$) to verify whether the identified trends hold, and will apply the same analysis to the improved variants from our previous work [15] to determine whether their optimal parameter settings differ from those of the original SOMA T3. In future analysis, a larger sample of LHS combinations will be studied.

Acknowledgments

This work was supported by the Silesian University in Opava under the Student Funding Plan, project SGS/9/2024.

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT-4o in order to: Grammar and spelling check. After using these tools/services, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] I. Zelinka, "SOMA – Self-Organizing Migrating Algorithm," in *New Optimization Techniques in Engineering*, Springer, Berlin, Heidelberg, 2004, pp. 167–217.

[2] I. Zelinka, J. Lampinen, and L. Nolle, "SOMA: Self-Organizing Migration Algorithm," in *Proc. 13th International Conference on Process Control*, 2001, pp. 100–105.

[3] Q. B. Diep, "Self-Organizing Migrating Algorithm Team To Team Adaptive – SOMA T3A," in *Proc. IEEE Congress on Evolutionary Computation (CEC)*, Wellington, New Zealand, 2019, pp. 1182–1187. doi:10.1109/CEC.2019.8790202.

[4] L. Škanderová, T. Fabian, and I. Zelinka, "Self-Adapting Self-Organizing Migration Algorithm," *Swarm and Evolutionary Computation*, vol. 51, 2019. doi:10.1016/j.swevo.2019.100593.

[5] L. Škanderová, "Self-Organizing Migrating Algorithm: Review, Improvements and Comparison," *Artificial Intelligence Review*, vol. 56, pp. 101–172, 2023. doi:10.1007/s10462-022-10167-8.

[6] T. Kadavy, A. Viktorin, M. Pluhacek, and S. Kovář, "On Modifications Towards Improvement of the Exploitation Phase for SOMA Algorithm with Clustering-Aided Migration and Adaptive Perturbation Vector Control," in *Proc. 2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2021, pp. 1–8. doi:10.1109/SSCI50451.2021.9659916.

[7] M. Matusikova, M. Pluhacek, T. Kadavy, A. Viktorin, and R. Senkerik, "Exploring Adaptive Components of SOMA," in *Proc. 2023 Genetic and Evolutionary Computation Conference Companion (GECCO)*, 2023, pp. 1803–1811. doi:10.1145/3583133.3596421.

[8] R. Senkerik *et al.*, "Maximizing Efficiency: A Comparative Study of SOMA Algorithm Variants and Constraint Handling Methods for Time Delay System Optimization," 2023, pp. 1821–1829. doi:10.1145/3583133.3596417.

[9] D. H. Wolpert and W. G. Macready, "No Free Lunch Theorems for Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 67–82, 1997.

[10] C. Yue, P. N. Suganthan, J. Liang, B. Qu, K. Yu, H. Zhu, and Y. Yan, "Problem Definitions and Evaluation Criteria for the CEC 2020 Special Session on Single Objective Bound Constrained Numerical Optimization," Technical Report, Nanyang Technological University, Singapore, 2019.

[11] N. V. Thieu, "Opfunu: An Open-Source Python Library for Optimization Benchmark Functions," *Journal of Open Research Software*, vol. 12, 2024. doi:10.5334/jors.508.

[12] M. D. McKay, R. J. Beckman, and W. J. Conover, "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics*, vol. 21, no. 2, pp. 239–245, 1979.

[13] P. Virtanen *et al.*, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[14] diepquocbao, *SOMA-T3A-Python*, GitHub repository, 2022. [Online]. Available: https://github.com/diepquocbao/SOMA-T3A-Python

[15] R. Poláková and D. Valenta, "Effective Modifications of Self-Organizing Migrating Algorithm," in L. Ciencialová et al. (eds.), *Proceedings of the 25th Conference Information Technologies – Applications and Theory (ITAT 2025)*, CEUR-WS, Aachen, 2025, pp. 280–291, ISSN 1613-0073.

[16] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," in *Proc. IEEE International Conference on Neural Networks*, vol. IV, 1995, pp. 1942–1948. doi:10.1109/ICNN.1995.488968.