fSDE: efficient evolutionary optimisation for many-objective aero-engine calibration

Jialin Liu¹ · Qingquan Zhang¹ · Jiyuan Pei¹ · Hao Tong¹ · Xudong Feng^{2,4} · Feng Wu^{3,4}

Received: 30 December 2020 / Accepted: 10 April 2021 / Published online: 13 May 2021
© The Author(s) 2021

Abstract

Engine calibration aims at simultaneously adjusting a set of parameters to ensure the performance of an engine under various working conditions using an engine simulator. Due to the large number of engine parameters to be calibrated, the performance measurements to be considered, and the working conditions to be tested, the calibration process is very time-consuming and relies on the human knowledge. In this paper, we consider non-convex constrained search space and model a real aero-engine calibration problem as a many-objective optimisation problem. A fast many-objective evolutionary optimisation algorithm with shift-based density estimation, called fSDE, is designed to search for parameters with an acceptable performance accuracy and improve the calibration efficiency. Our approach is compared to several state-of-the-art many- and multi-objective optimisation algorithms on the well-known many-objective optimisation benchmark test suite and a real aero-engine calibration problem, and achieves superior performance. To further validate our approach, the studied aero-engine calibration is also modelled as a single-objective optimisation problem and optimised by some classic and state-of-the-art evolutionary algorithms, compared to which fSDE not only provides more diverse solutions but also finds solutions of high-quality faster.

Keywords Engine calibration · Many-objective optimisation · Multi-objective optimisation · Constrained optimisation · Evolutionary algorithm

Introduction

Multi-objective optimisation has been widely applied to diverse real-world problems [12,24], while many-objective optimisation problems are under-exploited but commonly seen in real life [17]. An important example is the engine calibration within car engine design [28].

Engine calibration aims at simultaneously adjusting a set of parameters to ensure the performance of an engine under various working conditions using an engine simulator. A

working condition is often described by a number of parameters depending on the actual usage of engine, such as the speed. Traditional calibration process is achieved by manually changing the engine parameters, simulating its state under different working conditions, comparing its states on engine maps [23], and repeating the above steps until the states are closed to some desired states. Figure 1 illustrates the manual process of adjusting engine parameters. The calibration process is usually time-consuming and highly relies on the human knowledge. Ma et al. [22,23] proposed a multi-objective optimisation approach to automate the gasoline direct injection engine calibration process, in which five engine parameters were adjusted under four cases and three evaluation criteria were considered for each case. However, some of the engines, such as aero-engines, are controlled by dozens of parameters. To achieve optimal performance, the engines should be examined under more working conditions. Due to the large number of engine parameters to be adjusted, the performance measurements to be considered, as well as

✉ Feng Wu
wufeng_my@mail.nwpu.edu.cn

¹ Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China

² Xidian University, Xi'an, China

³ School of Power and Energy Northwestern Polytechnical University, Xi'an, China

⁴ AECC, Chengdu, China

the working conditions to be tested, the calibration process is a crucial many-objective optimisation problem.

In this work, we consider a real aero-engine with non-convex search space of 27 parameters (e.g., compressor pressure ratio, compressor efficiency, and combustion efficiency) and its expected performance under 33 different flight conditions. An engine state is described by 15 measurements, such as compressor exit pressure and fuel flow parameters. Therefore, 27 parameters need to be adjusted at the same time to make the engine's states as closed to the given measurement data as possible.

We model this calibration process as a many-objective optimisation problem. A fast many-objective evolutionary optimisation algorithm with shift-based density estimation is designed to search for parameter setting with acceptable performance accuracy and improve the calibration speed. During optimisation, a numerical simulation program is used to simulate the performance of the engine given a parameter setting and a working condition. Our approach is compared to several state-of-the-art many- and multi-objective optimisation algorithms on this real-world engine calibration problem and achieves superior performance.

The main contributions of this paper are as follows.

- Efficient calibration of a real aero-engine: a real aero-engine model and real data of performance measurements are used in this work.
 - Large parameter search space: We consider 27 parameters of this aero-engine. To our best knowledge, no existing work has ever calibrated a such big number of parameters.
 - Large number of criteria to be optimised: Four hundred and ninety-five criteria are considered in our aero-engine calibration process. To our best knowledge, no existing work on engine calibration has ever considered a such large number of criteria to be optimised.
 - Non-convex constrained parameter space: The calibration of model parameter is a constrained optimisation problem. The parameter space of our aero-engine model is non-convex and black-box. Given a parameter setting, the computational fluid dynamics (CFD) simulation of engine dynamics is able to determine if a given parameter setting is feasible or not, but it is time-consuming as an evaluation of a parameter setting.
- Two optimisation approaches for automating the calibration process:
 - Many-objective optimisation with a simple yet effective fSDE: We model the calibration process as a many-objective optimisation problem and propose

a fast many-objective evolutionary algorithm with Shift-based Density Estimation (fSDE) for automating and speeding up the calibration process. The proposed fSDE shows its superior or competitive performance to some state-of-the-art many- and multi-objective evolutionary algorithms on our real aero-engine calibration problem and a well-known many-objective optimisation benchmark test suite.

- Single-objective optimisation: To further validate our approach, the process is also modelled as a single-objective optimisation problem with a linear aggregation of criteria to be optimised. Several classic and state-of-the-art evolutionary algorithms have been applied, but their efficiency is worse than our proposed fSDE, probably due to the infeasible parameter settings.
- Inspiration to human engineers:
 - Fault diagnosis assistant: It has been observed that very different engine parameter settings lead to similar performance, which motivates human engineers to seek for the reasons and understand why it happened (e.g., sensibility of parameter, faults, or bias in computational model).
 - Additional parameters and measurements: As different settings lead to similar performance given the current parameters, measurements, and operation points, human decision makers can consider additional calibration parameters, measurements, or operation points.
 - Novelty of design: From a perspective of discovering engines, instead of problem solving, diverse settings found by fSDE inspire human engineers to design novel engines.

In the remainder of this paper, “Background” discusses the background and related work. “A real-world aero-engine calibration problem” presents the real aero-engine calibration problem studied in this paper. The many- and single-objective evolutionary optimisation for calibration are provided in “Many-objective evolutionary optimisation for aero-engine calibration” and “Single-objective evolutionary optimisation for aero-engine calibration”, respectively. In particular, “fSDE: fast MaOEA with shift-based density estimation” introduces our proposed fSDE. Our fSDE is compared to some state-of-the-art many- and multi-objective evolutionary algorithms on a well-known many-objective optimisation benchmark test suite (“Efficiency verification of fSDE on benchmark test suite”) and our real aero-engine calibration problem (“fSDE for aero-engine calibration”). Finally, “Conclusion” concludes.

Fig. 1 Manual engine calibration process. An acceptable parameter setting $\mathbf{x}$ should ensure measurement errors within certain threshold at several given operation points $s_1, s_2, \dots, s_k$

Background

Engine calibration

Engine calibration refers to the process of adjusting a set of parameters of a given engine model, denoted as $\mathbf{x} = (x_1, x_2, \dots, x_d)$, to ensure its performance at several operation points or working conditions (illustrated in Fig. 1). An *operation point* s is often determined by a number of values that describe the actual usage of the engine. Taking aero-engines as an example, an operating point is specified by a Mach number, a flight altitude, and a flight state (e.g., taking-off and cruising), which is quantified by a mechanical spool speed [14]. The quality of a parameter setting is indicated by the distances between a number of *measurements*, denoted as $m_1, m_2, \dots, m_h$, and their theoretically optimal values, denoted as $m_1^*, m_2^*, \dots, m_h^*$. Usually, dozens of measurements are considered at each operation point. The lower the distance is, the better the engine parameters are. It is worth mentioning that the engine parameters remain unchanged at different operation points.

Model-based engine calibration with meta-heuristics

The engine calibration process, illustrated in Fig. 1, can be formalised as an optimisation problem given a computational engine model, which simulates the model and calculates the

measurements given a parameter setting and an operation point. Applying black-box optimisation algorithms, such as meta-heuristics, to automate the engine calibration process is straightforward. [28] reviewed the applications of meta-heuristics in car engine calibration published till 2013. In the reviewed work [28], multi-objective evolutionary algorithms have often been applied to optimise at most six objectives at the same time [15,21,26], while single-objective algorithms, such as particle swarm optimisation (PSO), have also been used [25,32]. More recently, Ma et al. [22,23] proposed a multi-objective optimisation approach to automate the gasoline direct injection engine calibration process, in which five engine parameters were adjusted under four operation conditions and three measurements were considered for each operation condition.

Multi- and many-objective optimisation

Various multi-objective evolutionary algorithms (MOEAs) and many-objective evolutionary algorithms (MaOEAs) have been proposed to deal with multi- and many-objective optimisation problems [12,24], mainly categorised into three groups: (i) Pareto dominance-based, (ii) decomposition-based, and (iii) indicator-based. Pareto dominance-based MOEAs aim to use Pareto dominance mechanisms to select non-dominated solutions and apply some additional strategies to improve the diversity of solutions. The non-dominated sorting genetic algorithm II (NSGA-II) [10] is one typical

Fig. 2 Schematic diagram of the engine model studied in this paper. The numbers between brackets are the indices of measurements described in Table 2

MOEA, and other examples include the Pareto envelope-based selection algorithm II (PESA-II) [5] and the improved version of the Strength Pareto Evolutionary Algorithm (SPEA2) [36]. Although Pareto dominance-based MOEAs have good capability, the performance can be worsen rapidly with the increased number of objectives due to the dominance resistance phenomenon [17]. To overcome this issue, relaxed dominance-based MOEAs are proposed which aim to design new dominance relations, such as ϵ -dominance [16] and GrEA [33]. Examples of decomposition-based MOEAs/MaOEAs include the multi-objective evolutionary algorithm based on decomposition (MOEA/D) [34], the non-dominated sorting genetic algorithm III (NSGA-III) [9], and the reference vector-guided evolutionary algorithm (RVEA) [2]. These algorithms tend to decompose the original problem into some simpler multi-objective or single-objective optimisation problems. The two-archive algorithm (Two_Arch2) [31], the indicator-based evolutionary algorithm (IBEA) [35], and the fast hypervolume-based evolutionary algorithm (HypE) [1] are popular indicator-based MOEAs/MaOEAs.

A real-world aero-engine calibration problem

We consider a real aero-engine model parameterised by 27 parameters which need to be adjusted at 33 operation points. The considered engine model is illustrated in Fig. 2.

Twenty-seven engine parameters A total number of 27 engine parameters are adjusted, such as compressor pressure ratio, compressor efficiency, fan inner/outer pressure ratio, high-/low-pressure turbine efficiency, bypass ratio, and heat-transfer coefficient. Table 1 provides the lower and upper bounds of those parameters.

Fifteen measurements per operation point At each operation point, the computational model outputs 15 measurements of the considered real aero-engine, described in Table 2. These measurements describe the environmental condition, the working status, and the performance of the engine. The numbers between brackets in Fig. 2 correspond to the indices of measurements described in Table 2. Their desired values (called “targets” in this paper) for an engine of optimal status vary at different operation points. The targets of 33 operation points are available in our real problem.

Thirty-three operation points The measurements at 33 different operation points are provided. These operation points are carefully selected by a human engineer to represent various states of an aircraft.

Computational engine model A computational engine model (a Windows executable file with extension .exe) is available. It takes a parameter setting x and an operation point s as two inputs. During a simulation, several CFD calculations are performed inside the computational model to simulate the dynamics. If the CFD simulation does not converge, then the input setting is labelled as “infeasible” and the simulated measurement values are considered “unbelievable”; otherwise, the input setting is labelled as “feasible” and the simulated measurements $m_1, m_2, \dots, m_h$ are considered “believable”. The label and simulated measurements are returned, as shown in Algorithm 1. Each simulation consumes approximately 0.2 s. Note that this is not the case of expensive optimisation problem, but the real-time optimisation efficiency is very important in this real-world application.

Evaluation criterion Given a number of operation points s_j , $j \in \{1, \dots, 33\}$, the quality of an engine parameter setting x is often evaluated with the root-mean-square error (RMSE)

Table 1 Lower bound (*LB*) and upper bound (*UB*) of the aero-engine model parameters and a known optimum provided by a human engineer

	Optimum	LB	UB		Optimum	LB	UB		Optimum	LB	UB
1	0.855	0.75	0.89	10	1.00	0.95	1.05	19	0.99	0.99	1.00
2	3.20	2.20	4.50	11	0.98	0.98	1.00	20	1.00	0.99	1.00
3	3.40	2.20	4.50	12	0.98	0.98	1.00	21	0.96	0.92	0.98
4	0.84	0.75	0.89	13	0.9995	0.99	1.00	22	0.97	0.97	0.99
5	0.82	0.75	0.89	14	0.06	0.00	0.10	23	0.99	0.98	1.00
6	0.88	0.80	0.92	15	0.025	0.00	0.10	24	0.99	0.98	1.00
7	0.89	0.80	0.92	16	0.03	0.00	0.05	25	0.99	0.98	1.00
8	0.94	0.90	1.00	17	0.02	0.00	0.05	26	0.50	0.20	0.70
9	0.015	0.00	0.03	18	0.00	0.00	50.00	27	0.24	0.20	0.30

Table 2 Considered measurements at each operation point

Index	Measurement description	Index	Measurement description
1	Fan inner outlet pressure	9	Fan outlet inner temperature
2	Fan outer outlet pressure	10	Fan outlet outer temperature
3	Compressor outlet pressure	11	Compressor outlet temperature
4	Bypass outlet pressure ratio	12	Low-pressure turbine pre-pressure
5	Low-pressure turbine outlet temperature	13	Low-pressure turbine back-pressure
6	Fuel flow	14	Thrust
7	Bypass outlet temperature	15	Exit area of nozzle
8	Physical flow		

Algorithm 1 A call to the computational engine model.**Input:** parameter setting $\mathbf{x}$ and operation point s **Output:** feasibility of $\mathbf{x}$ and simulated measurement values**function** SIMULATE($\mathbf{x}, s$)Simulate engine with parameter $\mathbf{x}$ at operation point s with the computational model**if** CFD simulation converges **then****return** FEASIBLE and simulated measurement values $m_1, m_2, \dots, m_h$ **else****return** INFEASIBLE and simulated measurement values $m_1, m_2, \dots, m_h$ **end if****end function**

target value:

$$e_{i,j}(\mathbf{x}) = \left| \frac{\text{Simulate}(\mathbf{x}, s_j) - m_{i,j}^*}{m_{i,j}^*} \right|. \quad (2)$$

In our real-world scenario, an engine parameter setting $\mathbf{x}$ with $RMSE(\mathbf{x})$ lower than 0.5% is considered acceptable. For a better performance, an $RMSE(\mathbf{x})$ lower than 0.3% is favourable. To achieve a such quality, it takes on average 3 h manual work (as illustrated in Fig. 1) by an experienced human engineer.

computed as follows:

$$RMSE(\mathbf{x}) = \sqrt{\frac{1}{33 * 15} \sum_{j=1}^{33} \sum_{i=1}^{15} \left(\frac{\text{Simulate}(\mathbf{x}, s_j) - m_{i,j}^*}{m_{i,j}^*} \right)^2}, \quad (1)$$

where $m_{i,j}^*$ refers to the target value of the i^{th} measurement of the optimal engine at operation point s_j for any $i \in \{1, \dots, 15\}$ and $j \in \{1, \dots, 33\}$. The error rate of the i^{th} measurement at the j^{th} operation point is calculated as the normalised distance between the simulated value and its

Many-objective evolutionary optimisation for aero-engine calibration**fSDE: fast MaOEA with shift-based density estimation**

In the field of multi-/many-objective optimisation, a performance *indicator* refers to the measure used to evaluate the quality of a solution set provided by an optimiser (e.g., an MOEA) [19]. The quality of a solution set can be described by the extent of its approximation of the Pareto front, in terms of four aspects [19], namely convergence, spread, uniformity, and cardinality. Similarly, indicators can also be applied to approximate the contribution of each individual

(i.e., solution) in a population, and then used as a criterion to differentiate the individuals.

Indicator-based MaOEAs have a significant advantage. They can avoid low selection pressure towards Pareto front via non-dominated sorting thanks to a high proportion of non-dominated solutions [17]. The studies of [17, 19] showed that generally the performance of an indicator-based MaOEA strongly depends on the selection of appropriate indicators and strategies to use them. On one hand, different indicators have a different bias towards the Pareto front. For instance, the work of [18, 31] showed that IBEA using I_ϵ indicator may make the population converge to sub-regions of the Pareto front with bad diversity. On the other hand, strategies using indicators imply large computational consumption and may lead to inefficiency of algorithms [1].

Due to the large number of objectives in this aero-engine calibration problem and the requirement of real-time efficiency in real-world application, we intend to design an indicator-based MaOEA to achieve better performance with acceptable time cost.

Shift-based density estimation (SDE)

Li et al. [20] showed that the *shift-based density estimation* (SDE) is able to detect individuals with good convergence and diversity. Therefore, SDE [20] is chosen as the indicator to guide search processes as finding more diverse parameter settings of high quality is favourable in our case. The shifted density $D(p, \mathcal{P})$ of an individual p corresponding to its belonging population $\mathcal{P}$ is formalised as follows [20]:

$$D(p, \mathcal{P}) = \min_{q \in \mathcal{P}, q \neq p} \left\{ \sqrt{\sum_{i=1}^m (p_i - \max(q_i, p_i))^2} \right\}, \quad (3)$$

where m is the number of objectives. q_i and p_i ($i \in \{1, 2, \dots, m\}$) denote the i th objective value of individuals q and p , respectively. The value $D(p, \mathcal{P})$ is used as the fitness of individual p , thus $fitness_p = D(p, \mathcal{P})$.

Framework of fSDE: less is more

SDE has been integrated into NSGA-II, SPEA2, and PESA-II, and shown to improve the performance of the aforementioned algorithms [20]. However, the aforementioned algorithms are complex and the required operations, such as non-dominate sorting, are time-consuming; while in our scenario, real-time efficiency is crucial. Therefore, we proposed a simple MaOEA framework with SDE, called fSDE, presented in Algorithm 2. First, N solutions are randomly generated as the initial population $\mathcal{P}$. Then, at every generation, a binary tournament selection strategy is processed to generate a mating pool B of size N based on the fit-

ness of individuals in $\mathcal{P}$. After reproducing N offspring, denoted as O , through simulated binary crossover [7] and polynomial mutation [8], an *environmental selection* is processed to delete the N worst solutions from $\mathcal{P} \cup O$ and keep the remained N solutions as the new population. The above steps are repeated until termination criteria are reached. Considering a population of N individuals for a multi-objective optimisation problem with m objectives, calculating the fitness values (i.e., shifted density values) costs $O(mN^2)$. The deletion step in the environmental selection costs $O(N \log N)$. Therefore, the total complexity is $O(mN^2)$.

Algorithm 2 Fast MaOEA with Shift-based density estimation (fSDE).

Input: Population size N , crossover rate p_c , and mutation rate p_m
Output: Final population $\mathcal{P}$
 1: $\mathcal{P} \leftarrow$ Randomly initialise N individuals
 2: Evaluate SDE values of $\mathcal{P}$ // Evaluate initial population
 3: **while** termination criteria not fulfilled **do**
 4: $B \leftarrow$ Select $\frac{N}{2}$ pairs of parents from $\mathcal{P}$ according to their fitness values
 // Binary tournament selection
 5: $O \leftarrow$ Reproduce N offspring from B , with crossover rate p_c and mutation rate p_m
 // Simulated binary crossover and polynomial mutation
 6: Evaluate SDE values of O // Evaluate offspring
 7: $\mathcal{P} \leftarrow$ Find the top N individuals $\in \mathcal{P} \cup O$ based on their fitness values
 // Environmental selection
 8: **end while**

Efficiency verification of fSDE on benchmark test suite

To verify the effectiveness of our proposed fSDE, we compare it to state-of-the-art many- and multi-objective evolutionary algorithms on a commonly used benchmark function set, MaF [3]. The baseline algorithms considered in our comparison study are MOEA/D [34], NSGA-III [9], Two_Arch2 [31], and IBEA [35], taken from the MATLAB-based MOEA platform, PlatEMO-v2.5 [30]. The proposed fSDE is also implemented in MATLAB and integrated into PlatEMO-v2.5. MaF contains 15 test functions, designed for many-objective optimisation aiming to represent various real-world scenarios with many different shapes of Pareto fronts [3]. Table 3 shows the number of objectives considered in this comparison study, the corresponding number of reference vectors, and the corresponding number of function evaluations [29], determined by Das and Dennis's systematic approach [6, 9]. For all algorithms, its population size is set as the number of reference points.

Table 3 Experimental setting of tests on MaF, where d is the number of decision variables

#Objectives	p_c	p_m	#Vectors ($=N$)	#Evaluations
5	0.1	$\frac{1}{d}$	126	63,000
10			275	165,000
20			230	115,000

The hypervolume (HV) is adopted as the performance metric. Normalised objective values are used to compute HV values, and then, the reference point $(1.1, 1.1, \dots, 1.1)$ is used for computing HV. On each benchmark function of MaF, 30 independent optimisation trials have been performed by each algorithm. The averaged HV values, standard deviations, and the results of Wilcoxon rank sum test [4] in solving MaF with 5, 10, and 20 objectives are reported in Tables 4, 5, and 6, respectively.

According to Table 4 (5 objectives), Two_Arch2 performs better than fSDE and its pairwise win/tie/loss counts against fSDE are 8/4/3. fSDE outperforms MOEA/D and NSGA-III in terms of HV values. In addition, the performance of IBEA and NSGA-III is similar to that of fSDE. In solving MaF test suite with ten objectives (cf. Table 5), fSDE has the best overall performance in terms of convergence and diversity among the compared MaOEA. fSDE achieves significantly better HV values than MOEA/D in 12 out of 15 functions. When solving MaF with 20 objectives (cf. Table 6), NSGA-III obtains better HV values than fSDE, while fSDE achieves better quality (HV) than MOEA/D. Moreover, IBEA and Two_Arch2 have similar performance compared to fSDE. To summarise, fSDE achieves superior or competitive performance on MaF compared to the considered baseline algorithms.

fSDE for aero-engine calibration

We model the aero-engine calibration process as a many-objective optimisation problem, aiming at minimising simultaneously all the error rates, Eq. (2). To do so, two naive but effective dimensionality reduction methods (“Naive but effective dimensionality reduction”) and dimensionality reduction via principal component analysis (“Failure of dimensionality reduction via principal component analysis”) have been performed.

Naive but effective dimensionality reduction

Two naive dimensionality reduction methods have been considered: (i) modelling $\frac{1}{15} \sum_{i=1}^{15} e_{i,j}(\mathbf{x})$ with $j \in \{1, \dots, 33\}$ as 33 objectives; and (ii) modelling $\frac{1}{33} \sum_{j=1}^{33} e_{i,j}(\mathbf{x})$ with $i \in \{1, \dots, 15\}$ as 15 objectives. As a baseline, we also (iii) model all the error rates as $33 * 15$ objectives. Twenty inde-

pendent optimisation trials of fSDE with 20,000 simulation calls to the computational model as optimisation budget are applied to each of the above three many-objective optimisation problems, denoted as fSDE-33, fSDE-15, and fSDE-495, respectively.

Quality assessment in objectives Figure 3 illustrates the averaged RMSE, calculated with Eq. (1), over 20 independent trials with respect to the simulation number (left) and time in seconds (right). No obvious difference has been observed among the three cases at the early optimisation stage. However, along with the optimisation time, fSDE-15 (solid black curve) converges slightly better than fSDE-33 and fSDE-495. Figure 4 illustrates the lowest error rate among the 495 error rates among the 20 trials, named as *minmax error*. Lower minmax error implies overall better parameter setting. According to Figs. 3 and 4, although fSDE-33 achieves similar RMSE as the other two cases, its minmax error is higher after 5,000 model simulations. The experimental result is coherent with an engineering perspective, and thus, the values of an identical measurement at various operation points is highly correlated and can be aggregated to a single value, as in case (i), thus fSDE-15.

Quality assessment in decisions Four aero-engine parameter settings with RMSE lower than 0.5% found by fSDE-15, denoted as $\mathbf{x}_a$, $\mathbf{x}_b$, $\mathbf{x}_c$, and $\mathbf{x}_d$, are illustrated in Table 7. For reference, a known optimum parameter setting of this problem, denoted as $\mathbf{x}^*$, is given on the second column. On most of the coordinates, the illustrated parameter settings are very closed to the optimum, except for the parameters at coordinates 15–17. Those parameters at coordinates 15–17 are the volume of cooling flow of the high-pressure turbine guide vane, the high-pressure turbine rotor, the low-pressure turbine guide vane, and the low-pressure turbine rotor. The model is less sensitive to those four parameters.

Quality assessment in efficiency As mentioned previously, in this calibration problem, an experienced human engineer needs approximately 3 h to find a parameter setting with an RMSE value lower than 0.3%. With fSDE, it takes less than 15 min to achieve a such RMSE value and 22 min to achieve an RMSE value of 0.2%. All the experiments in this paper were ran on an i7-8700 Intel processor desktop with 8 cores.

Failure of dimensionality reduction via principal component analysis

To reduce the dimensionality of objectives, principal component analysis (PCA) has been performed. First, we sample 20,000 parameter settings uniformly at random in the legal ranges provided in Table 1 and simulate them at the given operation points. Among the 20,000 samples, only 13,331 settings are feasible. PCA is performed on the 15 averaged

Table 4 Averaged HV values in solving MaF with five objectives

Problem	<i>m</i>	<i>d</i>	MOEA/D	NSGA-III	Two_Arch2	IBEA	fSDE
MaF1	5	14	6.8380e-3 (1.81e-3) –	4.4297e-3 (3.13e-4) –	1.0881e-2 (1.57e-4) +	9.7919e-3 (1.87e-4) +	9.2189e-3 (2.76e-4)
MaF2	5	14	1.7407e-1 (4.01e-4) +	1.7266e-1 (2.92e-3) +	1.8945e-1 (1.77e-3) +	1.8599e-1 (1.90e-3) +	1.5513e-1 (8.55e-3)
MaF3	5	14	9.8285e-1 (3.00e-3) –	9.4750e-1 (1.89e-1) –	8.9579e-1 (2.63e-1) =	6.4381e-1 (3.44e-1) –	9.8932e-1 (2.45e-3)
MaF4	5	14	5.3167e-3 (1.50e-3) –	5.2748e-2 (1.47e-2) –	5.5634e-2 (1.86e-2) –	5.2473e-3 (8.38e-4) –	1.0540e-1 (4.92e-3)
MaF5	5	14	4.1751e-1 (1.15e-1) –	7.6687e-1 (5.79e-2) +	7.2390e-1 (7.62e-3) =	7.7719e-1 (4.63e-2) +	7.0983e-1 (4.73e-2)
MaF6	5	14	9.6949e-2 (4.87e-2) –	1.2252e-1 (1.89e-3) +	1.2955e-1 (3.23e-4) +	1.1494e-1 (6.24e-3) +	9.7414e-2 (2.01e-2)
MaF7	5	24	1.1380e-1 (2.80e-2) –	2.3812e-1 (6.54e-3) +	2.3259e-1 (7.94e-3) =	2.6730e-1 (4.04e-3) +	2.3331e-1 (7.43e-3)
MaF8	5	2	1.0415e-1 (6.61e-3) =	9.8006e-2 (2.48e-3) –	1.1493e-1 (2.17e-3) +	6.1489e-3 (6.13e-4) –	1.0295e-1 (6.91e-3)
MaF9	5	2	2.9194e-1 (1.22e-2) +	1.7434e-1 (5.17e-2) –	3.0432e-1 (7.76e-3) +	6.1504e-2 (1.69e-2) –	2.5118e-1 (2.64e-2)
MaF10	5	14	9.1465e-1 (4.26e-2) +	8.4923e-1 (4.92e-2) +	9.7018e-1 (2.67e-2) +	9.8040e-1 (2.38e-2) +	7.1510e-1 (6.63e-2)
MaF11	5	14	9.4680e-1 (1.14e-2) –	9.9258e-1 (1.14e-3) +	9.9226e-1 (1.12e-3) +	9.7429e-1 (5.67e-3) +	9.6445e-1 (4.95e-3)
MaF12	5	14	5.8013e-1 (4.18e-2) –	7.2502e-1 (1.94e-2) +	6.9718e-1 (7.96e-3) +	7.2953e-1 (8.12e-3) +	6.7981e-1 (1.09e-2)
MaF13	5	5	2.3055e-1 (6.07e-2) =	1.7681e-1 (2.33e-2) –	2.3483e-1 (1.45e-2) =	7.2404e-3 (2.35e-2) –	2.3992e-1 (1.21e-2)
MaF14	5	100	2.2179e-1 (1.58e-1) +	3.8670e-3 (2.12e-2) –	0.0000e+0 (0.00e+0) –	0.0000e+0 (0.00e+0) –	4.3744e-2 (6.34e-2)
MaF15	5	100	2.5820e-2 (1.19e-2) +	0.0000e+0 (0.00e+0) –	0.0000e+0 (0.00e+0) –	1.1440e-5 (2.84e-5) –	2.9271e-3 (4.65e-3)
+ / = / –			5/2/8	7/0/8	8/4/3	8/0/7	

The standard deviation are provided in brackets. The best HV value for each case is highlighted in bold value. Symbols '+', '-', and '=' indicate that the performance is statistically better, worse, or approximately the same as that obtained by fSDE, respectively, according to the Wilcoxon rank sum test (0.05 significance level)

Table 5 Averaged HV values in solving MaF with ten objectives

Problem	<i>m</i>	<i>d</i>	MOEA/D	NSGA-III	Two_Arch2	IBEA	fSDE
MaF1	10	19	1.9415e-8 (6.47e-9) –	4.4887e-7 (3.19e-8) =	2.6269e-7 (4.46e-7) –	4.5594e-7 (2.68e-8) =	4.3854e-7 (1.06e-7)
MaF2	10	19	2.1051e-1 (4.39e-4) +	2.1661e-1 (5.55e-3) +	2.0552e-1 (2.34e-3) +	2.1756e-1 (3.57e-3) +	1.6665e-1 (9.07e-3)
MaF3	10	19	9.6714e-1 (1.52e-3) –	1.8188e-1 (3.72e-1) –	2.6664e-1 (3.33e-1) –	8.5377e-1 (2.08e-1) –	9.9907e-1 (4.33e-4)
MaF4	10	19	2.2392e-8 (4.41e-9) –	2.3567e-4 (1.82e-5) –	2.7426e-5 (1.39e-5) –	5.0958e-8 (1.85e-7) –	4.1324e-4 (3.73e-5)
MaF5	10	19	3.5660e-1 (9.42e-2) –	9.6952e-1 (2.00e-4) +	7.9539e-1 (1.77e-2) –	9.7487e-1 (2.45e-4) +	9.3955e-1 (3.99e-3)
MaF6	10	19	9.4592e-2 (1.96e-2) +	7.7923e-3 (1.52e-2) –	9.2172e-2 (3.44e-3) +	8.9164e-2 (8.77e-3) +	6.1204e-2 (2.96e-2)
MaF7	10	29	1.3340e-4 (2.93e-4) –	1.7002e-1 (5.56e-3) =	1.1561e-1 (1.57e-2) –	1.9696e-1 (4.90e-3) +	1.6484e-1 (2.06e-2)
MaF8	10	2	6.8674e-3 (8.91e-5) –	9.3474e-3 (2.79e-4) –	1.0216e-2 (1.72e-4) +	8.3470e-5 (1.42e-5) –	9.9408e-3 (3.23e-4)
MaF9	10	2	1.4295e-2 (2.70e-3) –	8.4605e-3 (1.98e-3) –	7.6836e-4 (1.48e-4) –	9.1516e-4 (1.93e-4) –	1.6877e-2 (3.20e-4)
MaF10	10	19	6.4428e-1 (1.35e-1) –	8.9169e-1 (4.90e-2) –	9.9209e-1 (1.37e-2) –	9.9340e-1 (1.59e-3) –	9.9373e-1 (1.40e-2)
MaF11	10	19	9.3198e-1 (5.69e-3) –	9.9582e-1 (2.92e-3) +	9.9688e-1 (8.21e-4) +	9.8636e-1 (3.22e-3) –	9.9111e-1 (1.52e-3)
MaF12	10	19	3.1878e-1 (1.08e-1) –	8.6516e-1 (4.26e-2) +	7.5405e-1 (1.72e-2) –	8.7919e-1 (2.77e-2) +	8.0929e-1 (8.19e-3)
MaF13	10	5	7.9060e-2 (2.26e-2) –	9.9999e-2 (1.57e-2) –	1.2279e-1 (1.17e-2) +	4.1643e-5 (2.11e-4) –	1.1860e-1 (1.01e-2)
MaF14	10	200	5.4624e-1 (2.56e-1) +	0.0000e+0 (0.00e+0) –	6.3253e-3 (2.06e-2) =	1.1813e-2 (4.01e-2) =	2.9715e-3 (1.23e-2)
MaF15	10	200	5.2418e-7 (7.73e-7) –	0.0000e+0 (0.00e+0) –	0.0000e+0 (0.00e+0) –	2.8154e-6 (6.68e-6) –	1.1150e-5 (1.22e-5)
+ / = / –			3/0/12	4/2/9	5/1/9	5/2/8	

The standard deviation are provided in brackets. The best HV value for each case is highlighted in bold value. Symbols '+', '-', and '=' indicate that the performance is statistically better, worse, or approximately the same as that obtained by fSDE, respectively, according to the Wilcoxon rank sum test (0.05 significance level)

Table 6 Averaged HV values in solving MaF with 20 objectives

Problem	<i>m</i>	<i>d</i>	MOEA/ID	NSGA-III	Two_Arch2	IBEA	fSDE
MaF1	20	29	3.7199e-19 (1.17e-19) +	1.7599e-17 (8.02e-18) +	0.0000e+0 (0.00e+0) =	7.3058e-17 (2.56e-16) +	0.0000e+0 (0.00e+0)
MaF2	20	29	1.7914e-1 (1.28e-3) +	1.2281e-1 (2.25e-2) +	1.0588e-1 (1.36e-2) +	1.9912e-1 (7.45e-3) +	8.4386e-2 (1.13e-2)
MaF3	20	29	9.7226e-1 (8.75e-4) –	0.0000e+0 (0.00e+0) –	0.0000e+0 (0.00e+0) –	6.8881e-1 (3.39e-1) –	9.9979e-1 (2.39e-4)
MaF4	20	29	7.3557e-19 (1.81e-19) –	3.3180e-11 (2.77e-11) =	1.5048e-14 (8.24e-14) –	7.2428e-19 (1.00e-18) –	3.0494e-11 (2.84e-11)
MaF5	20	29	2.5470e-1 (8.71e-2) –	9.9377e-1 (7.57e-3) +	6.7339e-1 (1.46e-2) –	9.9823e-1 (1.83e-4) +	9.7986e-1 (3.75e-3)
MaF6	20	29	7.1232e-2 (3.92e-2) +	2.9238e-3 (1.02e-2) +	9.0811e-2 (8.68e-5) +	1.6181e-2 (2.12e-2) +	0.0000e+0 (0.00e+0)
MaF7	20	39	5.3891e-7 (1.64e-6) –	1.1693e-1 (1.20e-2) –	7.5390e-4 (1.36e-3) –	1.3680e-1 (1.18e-2) –	1.4155e-1 (1.83e-2)
MaF8	20	2	8.8723e-6 (1.33e-6) –	1.9342e-5 (3.91e-6) –	3.0177e-5 (4.31e-6) +	0.0000e+0 (0.00e+0) –	2.6384e-5 (3.69e-6)
MaF9	20	2	1.3473e-5 (6.00e-6) +	5.6707e-6 (1.04e-5) +	1.9137e-7 (3.30e-7) –	3.3746e-7 (5.07e-7) +	3.0521e-7 (1.67e-6)
MaF10	20	29	2.7858e-1 (6.61e-2) –	9.9884e-1 (6.85e-4) +	9.9415e-1 (7.14e-3) =	9.8920e-1 (2.64e-2) –	9.9605e-1 (1.01e-3)
MaF11	20	29	9.1971e-1 (1.80e-2) –	9.9359e-1 (3.50e-3) =	9.9724e-1 (8.44e-4) +	9.8722e-1 (3.18e-3) –	9.9395e-1 (1.40e-3)
MaF12	20	29	3.0787e-1 (2.33e-1) –	8.0835e-1 (4.56e-2) +	6.1613e-1 (3.08e-2) –	8.8147e-1 (1.66e-2) +	7.5430e-1 (5.59e-2)
MaF13	20	5	2.0461e-2 (1.41e-2) –	4.5304e-2 (1.14e-2) =	5.2085e-2 (1.32e-2) +	3.4603e-4 (1.89e-3) –	4.2698e-2 (1.16e-2)
MaF14	20	400	5.9554e-1 (1.95e-1) +	5.4690e-3 (2.09e-2) =	1.0344e-3 (5.67e-3) =	2.7519e-2 (4.55e-2) +	0.0000e+0 (0.00e+0)
MaF15	20	400	4.6563e-19 (1.55e-18) +	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	0.0000e+0 (0.00e+0) =	1.1395e-33 (6.24e-33)
+ / = / –			6/0/9	7/5/3	5/4/6	7/1/7	

The standard deviation are provided in brackets. The best HV value for each case is highlighted in bold value. Symbols '+', '+', and '=' indicate that the performance is statistically better, worse, or approximately the same as that obtained by fSDE, respectively, according to the Wilcoxon rank sum test (0.05 significance level)

Fig. 3 Averaged RMSE over 20 independent trials with respect to the simulation number of the computational model (left) and time (right)

Fig. 4 Averaged minmax error over 20 independent trials with respect to the simulation number of the computational model (left) and time (right)

measurements over operation points of feasible samples, according to the results of which the number of objectives is reduced to 7. Again, fSDE is applied to optimise the resulted 7-objective problem, denoted as fSDE-7. However, after dimensionality reduction via PCA, the percentage of infeasible settings sampled by fSDE (left of Fig. 5) increases along with the increased generation number, while without dimension reduction, the percentage of infeasible settings sampled by fSDE-15 (cf. Sect. 4.3.1) decreases, as shown in the right of Fig. 5. Due to the large number of sampled infeasible parameter settings, the fSDE fails and does not converge given more optimisation time. The dimensionality reduction via PCA loses the implicit information of parameter feasibility and infeasibility.

Comparison with baselines

As a supplementary verification, IBEA, MOEA/D, NSGA-III, and Two_Arch2 with the second dimensionality reduction method, i.e., case (i), have also been applied given the same number of simulations and the results are illustrated in

Figs. 3 and 4. Our proposed fSDE outperforms NSGA-III and Two_Arch2 in terms of solutions quality (RMSE) and outperforms MOEA/D in terms of real-time performance (right of Fig. 3) on our real aero-engine calibration problem.

Discussion

When modelling our aero-engine calibration as a many-objective optimisation problem, the usage of sophisticated dimensionality reduction method should be reconsidered. On the contrary, naive dimensionality reduction methods meet better the needs in engineering and lead to better results. Many-objective evolutionary algorithms are able to find diverse solutions of high quality. However, their computational cost is sometimes relatively high if an evaluation of solution is not feasible. As in our case, an evaluation takes only around 0.2 s and the optimisation efficiency is a more important indicator to be considered when selecting an optimisation algorithm. Our proposed fSDE with a simple

Table 7 Four aero-engine parameter settings with RMSE lower than 0.5% found by fSDE

Dimension	x^*	x_a	x_b	x_c	x_d
1	0.8550	0.8543 (0.08%)	0.8456 (1.10%)	0.8487 (0.74%)	0.8442 (1.26%)
2	3.2000	3.1994 (0.02%)	3.1997 (0.01%)	3.1996 (0.01%)	3.2002 (0.01%)
3	3.4000	3.3987 (0.04%)	3.3996 (0.01%)	3.3992 (0.02%)	3.3990 (0.03%)
4	0.8400	0.8396 (0.05%)	0.8398 (0.02%)	0.8397 (0.04%)	0.8400 (0.00%)
5	0.8200	0.8196 (0.05%)	0.8201 (0.01%)	0.8198 (0.02%)	0.8198 (0.02%)
6	0.8800	0.8895 (1.08%)	0.8729 (0.81%)	0.8638 (1.84%)	0.8549 (2.85%)
7	0.8900	0.8699 (2.26%)	0.8797 (1.16%)	0.8912 (0.13%)	0.8886 (0.16%)
8	0.9400	0.9404 (0.04%)	0.9402 (0.02%)	0.9400 (0.00%)	0.9401 (0.01%)
9	0.0150	0.0146 (2.67%)	0.0151 (0.67%)	0.0149 (0.67%)	0.0149 (0.67%)
10	1.0000	0.9803 (1.97%)	0.9642 (3.58%)	1.0048 (0.48%)	0.9878 (1.22%)
11	0.9800	0.9817 (0.17%)	0.9970 (1.73%)	0.9919 (1.21%)	0.9985 (1.89%)
12	0.9800	0.9932 (1.35%)	0.9824 (0.24%)	0.9847 (0.48%)	0.9836 (0.37%)
13	0.9995	0.9995 (0.00%)	0.9996 (0.01%)	0.9994 (0.01%)	0.9995 (0.00%)
14	0.0600	0.0387 (35.50%)	0.0567 (5.50%)	0.0641 (6.83%)	0.0894 (49.00%)
15	0.0250	0.0368 (47.20%)	0.0169 (32.40%)	0.0030 (88.00%)	0.0003 (98.80%)
16	0.0300	0.0343 (14.33%)	0.0379 (26.33%)	0.0360 (20.00%)	0.0390 (30.00%)
17	0.0200	0.0000 (100%)	0.0000 (100%)	0.0268 (34.00%)	0.0128 (36.00%)
18	0.0000	0.0374 (–)	0.1563 (–)	0.1570 (–)	0.3400 (–)
19	0.9900	0.9952 (0.53%)	0.9916 (0.16%)	0.9923 (0.23%)	0.9986 (0.87%)
20	1.0000	0.9967 (0.33%)	0.9973 (0.27%)	0.9976 (0.24%)	0.9913 (0.87%)
21	0.9600	0.9471 (1.34%)	0.9595 (0.05%)	0.9668 (0.71%)	0.9686 (0.90%)
22	0.9700	0.9746 (0.47%)	0.9768 (0.70%)	0.9739 (0.40%)	0.9711 (0.11%)
23	0.9900	0.9866 (0.34%)	0.9840 (0.61%)	0.9839 (0.62%)	0.9901 (0.01%)
24	0.9900	0.9803 (0.98%)	0.9864 (0.36%)	0.9900 (0.00%)	0.9890 (0.10%)
25	0.9900	0.9898 (0.02%)	0.9890 (0.10%)	0.9916 (0.16%)	0.9893 (0.07%)
26	0.5000	0.4880 (2.40%)	0.4969 (0.62%)	0.4966 (0.68%)	0.5052 (1.04%)
27	0.2400	0.2402 (0.08%)	0.2382 (0.75%)	0.2373 (1.12%)	0.2374 (1.08%)

The number in brackets indicates the error rate of the value on dimension i of the found solution x with respect to the optimum x^* , calculated as $|\frac{x_i - x_i^*}{x_i^*}|$

Fig. 5 Percentage of infeasible parameter settings sampled by fSDE-7 (left) and fSDE-15 (right) at every generation

framework outperforms the baseline algorithms in terms of real-time performance on the aero-engine calibration.

Single-objective evolutionary optimisation for aero-engine calibration

For a further verification of our approach, we model the calibration process as a single-objective optimisation problem with a linear aggregation of the distance between the target values and the simulated values of measurements given a setting. The fitness function is designed as follows:

$$fitness(\mathbf{x}) = 1 - RMSE(\mathbf{x}), \quad (4)$$

where $RMSE(\cdot)$ is defined in Eq. (4). A huge negative value is returned as fitness in case of invalid calculation or constraint violation in CFD calculations.

Failure of PSO, DE, and CMA-ES Although classic single-objective evolutionary algorithms can be applied directly to a such problem, two issues arise: (i) sampling infeasible settings leads to slow convergence due to the time consumed to determine the feasibility of sampled settings, and (ii) re-samplings due to sampled out-of-range settings lead to divergent values. The optimisation results by a classic PSO [13], a differential evolution (DE) [27], and a covariance matrix adaptation evolution strategy (CMA-ES) [11] shown in Figs. 6 and 7 and Table 8 illustrate examples of the above two issues. Therefore, particular operations should be taken for tackling the above issue.

The classic PSO and DE are implemented by us. The PSO uses $\lambda = 90$, $\omega = 0.5$, and $C_1 = C_2 = 0.5$, and the DE uses $\lambda = 90$, $C_r = 0.3$ and $F = 1$. The program of CMA-ES is taken from <https://github.com/AlexanderFabisch/CMA-ESpp>, and uses $\mu = \lambda = 13$, while the other parameters use the default setting. Each curve in Figs. 6 and 7 is an averaged result over 20 independent optimisation trials with 20,000 model simulations as budget.

Self-adaptive particle swarm optimisation (saPSO)

In the classic PSO, the location of particles ($\mathbf{x}$) is updated by the following equations:

$$v_i = \omega * v_i + C_1 * r_1 * (gbest - \mathbf{x}_i) + C_2 * r_2 * (pbest_i - \mathbf{x}_i), \quad (5)$$

$$\mathbf{x}_i = \mathbf{x}_i + v_i, \quad (6)$$

where $\mathbf{x}_i$ is the i^{th} individual, and r_1 and r_2 are two random vectors $\in (0, 1)^d$, where d is the dimension. Each coordinate of the velocity v_i is the sum of three items: inertia ($\omega * v_i$), the vector with random weight pointing to the global best

location of all particles found during optimisation ($C_1 * r_1 * (gbest - \mathbf{x}_i)$) and the vector with random weight pointing to the best location of this particle found during optimisation ($C_2 * r_2 * (pbest_i - \mathbf{x}_i)$).

To avoid the change of sampling distribution, the way to handle the out-of-range or infeasible solutions is re-sampling. However, when a particle is close to the boundary of feasible area in solution space and the inertia ($\omega * v_i$) points out to the boundary, the updated location ($\mathbf{x}_i$) is likely to be infeasible. Larger the inertia is, lower the possibility of sampling a feasible solution is. As a consequence, more computational resource will be consumed by re-sampling. In PSO, the waste of resource on re-sampling leads to a significant reduction of final solution's quality given limited optimisation budget.

We propose a self-adaptive PSO (saPSO) to handle this situation. In saPSO, for each re-sampling, ω is assigned as ω^2 . As $\omega \in (0, 1)$, the inertia is attractively reduced during re-sampling. Therefore, the possibility that the generated location is feasible is increasing. Algorithm 3 describes the process of our saPSO. The saPSO with the same parameters as PSO and $\omega = 0.9$ is performed on the engine calibration using the fitness defined in Eq. (4). Experimental results are shown in Figs. 6 and 7, as well as Table 8. It is observed that the saPSO converges as fast as CMA-ES, but does not lead to solution value far from the optimum on the 18th coordinate as the CMA-ES, PSO, and DE do. The extremely poor values found by CMA-ES, PSO, and DE on the 18th coordinate can be explained by the sampled out-of-range or infeasible parameter settings. When an out-of-range parameter setting is sampled, CMA-ES, PSO, and DE will re-sample a setting uniformly at random within the ranges given in Table 1. Although the particular design of saPSO successfully reduces the probability of sampling out-of-range or infeasible parameter settings, its final recommended parameter setting is not as good as the ones recommended by fSDE given the same amount of model simulations (cf. Fig. 8).

Discussion

Modelling the aero-engine calibration process as a single-objective optimisation problem is straightforward. However, due to the characteristics of our real-world calibration problem, the single-objective evolutionary algorithms are not efficient due to the sampling of out-of-range or infeasible parameter settings. Although we have proposed a simple self-adaptive PSO particularly to tackle this issue and obtained better performance comparing to some classic single-objective evolutionary algorithms, the efficiency of the self-adaptive PSO is still lower than the fSDE (cf. Fig. 8).

Table 8 Optimised aero-engine parameter settings by single-objective evolutionary algorithms

Dimension	$\mathbf{x}^*$	CMA-ES	PSO	DE	saPSO
1	0.8550	0.8520 (0.35%)	0.8391 (1.86%)	0.8697 (1.72%)	0.8521 (0.34%)
2	3.2000	3.1907 (0.29%)	3.2045 (0.14%)	3.1787 (0.67%)	3.1992 (0.02%)
3	3.4000	3.3918 (0.24%)	3.3505 (1.45%)	3.4325 (0.96%)	3.3996 (0.01%)
4	0.8400	0.8410 (0.12%)	0.8318 (0.98%)	0.8181 (2.61%)	0.8397 (0.04%)
5	0.8200	0.8217 (0.20%)	0.8194 (0.07%)	0.8392 (2.35%)	0.8199 (0.01%)
6	0.8800	0.8757 (0.49%)	0.8887 (0.99%)	0.8966 (1.89%)	0.8903 (1.17%)
7	0.8900	0.8734 (1.87%)	0.8854 (0.52%)	0.8948 (0.54%)	0.8883 (0.19%)
8	0.9400	0.9392 (0.09%)	0.9584 (1.96%)	0.9343 (0.61%)	0.9402 (0.02%)
9	0.0150	0.0143 (4.36%)	0.0144 (4.00%)	0.0211 (40.83%)	0.0151 (0.34%)
10	1.0000	0.9929 (0.71%)	1.0240 (2.40%)	1.0183 (1.83%)	1.0030 (0.30%)
11	0.9800	0.9801 (0.01%)	0.9878 (0.80%)	0.9923 (1.26%)	0.9851 (0.52%)
12	0.9800	0.9800 (0.00%)	0.9918 (1.21%)	0.9803 (0.03%)	0.9818 (0.18%)
13	0.9995	1.0000 (0.05%)	0.9980 (0.15%)	0.9986 (0.09%)	0.9996 (0.01%)
14	0.0600	0.0876 (46.08%)	0.0519 (13.54%)	0.0875 (45.85%)	0.0732 (22.07%)
15	0.0250	0.0471 (88.40%)	0.0262 (4.70%)	0.0322 (28.77%)	0.0527 (110.70%)
16	0.0300	0.0000 (99.90%)	0.0205 (31.52%)	0.0061 (79.61%)	0.0121 (59.63%)
17	0.0200	0.0001 (99.73%)	0.0294 (47.19%)	0.0374 (86.86%)	0.0056 (72.01%)
18	0.0000	25.0216 (–)	23.6440 (–)	6.7465 (–)	1.3001 (–)
19	0.9900	0.9986 (0.87%)	0.9956 (0.57%)	0.9966 (0.67%)	0.9944 (0.44%)
20	1.0000	1.0000 (0.00%)	0.9928 (0.72%)	0.9990 (0.10%)	0.9924 (0.76%)
21	0.9600	0.9719 (1.24%)	0.9551 (0.51%)	0.9261 (3.53%)	0.9528 (0.75%)
22	0.9700	0.9700 (0.00%)	0.9834 (1.38%)	0.9821 (1.25%)	0.9812 (1.15%)
23	0.9900	0.9800 (1.01%)	0.9907 (0.07%)	0.9830 (0.70%)	0.9836 (0.64%)
24	0.9900	0.9801 (1.00%)	0.9913 (0.13%)	0.9886 (0.14%)	0.9811 (0.90%)
25	0.9900	0.9998 (0.99%)	0.9922 (0.23%)	0.9948 (0.49%)	0.9823 (0.77%)
26	0.5000	0.4466 (10.67%)	0.5016 (0.33%)	0.5950 (19.00%)	0.5105 (2.11%)
27	0.2400	0.2495 (3.95%)	0.2576 (7.33%)	0.2644 (10.16%)	0.2572 (7.17%)

The number in brackets indicates the error rate of the value on dimension i of the found solution $\mathbf{x}$ with respect to the optimum $\mathbf{x}^*$, calculated as $\left| \frac{x_i - x_i^*}{x_i^*} \right|$

Fig. 6 Averaged RMSE over 20 independent trials with respect to the simulation number of the computational model (left) and time (right)

Fig. 7 Averaged minmax error over 20 independent trials with respect to the simulation number of the computational model (left) and time (right)

Fig. 8 Comparison among saPSO, CMA-ES, and fSDE. Figures on the bottom are the cropped version of the ones on top for better visualising the difference

Algorithm 3 Self-adaptive particle swarm optimisation (saPSO).

Input: Population size λ , ω , C_1 , C_2
Output: Final population

```

1:  $x_1, x_2, \dots, x_\lambda \leftarrow$  Randomly initialise  $\lambda$  individuals
2:  $v_1, v_2, \dots, v_\lambda \leftarrow$  Randomly initialise  $\lambda$  velocities
3:  $g_{best} = x_1$ 
4: for  $i = 1 : \lambda$  do
5:    $p_{best_i} = x_i$ 
6:   if  $fitness(x_i) > fitness(g_{best})$  then
7:      $g_{best} = x_i$ 
8:   end if
9: end for
10: while termination criteria not fulfilled do
11:   for  $i = 1 : \lambda$  do
12:      $\omega' = \omega$ 
13:     Sample two random vectors  $r_1$  and  $r_2 \in (0, 1)^d$ ,
14:   do
15:      $v'_i = \omega' v_i + C_1 * r_1 * (g_{best} - x_i) + C_2 * r_2 * (p_{best_i} - x_i)$ 
16:      $x'_i = x_i + v'_i$ 
17:      $\omega' = \omega' * \omega'$ 
18:     while  $x'_i$  is infeasible
19:     if  $fitness(x'_i) > fitness(p_{best_i})$  then
20:        $p_{best_i} = x'_i$ 
21:     end if
22:     if  $fitness(p_{best_i}) > fitness(g_{best})$  then
23:        $g_{best} = p_{best_i}$ 
24:     end if
25:      $x_i = x'_i$ 
26:      $v_i = v'_i$ 
27:   end for
28: end while
29: return  $x_1, x_2, \dots, x_\lambda$ 

```

This observation can be explained by the mechanics of fSDE. The crossover and mutation operators of fSDE will not lead to out-of-range samples.

Conclusion

In this paper, we consider a real aero-engine calibration problem with 27 parameters to be calibrated, 15 measurements at each operation point as quality criteria, and their values at 33 operation points as calibration data. The calibration process is modelled as a black-box optimisation problem aiming at minimising the distance between the simulated measurement values using a computational engine model and their calibration data. Both many- and single-objective evolutionary algorithms have been applied to calibrate the parameters. A fast many-objective evolutionary optimisation algorithm with shift-based density estimation, called fSDE, and a novel single-objective evolutionary algorithm, called saPSO, have been proposed and compared to several classic and state-of-the-art algorithms. The performance of fSDE has been assessed in terms of objective values, recommended decisions, and efficiency. Without parallelising the calculation or evaluation of population, the proposed fSDE

consumes ten times less time to find diverse parameter settings of high quality, compared to the time needed to find one single setting by an experienced human engineer using the traditional manual calibration process. fSDE not only provides diverse solutions but also finds solutions of high-quality faster, compared to some state-of-the-arts, thank to its simple operations.

Besides problem solving, our work also contributes in various ways. When finding different engine parameter settings with similar performance, human engineers may be assisted to diagnose the computational engine model and the measurements, and inspired to discover novel engine models. As a future work, we are interested in training a predictor to determine the feasibility of a sampled parameter setting to further reduce the required number of model simulations and improve the optimisation efficiency.

Funding This work was supported by the AECC, the National Natural Science Foundation of China (Grant Nos. 61906083, 61976111), the Guangdong Provincial Key Laboratory (Grant No. 2020B121201001), the Program for Guangdong Introducing Innovative and Entrepreneurial Teams (Grant No. 2017ZT07X386), the Shenzhen Science and Technology Program (Grant No. KQTD2016112514355531), and the Science and Technology Innovation Committee Foundation of Shenzhen (Grant No. JCYJ20190809121403553).

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Bader J, Zitzler E (2011) HypE: an algorithm for fast hypervolume-based many-objective optimization. *Evol Comput* 19(1):45–76
2. Cheng R, Jin Y, Olhofer M, Sendhoff B (2016) A reference vector guided evolutionary algorithm for many-objective optimization. *IEEE Trans Evol Comput* 20(5):773–791
3. Cheng R, Li M, Tian Y, Zhang X, Yang S, Jin Y, Yao X (2017) A benchmark test suite for evolutionary many-objective optimization. *Complex Intell Syst* 3(1):67–81
4. Conover WJ (1998) *Practical nonparametric statistics*, vol 350. Wiley, Amsterdam

5. Corne DW, Jerram NR, Knowles JD, Oates MJ (2001) PESA-II: Region-based selection in evolutionary multiobjective optimization. In: Proceedings of the 3rd annual conference on genetic and evolutionary computation, pp 283–290
6. Das I, Dennis JE (1998) Normal-boundary intersection: a new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM J Optim* 8(3):631–657
7. Deb K (2001) Multi-objective optimization using evolutionary algorithms, vol 16. Wiley, Amsterdam
8. Deb K, Goyal M (1996) A combined genetic adaptive search (geneas) for engineering design. *Comput Sci Inf* 26:30–45
9. Deb K, Jain H (2013) An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. *IEEE Trans Evol Comput* 18(4):577–601
10. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans Evol Comput* 6(2):182–197
11. Hansen N, Ostermeier A (1996) Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation. In: Proceedings of IEEE international conference on evolutionary computation, IEEE, pp 312–317
12. Ishibuchi H, Tsukamoto N, Nojima Y (2008) Evolutionary many-objective optimization: a short review. In: 2008 IEEE congress on evolutionary computation, IEEE, pp 2419–2426
13. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of IEEE international conference on neural networks, pp 1942–1948
14. Kurzke J, Halliwell I (2018) Propulsion and power: an exploration of gas turbine performance modeling. Springer, New York
15. Langouët H, Métivier L, Sinoquet D, Tran QH (2011) Engine calibration: multi-objective constrained optimization of engine maps. *Optim Eng* 12(3):407–424
16. Laumanns M, Thiele L, Deb K, Zitzler E (2002) Combining convergence and diversity in evolutionary multiobjective optimization. *Evol Comput* 10(3):263–282
17. Li B, Li J, Tang K, Yao X (2015) Many-objective evolutionary algorithms: a survey. *ACM Comput Surv (CSUR)* 48(1):1–35
18. Li B, Tang K, Li J, Yao X (2016) Stochastic ranking algorithm for many-objective optimization based on multiple indicators. *IEEE Trans Evol Comput* 20(6):924–938
19. Li M, Yao X (2019) Quality evaluation of solution sets in multiobjective optimisation: a survey. *ACM Comput Surv (CSUR)* 52(2):1–38
20. Li M, Yang S, Liu X (2013) Shift-based density estimation for pareto-based algorithms in many-objective optimization. *IEEE Trans Evol Comput* 18(3):348–365
21. Lygoe RJ, Cary M, Fleming PJ (2010) A many-objective optimisation decision-making process applied to automotive diesel engine calibration. In: Asia-Pacific conference on simulated evolution and learning, Springer, pp 638–646
22. Ma H, Li Z, Tayarani M, Lu G, Xu H, Yao X (2018) Computational intelligence nonmodel-based calibration approach for internal combustion engines. *J Dyn Syst Measur Control* 140(4)
23. Ma H, Li Z, Tayarani M, Lu G, Xu H, Yao X (2019) Model-based computational intelligence multi-objective optimization for gasoline direct injection engine calibration. *Proc Inst Mech Eng Part D: J Automob Eng* 233(6):1391–1402
24. Marler RT, Arora JS (2004) Survey of multi-objective optimization methods for engineering. *Struct Multidiscip Optim* 26(6):369–395
25. Rezapour K (2012) Exergy based SI engine model optimisation: exergy based simulation and modelling of bi-fuel si engine for optimisation of equivalence ratio and ignition time using artificial neural network (ANN) emulation and particle swarm optimisation (PSO). PhD thesis, University of Bradford
26. Samadani E, Shamekhi AH, Behrooz MH, Chini R (2009) A method for pre-calibration of di diesel engine emissions and performance using neural network and multi-objective genetic algorithm. *IJCCE* 28(4):61–70
27. Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *J Global Optim* 11(4):341–359
28. Tayarani-N MH, Yao X, Xu H (2014) Meta-heuristic algorithms in car engine design: a literature survey. *IEEE Trans Evol Comput* 19(5):609–629
29. Tian Y, Cheng R, Zhang X, Cheng F, Jin Y (2017a) An indicator-based multiobjective evolutionary algorithm with reference point adaptation for better versatility. *IEEE Trans Evol Comput* 22(4):609–622
30. Tian Y, Cheng R, Zhang X, Jin Y (2017b) PlatEMO: a MATLAB platform for evolutionary multi-objective optimization. *IEEE Comput Intell Mag* 12(4):73–87
31. Wang H, Jiao L, Yao X (2014) Two_arch2: an improved two-archive algorithm for many-objective optimization. *IEEE Trans Evol Comput* 19(4):524–541
32. Wong KI, Wong PK, Cheung CS, Vong CM (2013) Modeling and optimization of biodiesel engine performance using advanced machine learning methods. *Energy* 55:519–528
33. Yang S, Li M, Liu X, Zheng J (2013) A grid-based evolutionary algorithm for many-objective optimization. *IEEE Trans Evol Comput* 17(5):721–736
34. Zhang Q, Li H (2007) MOEA/D: a multiobjective evolutionary algorithm based on decomposition. *IEEE Trans Evol Comput* 11(6):712–731
35. Zitzler E, Künzli S (2004) Indicator-based selection in multiobjective search. In: International conference on parallel problem solving from nature, Springer, pp 832–842
36. Zitzler E, Laumanns M, Thiele L (2001) SPEA2: improving the strength pareto evolutionary algorithm. TIK-report 103

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.