Multi-Objective Particle Swarm Optimization for Key Quality Feature Selection in Complex Manufacturing Processes

An-Da Li[a,b,*], Bing Xue[c], Mengjie Zhang[c]

[a]*School of Management, Tianjin University of Commerce, Tianjin 300134, China*
[b]*Research Center for Management Innovation and Evaluation, Tianjin University of Commerce, Tianjin 300134, China*
[c]*School of Engineering and Computer Science, Victoria University of Wellington, Wellington 6140, New Zealand*

Abstract

In this paper, a feature selection (FS) method is proposed to identify key quality features (KQFs) in complex manufacturing processes. We propose a multi-objective binary particle swarm optimization algorithm, called MPBPSO, with three new components to optimize a bi-objective FS model of maximizing the geometric mean (GM) measure and minimizing the number of selected features. First, MPBPSO uses a modified probability-based solution update (PSU) mechanism which utilizes a flipping vector to update particles. A mutation operator with three basic operations, i.e., add, eliminate, and interchange, is also utilized in MPBPSO to improve the exploration performance. Second, a strategy combining the Pareto dominance concept with a distance measure is proposed for MPBPSO to update *pbest* (personal best position). Finally, a selection strategy based on the roulette wheel selection is proposed to determine the *gbest* (global best position) from the non-dominated set during iterations. The experimental results on four datasets have shown that the proposed FS method can identify a small number of KQFs that have good predictive ability for product quality. Further analysis indicates that MPBPSO obtains better search performance than eight benchmark optimization algorithms and the new components in MPBPSO are effective for improving its search performance.

Keywords: Particle swarm optimization, feature selection, multi-objective optimization, classification, quality control

1. Introduction

The complex manufacturing processes of modern industries generally contain numerous quality features (QFs) including part characteristics and process variables [21]. Identifying key QFs (KQFs) that significantly

*Please cite as: Li, A.-D., Xue, B., & Zhang, M. (2023). Multi-objective particle swarm optimization for key quality feature selection in complex manufacturing processes. Information Sciences, 641, 119062. https://doi.org/10.1016/j.ins.2023.119062

*Corresponding author
Email addresses: adli@tjcu.edu.cn (An-Da Li), Bing.Xue@ecs.vuw.ac.nz (Bing Xue), Mengjie.Zhang@ecs.vuw.ac.nz (Mengjie Zhang)

affect product quality can reduce the difficulty of applying quality control and improvement tools in complex manufacturing processes [19]. KQF identification can also improve the effectiveness and efficiency of quality prediction models [2], which is essential for accurately and timely adjusting the manufacturing processes to control process quality [29]. For example, the manufacturing process of a high-precision laser gyroscope of one company in China contains several stages. First, an optical cavity with qualified holes is manufactured based on the perforating operations. Second, an intermediate laser gyroscope is manufactured by assembling optical components in the cavity so that the laser gyroscope can generate qualified light sources. Finally, the final laser gyroscope is manufactured by further assembling electronic components in the intermediate laser gyroscope. A large number of QFs including performance/dimensional features of parts (i.e., part characteristics) and process parameters (e.g., the temperatures, air pressures, and voltages) at the three manufacturing stages are the potential factors affecting the quality of the final laser gyroscope. Due to the complexity of the laser gyroscope and its manufacturing process, there are complicated relationships between QFs in the manufacturing process and the final product quality. Identifying the KQFs strongly related to the final product quality is essential for effectively controlling the quality of laser gyroscopes. The wide application of Internet of Things (IoT) technologies to industries in recent years substantially simplifies the processes of data collection, data integration, and storage, which lays a foundation for the data-driven quality engineering techniques [43]. It is required to propose an effective KQF identification method based on the high dimensional manufacturing process data with numerous QFs.

Feature selection (FS) is a widely used technique for reducing data dimensionality by selecting the most critical features (variables) related to the class label (or response variable) to build an effective and compact machine learning model [9, 42]. In recent years, many FS-based KQF identification methods have been proposed due to their effectiveness in dealing with high dimensional data [2, 13, 19, 20, 21]. For example, Anzanello et al. [2] proposed an FS method that first uses a partial least squares regression based measure to rank features and then uses a multicriteria decision method to select the most important features for product quality prediction. Li et al. [20] proposed an FS method that uses an improved direct multi-search (IDMS) algorithm to search for the best feature subset with competent prediction ability for product quality. In these FS-based KQF identification applications, the QFs and product quality (e.g., conforming or nonconforming) are treated as the features and the class label in classification tasks. Similar to [20], we address the KQF identification problem based on the manufacturing process data of two discrete quality levels, which is related to a classification task in machine learning.

From the perspective of feature evaluation, FS methods can be categorized into the filter and wrapper approaches. The filters generally use a measure based on distance, information theory, or statistical theory to evaluate the importance of a feature or feature subset [44]. The wrappers introduce a learning model in the feature importance evaluation process where a classification performance measure is used to evaluate a feature subset [17]. Since the performance of a learning model is directly used to evaluate the feature

importance, wrappers can generally select features with better classification performance than that of the filters.

The manufacturing process data are unbalanced because the number of instances (products) of different classes (quality levels) differs substantially. It is usually seen that the manufactured regular quality products (majority class instances) are substantially more than the premium quality products (minority class instances), or the conforming products are substantially more than nonconforming products. Most FS methods [26, 32, 36, 39, 40] in literature adopt *accuracy* to measure the classification performance or the quality of a feature subset. However, accuracy without considering the data imbalance issue may not be an effective classification measure for the unbalanced data, and thus it can lead to a biased FS (KQF identification) result. To solve this problem, an FS model constructed as maximizing the geometric mean (GM) of the true positive rate (TPR) and true negative rate (TNR) and minimizing the number of selected features was proposed in [20] for KQF identification. Different from accuracy, GM is sensitive to both the minority and majority class instances, so that it is a desirable feature importance measure for unbalanced data.

From the perspective of optimization, a wrapper approach often models FS as a multi-objective optimization problem (MOP) of maximizing the classification performance and minimizing the number of selected features. This problem has shown to be an NP-hard problem that has a very large solution space when there are a large number of original features [1]. Various heuristic search algorithms, such as sequential forward selection (SFS) [17] and sequential backward selection (SBS) [17] have been applied to solve such an optimization problem. Moreover, evolutionary algorithms (EAs), such as genetic algorithms (GAs) [13, 23], differential evolution (DE) [8, 36], and particle swarm optimization (PSO) algorithms [32, 48], have been widely applied to FS due to their superior search performance. The single objective EAs can solve the multi-objective FS problem by aggregating the two objectives into one objective, which needs substantial domain knowledge to determine how to aggregate.

Compared with single objective EAs, multi-objective EAs (MOEAs) can solve a multi-objective FS problem more straightforwardly as they can simultaneously handle multiple objectives during the optimization process [39]. In MOEAs, the Pareto dominance concept is generally used to determine the quality of solutions. A solution a is said to dominate another solution b if 1) all the objective function values of a are not worse than that of b and 2) a is better than b on at least one objective function. The solutions that are not dominated by any other solution are said to be the non-dominated solutions, which are the output of the MOEAs. In recent years, many MOEAs including multi-objective GAs and multi-objective DE algorithms have been applied to FS. For example, Li et al. [21] proposed a hybrid algorithm that combines a GA with the direct multi-search (DMS) algorithm, where DMS aims to locally update the non-dominated solutions during the optimization process. The proposed approach has been shown to obtain good performance for KQF identification. Zhang et al. [47] proposed a multi-objective binary DE algorithm called MOFS-BDE

for FS. This algorithm applied a local search step called one-bit purifying to periodically update the non-dominated solutions evolved by the DE procedure. Although the aforementioned GA and DE algorithms applied different strategies to improve their search performance for solving FS problems, the convergence speeds of these EAs are still limited especially for FS in high-dimensional data with a large number of features. Except for GA and DE, PSO [15] is also one of the most popular EAs. PSO updates each particle (solutions) based on its own best experience (i.e., personnel best position, *pbest*) and the best experience of the swarm (i.e., global best position, *gbest*). The application of *gbest* substantially increases the convergence speed of PSO, making it a desirable optimization approach for FS (KQF identification) in high dimensional data. The multi-objective PSO algorithms [39, 25, 11] have been widely studied for FS problems in recent years. However, these studies have applied continuous PSO (CPSO) algorithms which are designed to search in a continuous decision space. A binary multi-objective PSO algorithm for more effective optimization of the FS problem is worth studying seeing that FS is a combinatorial optimization problem.

Recently, a binary PSO (BPSO) algorithm called probability BPSO (PBPSO) [40] that adopts a probability-based solution update (PSU) mechanism was proposed for FS. It has been shown to obtain competitive search performance for FS problems. The PSU mechanism maintains the basic concept of PSO. It generates a probability vector based on the difference between *pbest* and *gbest*, and then uses this vector to decide the flipping probability for each element/bit in the position. A constant p_0 is also used when calculating the probability vector. It causes an effect to randomly flip the elements in the particle position to enhance the exploration performance of PBPSO. However, the random search behavior caused by p_0 yields a negative effect that prevents the reduction of features during the late phase of the optimization (see Section 3.3.1 for details). This motivates us to design an improved PSU mechanism by considering the unique characteristics of FS, based on which we will then build a multi-objective PBPSO for FS. In many existing multi-objective PSO algorithms, the Pareto dominance concept is used to determine the update of *pbest* [25, 39] during the iterations. However, different from that in the single objective scenario, there is a much higher probability that the current position and *pbest* of a particle do not dominate each other in the multi-objective scenario, which can substantially reduce the frequency of updating *pbest*. This motivates us to build a more effective *pbset* update strategy for building a multi-objective PBPSO algorithm in this paper.

In this paper, we propose a wrapper-based FS method by developing a multi-objective PSO algorithm for identifying KQFs in complex manufacturing processes that have good predictive ability for product quality. The proposed FS method is referred to as MPBPSO-IPM, which is composed of two phases. In the first phase, a multi-objective PBPSO algorithm, called MPBPSO, is proposed to solve the KQF identification (FS) model [20] for unbalanced data to obtain a set of non-dominated solutions (QF subsets). A modified non-dominated sorting strategy [19] that can handle the duplicate solutions in the swarm is used in MPBPSO to sort the solutions and find the non-dominated solutions during iterations. In the second phase, the ideal point method (IPM) [19] is adopted to select the best compromise solution (KQF set) from the solutions

found by MPBPSO. The contributions of the proposed FS method are summarized as follows:

- We propose a modified PSU mechanism for MPBPSO to update particle positions during the evolutionary process. The modified PSU mechanism discards the random flipping term p_0 in the standard PSU mechanism of PBPSO [40]. To maintain the exploration performance of MPBPSO, a mutation operator considering the characteristics of FS is adopted to further update particles after the modified PSU mechanism. The proposed mutation operator adopts three basic operations, i.e., add, eliminate, and interchange, to update a feature subset (solution) with the same probability. The proposed mutation operator does not change the expected number of selected features in a particle so that the drawback of p_0 in the standard PSU mechanism that prevents the reduction of features during the late phase of the evolutionary process can be avoided.

- We propose a distance-based *pbest* update (DPU) strategy that combines the Pareto dominance concept with a distance measure to update *pbest* in MPBPSO. Specifically, the distance to the non-dominated front is used as the measure to determine the update of *pbest* when the current position and *pbest* do not dominate each other. Compared with the Pareto dominance concept used in existing multi-objective PSO algorithms [25, 39], the designed DPU strategy can determine the goodness of *pbest* and the current position in the case that they do not dominate each other. Therefore, the *pbest* can be updated more frequently and reasonably, which improves the search performance of the proposed MPBPSO algorithm.

- We propose a roulette wheel selection based *gbest* selection (RWGS) strategy in MPBPSO. This strategy guarantees that non-dominated solutions are uniformly selected as *gbest*. So, similar computation resources are allocated to different regions of the non-dominated front.

We verify the effectiveness of the proposed FS method MPBPSO-IPM on four unbalanced manufacturing process datasets. The experimental results show that MPBPSO-IPM outperforms eight benchmark FS methods in identifying KQFs. Further analysis verifies the effectiveness of the unique components added in MPBPSO, including the solution update mechanism, the DPU strategy, and the RWGS strategy. Moreover, the overall search performance of MPBPSO is verified by comparing MPBPSO with eight well-known multi-objective optimization algorithms.

The rest of this paper is organized as follows. Section 2 presents the background and related work, including the KQF identification (FS) model, IPM, PBPSO, and literature review. Section 3 presents the proposed multi-objective optimization algorithm MPBPSO. Section 4 describes the experimental design. Section 5 presents and analyzes the KQF identification results. Section 6 evaluates the search performance of MPBPSO. Section 7 presents conclusions and future research interests.

2. Background and related work

2.1. The multi-objective KQF identification (FS) model

Let $\mathcal{D} = \{(\mathbf{e}_i, y_i)\}$ ($i \in \{1, 2, ..., M\}$, $\mathbf{e}_i \in \mathbb{R}^N$, $y_i \in \{-1, 1\}$) be a dataset of M products (i.e., instances) and N QFs (i.e., features) collected from a complex manufacturing process, $\mathbb{F} = \{f_1, f_2, ..., f_N\}$ be the set of original QFs, $\mathbf{e}_i = (e_{i,1}, e_{i,2}, ..., e_{i,N})$ be the observations of the N QFs, and $y_i \in \{-1, +1\}$ be the quality level of product i, where $+1$ and -1 denote the minority class (e.g., premium quality) and majority class (e.g., regular quality), respectively. Then, KQF identification can be described as an FS problem that aims to select a QF subset (i.e., feature subset) $X \subseteq \mathbb{F}$ that has a competent predictive ability for product quality.

In [20], a KQF identification (i.e., FS) model considering the imbalance problem of the manufacturing process data was proposed. This model is defined as a multi-objective optimization problem of maximizing GM and minimizing the number of selected QFs, and it can be described as

$$\begin{aligned} \min \quad & F = \{1 - GM(X), |X|\} \\ \text{s.t.} \quad & X \subseteq \mathbb{F}, X \neq \emptyset \end{aligned} \tag{1}$$

where $GM(X)$ denotes the GM value obtained by X, and $|X|$ denotes the number of QFs selected by X. In Eq. (1), the first objective aims to maximize the classification performance of the selected QFs. The second objective aims to reduce the number of selected QFs, which can promote the KQF identification method to reduce as many irrelevant and redundant QFs as possible.

This KQF identification model uses GM as the measure to evaluate feature importance. The GM measure is defined as the geometric mean of TPR and TNR, i.e.,

$$GM = \sqrt{TPR * TNR} \tag{2}$$

where

$$TPN = \frac{TP}{TP + FN}, \tag{3}$$

$$TNR = \frac{TN}{TN + FP}. \tag{4}$$

In Eqs. (3) and (4), TP, TN, FP, and FN denote the number of true positives, true negatives, false positives, and false negatives respectively in the binary classification scenario. GM is a suitable measure for unbalanced data because a high GM value requires both high TPR and TNR values, which reflects the classification performance of the minority and majority classes, respectively. In comparison, a good classification result for the majority class only is enough to yield a high accuracy rate when the data is unbalanced. So, accuracy may bias evaluate the classification performance, especially for the minority class.

But in fact, the minority class is often the key issue [3]. For example, in the manufacturing processes, the minority defective/premium (i.e., minority class) products gain much more attention than the majority regular products for quality control purposes. The performance of GM for KQF identification on unbalanced data was justified in [21]. In this study, the performance of GM for building a KQF identification model was compared with accuracy and the F_1 score (for the minority class). The experimental results have shown that GM is the most effective measure for KQF identification on unbalanced manufacturing process data. Therefore, in this paper, we adopt GM to build the KQF identification model as shown in Eq. (1).

2.2. Ideal point method (IPM)

The KQF identification model shown in Eq. (1) defines a MOP. MOEAs can be used to optimize the defined MOP without further converting it into a single objective optimization problem, and a set of non-dominated solutions will be obtained. From a practical point of view, further selecting the best compromise solution from these non-dominated solutions is required. The ideal point method (IPM) [19], a widely used multi-objective decision method, can be used to achieve such a goal. In IPM, an ideal point (solution) in the objective space is first defined. Then, the solution closest to the ideal point is selected as the best compromise solution. In this paper, we apply IPM to select the final KQF set from the non-dominated solutions obtained by the proposed MPBPSO algorithm.

Let Ω be the non-dominated set found by a multi-objective optimization algorithm, m be the number of objective functions, and $f_i(X)$, $(i = 1, ..., m,\ X \in \Omega)$ be the ith objective function. The procedure of IPM can be illustrated as the following three steps:

1. The objective function values for each $X \in \Omega$ is normalized by the Z-score normalization method as

$$f_i^n(X) = (f_i(X) - \overline{f_i})/\sigma(f_i), \tag{5}$$

   where $\overline{f_i}$ and $\sigma(f_i)$ denotes the mean and standard deviation of the ith objective function calculated by the solutions in Ω.

2. The ideal point is defined based on the normalized objective function values as

$$[f_1^*, ..., f_m^*] = [min_{x\in\Omega}(f_1^n(X)), ..., min_{x\in\Omega}(f_m^n(X)]. \tag{6}$$

3. The best compromise solution X^* is obtained as

$$X^* = \arg\min_{x\in\Omega} \sqrt{\sum_{i=1}^{m}(f_i^n(X) - f_i^*)^2}, \tag{7}$$

which means the solution in Ω with the lowest Euclidean distance to the ideal point is selected as the best compromise solution.

2.3. Probability BPSO (PBPSO)

In [40] a BPSO algorithm called PBPSO was proposed for FS. PBPSO maintains the basic search concept of PSO that particles are updated based on *pbest* and *gbest*. The difference between PBPSO and the standard BPSO [15] is that PBPSO adopts a PSU mechanism that uses a flipping probability vector instead of velocity to update particles. This vector reflects the flipping probability for each element in the particle position.

Let $\mathbf{X}_i^t = (x_{i,1}^t, x_{i,2}^t, ..., x_{i,N}^t)$ $(x_{i,j}^t \in \{0,1\},\ j = 1,2,...,N)$ be the position of a particle in the swarm at the tth iteration. In PBPSO, a probability vector $\mathbf{P}_i = (p_{i,1}, p_{i,2}, .., p_{i,N})$ updates each $x_{i,j}^t$ $(j = 1,2,...,N)$ in $\mathbf{X}_i^t$ as

$$x_{i,j}^{t+1} = \begin{cases} 1 - x_{i,j}, & \text{if} \quad r < p_{i,j} \\ x_{i,j}^t, & \text{otherwise} \end{cases} \tag{8}$$

where r denotes a random value from the uniform distribution $U(0,1)$. The probability vector $\mathbf{P}_i$ is calculated at each iteration and each element $p_{i,j}$ $(j = 1,2,...,N)$ in $\mathbf{P}_i$ is obtained by

$$p_{i,j} = p_0 + p_{i,j}^p + p_{i,j}^g. \tag{9}$$

In Eq. (9), $p_{i,j}^p$ and $p_{i,j}^g$ are obtained by comparing $x_{i,j}^t$ with *pbest* and *gbest*. They are defined as

$$p_{i,j}^p = |x_{i,j}^t - pbest_{i,j}| \cdot p_1, \tag{10}$$

$$p_{i,j}^g = |x_{i,j}^t - gbest_j| \cdot p_2, \tag{11}$$

where $pbest_{i,j}$ and $gbest_j$ denote the jth element of *pbest* and *gbest*. According to Eqs. (10) and (11), $p_{i,j}^p$ and $p_{i,j}^g$ are set as p_1 and p_2 if the current particle position $x_{i,j}^t$ is different from *pbest* and *gbest*, otherwise, they are set as 0. Since, as shown in Eq. (9), $p_{i,j}^p$ and $p_{i,j}^g$ contribute to the flipping probability $p_{i,j}$, they reflect the influence of *pbest* and *gbest* on the change of the particle position. Except for $p_{i,j}^p$ and $p_{i,j}^g$, p_0 is also used for calculating the final value of $p_{i,j}$. This means that, an element in the particle position still has a small probability of p_0 to flip when both $p_{i,j}^p$ and $p_{i,j}^g$ equal 0. Thus, p_0 reflects the strength of random search in PBPSO. In PBPSO, p_0, p_1, and p_2 are three user-defined parameters and the sum of these three parameters is defined as 1, which guarantees the flipping probability $p_{i,j}$ does not exceed 1.

2.4. Related work on evolutionary FS methods

FS is an optimization problem with $2^N - 1$ possible feature subsets (solutions) given a dataset with N original features. Due to superior search capabilities, MOEAs have been increasingly applied to FS in order to find a feature subset with both good classification performance and a small size [42]. Li et al. [19] proposed a modified non-dominated sorting algorithm II (NSGA-II) to build an FS method for KQF identification. The proposed modified NSGA-II algorithm adopts a strategy in the non-dominated sorting process to handle redundant solutions in the population to improve population diversity, which improves the search performance of NSGA-II in solving FS problems. Wang et al. [35] proposed a multi-objective DE algorithm that simultaneously selects features and optimizes the structure of the extreme learning machines. With this algorithm, an ensemble learning algorithm is established for predicting silicon content in hot metal. Wang et al. [36] proposed a multi-objective DE algorithm that adopts a niching and global interaction mutation operator for generating diversified and high-quality solutions (feature subsets) during the iterations. A repairing mechanism that can reduce redundant solutions in the population is proposed to increase FS performance. Except for GAs and DE, PSO is one popular EA widely used in various optimization tasks. For its simplicity, high convergence speed, and good global search performance, PSO-based FS methods have been extensively studied in recent years. Specifically, many multi-objective PSO algorithms have been widely applied to FS in order to maximize classification performance and minimize the size of feature subsets simultaneously. Xue et al. [39] applied two multi-objective PSO algorithms to build FS methods, CMDPSOFS and NSPSOFS. CMDPSOFS is based on the CMDPSO algorithm which adopts the ideas of crowding, mutation, and dominance for multi-objective optimization. NSPSOFS is based on the NSPSO algorithm which adopts the idea of non-dominated sorting for multi-objective optimization. Nguyen et al. [25] proposed a multi-objective PSO algorithm called ISRPSO, which adopts a local search process to polish the non-dominated solutions obtained at each iteration. Han et al. [11] proposed a multi-objective PSO algorithm that adopts a penalty mechanism for archive preservation based on penalty boundary interaction (PBI) decomposition and an adaptive leading particle selection approach using feature information to improve the optimization performance for FS. These multi-objective FS methods are based on CPSO algorithms where the decision variables (i.e., particles) to be optimized are continuous. Therefore, each element in a particle is continuous and should be compared to a predefined threshold parameter to determine if the corresponding feature is selected (e.g., larger than the threshold) or not (e.g., smaller than or equal to the threshold) [24, 41]. Building a multi-objective BPSO algorithm that can effectively search in the discrete decision space of FS problems is required to be further studied. Moreover, the PSO algorithms proposed in [25] and [39] have adopted the Pareto dominance concept to update *pbest*. However, when a new particle position and *pbest* are not dominated by each other, the Pareto dominance concept loses efficacy in determining the goodness of the two positions. Although the PSO algorithm in [11] adopts the PBI approach to transform the values of multiple objective functions of a particle into a scalar value, which

is used to determine the update of *pbset*. However, the PBI approach requires a user-defined parameter to guarantee its performance, which requires extensive efforts for parameter tuning.

Data imbalance is also a problem faced in many machine learning tasks [3, 28]. In a classification task, a dataset is unbalanced when the number of instances belonging to different classes substantially differs. For example, in the product quality prediction scenario, the number of products of different quality levels is substantially different. Most wrapper-based FS methods adopt accuracy as a measure to evaluate feature importance. However, accuracy is not a good classification performance measure for unbalanced data, since its value is mainly decided by the majority class instances. The classification performance for the minority class cannot be well measured by accuracy on unbalanced data. To tackle this problem, many studies have adopted other classification performance measures, such as TPR and TNR (or type I and type II errors) [27], recall and precision [6], and Jaccard index [49], instead of accuracy for FS. However, adopting the above-mentioned measures, such as TPR and TNR, for unbalanced data instead of accuracy means one additional optimization objective is involved. The increase in the number of objectives affects the performance of an MOEA [12]. To cope with this problem, some studies adopted an aggregated classification measure (e.g., expected maximum profit (EMP) [18], and GM [20]) instead of TPR and TNR, recall and precision, etc., for feature subset evaluation in FS. For example, Kozodoi et al. [18] proposed a multi-objective FS approach that maximizes the EMP measure and minimizes the number of selected features for credit scoring. The EMP measure is defined by considering the profit and cost for correctly and incorrectly classifying instances. Zhang et al. [48] proposed a clustering guided PSO algorithm for FS. In this method, a modified F-measure combining the filling risk of missing values, precision, and recall was proposed to tackle the FS problem on unbalanced data with missing values. Li et al. [20, 21] adopted an FS model that maximizes GM and minimizes the number of selected features for KQF identification on unbalanced manufacturing process data. The experimental results have shown that the GM measure can effectively handle the unbalanced manufacturing process data for a KQF identification objective.

FS is naturally a combinatorial optimization problem, i.e., searching for different combinations of features in order to obtain the optimal feature subset. In an FS problem, a discrete variable can represent whether a feature is selected (e.g., using 1) or not (e.g., using 0). Therefore, it is more straightforward to represent a feature subset in BPSO than CPSO for an FS problem. The standard BPSO algorithm proposed in [15] uses a sigmoid function to transform the velocity to a real value in $[0, 1]$ which is the probability to set an element in the particle position as 1. However, as mentioned in [16], by introducing the sigmoid function in BPSO, the velocity of each particle shows an opposite effect compared with that in CPSO. That is, a small velocity value in BPSO promotes exploration while a large velocity value in CPSO promotes exploration. This effect limits the search performance of standard BPSO. Recently, Xue et al. [40] proposed a BPSO variant called PBPSO, which updates the position of a particle based on the probability vector generated by the PSU mechanism without using the sigmoid function. Later, Nguyen et al. [26] proposed a modified

PBPSO called sticky BPSO (SBPSO) which introduces a stickiness parameter in PBPSO. The stickiness parameter is used in SBPSO to mimic the momentum property of CPSO. Moreover, Li et al. [22] proposed a BPSO-based FS method based on the PSU mechanism used in SBPSO. This method adopts a mutual-information-based swarm initialization strategy and a bits-masking-based search space reduction strategy, which substantially increase the FS performance on high dimensional data.

Existing studies [22, 26, 40] have shown the effectiveness of the PSU mechanism in PBPSO for FS. However, these methods are all proposed for the single objective FS models. Therefore, proposing a multi-objective PBPSO algorithm for FS is worth studying. Several issues should be addressed when building a multi-objective PBPSO algorithm. First, the constant p_0 in PSU that reflects a random flipping probability for elements in a particle can weaken the feature reduction performance during the late optimization phase for FS problems (see Section 3.3.1 for details). Second, an effective strategy is required in the multi-objective PBPSO to properly update *pbest* considering that two solutions are more difficult to be compared in the multi-objective scenario than that in the single-objective scenario as they may be not dominated by each other. Therefore, in Section 3, we propose a multi-objective PBPSO called MPBPSO, which utilizes a modified PSU mechanism and a distance-based *pbest* update strategy to address the above-mentioned issues.

3. The proposed approach

In this section, we propose a multi-objective FS method called MPBPSO-IPM for KQF identification. MPBPSO-IPM first adopts MPBPSO to select a set of candidate feature subsets (i.e., QF subsets), and then, utilizes IPM (please refer to Section 2.2) to select the final feature subset (i.e., KQF set) from the candidate feature subsets. The overall procedure of MPBPSO and its main components are introduced below.

3.1. Overall procedure of MPBPSO

To solve the multi-objective KQF identification model defined in Eq. (1), we propose the MPBPSO algorithm which is established by extending PBPSO [40] to the multi-objective scenario. MPBPSO inherits the main concept of PBPSO that utilizes a probability vector obtained from the particle position, *pbest*, and *gbest* for solution update. However, different from PBPSO, the p_0 term in Eq. (9) that yields the random flipping behavior for particle positions is discarded in MPBPSO. Instead, a mutation operator with three basic operations (add, eliminate, and interchange) is adopted in MPBPSO. Moreover, we propose the DPU strategy which combines the Pareto dominance concept and a distance measure to update *pbest*. It can effectively determine the update of *pbest* in the case that the new particle position and *pbest* are not dominated by each other. In the multi-objective scenario, all the solutions in the non-dominated set can

be a candidate for the *gbest*. Therefore, we further propose the RWGS strategy to determine the *gbest* at each iteration. Compared with selecting the *gbest* randomly, this strategy guarantees the non-dominated solutions are uniformly selected as the *gbest*, which is beneficial for uniformly allocating computational resources to different regions of the non-dominated front.

Algorithm 1 shows the overall procedure of MPBPSO. First, the algorithm initializes a set $\mathcal{S}^0$ of particles, from which the non-dominated solutions are obtained and added to the set Ω. Then, the *gbest* determined by the RWGS strategy is used by the modified PSU mechanism to update the position of each particle from $\mathbf{X}_i$ to $\mathbf{X}_i^p$. Next, the mutation operator is used to update $\mathbf{X}_i^p$ to $\mathbf{X}_i^m$. After the positions of all particles are updated, we obtain an intermediate swarm $\mathcal{S}^{'}$ containing the updated positions and *pbest*s of particles. Next, similar to NSGA-II [5], the best K particles in the union $\mathcal{S}^u$ of the parent swarm $\mathcal{S}^t$ and intermediate swarm $\mathcal{S}^{'}$ are kept for the next iteration. Specifically, the particles in $\mathcal{S}^u$ are sorted (see Section 3.6) according to the objective function values of their current positions to determine the best K particles, which are added to the new swarm $\mathcal{S}^{t+1}$. After that, the *pbest* of each particle in $\mathcal{S}^{t+1}$ is updated with the DPU strategy, and the algorithm moves to the next iteration. Finally, we output the non-dominated set Ω when the algorithm reaches the termination criteria.

3.2. Solution representation

In MPBPSO, the position $\mathbf{X}_i = (x_{i,1}, x_{i,2}, ..., x_{i,N})$ of the ith particle is defined as a vector of N elements, where N is the number of original features (QFs). Each element $x_{i,j} \in \{0, 1\}$, $j = 1, 2, ..., N$ denotes if the jth feature is selected ($x_{i,j} = 1$) or eliminated ($x_{i,j} = 0$) by $\mathbf{X}_i$.

3.3. Proposed solution update mechanism

3.3.1. Limitation of the standard probability-based solution update (PSU) mechanism

Each element $p_{i,j}$ in the probability vector $\mathbf{P}_i$ defined in Eq. (9) reflects the flipping probability of each element in a particle position. It is composed of three parts, i.e., p_0, $p_{i,j}^p$, and $p_{i,j}^g$, reflecting the strengths of random search, moving towards *pbest*, and moving towards *gbest*, respectively. In PBPSO [40], p_0 is defined as a small constant that contributes to the flipping probability of each element in the particle position. Thus, p_0 has a similar effect to the mutation operator in GAs.

However, the p_0 term in Eq. (9) may limit the PSO algorithm's performance for effectively eliminating features for an FS problem, which contradicts the main objective of FS, i.e., selecting a small number of key features. Let $\mathbf{X}_i^t = (x_{i,1}^t, x_{i,2}^t, ..., x_{i,N}^t)$ ($x_{i,j}^t \in \{0, 1\}$, $j = 1, 2, ..., N$) be a particle in the swarm at the tth iteration, $\Gamma = \{j | x_{i,j}^t = 1, j = 1, 2, ..., N\}$ be the set of indexes for the elements of 1 in $\mathbf{X}_i^t$, and $\Theta = \{j | x_{i,j}^t = 0, j = 1, 2, ..., N\}$ be the set of indexes for the elements of 0 in $\mathbf{X}_i^t$. The p_0 term denotes the flipping probability for each element in $\mathbf{X}_i^t$. Thus, the expected number of elements changing from 1 to 0 is $\#r = |\Gamma| * p_0$, and the expected number of elements changing from 0 to 1 is $\#a = |\Theta| * p_0$, where $|\quad|$

Algorithm 1: Algorithmic procedure of MPBPSO.

Input: Swarm size K, maximum number of iterations T, learning rate p_l, mutation rate p_m;
Output: The non-dominated set Ω;

$\Omega \leftarrow \emptyset$, $t \leftarrow 0$; `/* Initialize the non-dominated set` Ω `and iteration counter` t `*/`
$\mathcal{S}^0 \leftarrow \{(\mathbf{X}_1^0, pbest_1), ..., (\mathbf{X}_K^0, pbest_K)\}$; `/* Initialize a swarm of` K `particles.` $\mathbf{X}_i^0$ `(`$i \in \{1, 2, ..., K\}$`) is the position (solution) of the` i`th particle and` $pbest_i$ `is its personal best position. */`
Evaluate the objective function values for each particle in $\mathcal{S}^0$ with Eq. (1);
$\Omega \leftarrow$ Find the non-dominated particle positions (solutions) from $\{\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_K\}$ obtained from swarm $\mathcal{S}^0$;
`/* Begin iterations. */`
while $t < T$ **do**
  $\mathcal{S}^{'} \leftarrow \mathcal{S}^t$;
  gbest $\leftarrow$ Select a solution from Ω using the *roulette wheel selection based gbest selection (RWGS)* strategy (see Section 3.5);
  for each $\mathbf{X}_i^t \in \mathcal{S}^t$ **do**
    `/* Update the position with the modified PSU mechanism (see Section 3.3.2). */`
    $\mathbf{P}_i \leftarrow$ Obtain the probability vector based on p_l, $\mathbf{X}_i^t$, *pbest*, and *gbest* with Eq. (12);
    $\mathbf{X}_i^p \leftarrow$ Update $\mathbf{X}_i^t$ with $\mathbf{P}_i$ using Eq. (13);
    `/* Update the position with the proposed mutation operator (see Section 3.3.3). */`
    if $rand() < p_m$ **then**
      $\mathbf{X}_i^m \leftarrow$ Perform the mutation operation on $\mathbf{X}_i^p$ using Eq. (14) ;
    end
    Evaluate the objective function values of $\mathbf{X}_i^m$ with Eq. (1);
  end
  $\mathcal{S}^u \leftarrow \mathcal{S}^t \cup \mathcal{S}^{'}$;
  $\mathcal{S}^u \leftarrow$ Sort particles in $\mathcal{S}^u$ according to the goodness of particle positions in descending order (see Section 3.6);
  $\mathcal{S}^{t+1} \leftarrow$ The first K particles in $\mathcal{S}^u$;
  $\Omega \leftarrow$ The non-dominated particle positions in $\mathcal{S}^u$;
  Update the *pbest* of each particle in $\mathcal{S}^{t+1}$ with the *distance-based pbest update* (DPU) strategy (see Section 3.4);
  $t \leftarrow t + 1$;
end
return The non-dominated set Ω;

denotes the number of elements in a set. This means that, if the particle contains more elements of 0 than that of 1 (i.e., $|\Theta| > |\Gamma|$), it is expected that the number of features selected by this particle will increase (i.e., $\#r < \#a$) by applying the flipping operation caused by p_0. So, p_0 causes an effect that prevents the elimination of features, which is not beneficial for feature reduction. To cope with the above-mentioned issue of p_0, we propose a solution update mechanism combining a modified PSU mechanism with a new mutation operator for MPBPSO, which is introduced below.

3.3.2. The modified PSU mechanism

In the modified PSU mechanism, we discard the p_0 term in Eq. (9) to obtain the flipping probability vector. Let $\mathbf{X}_i^t = (x_{i,1}^t, x_{i,2}^t, ..., x_{i,N}^t)$ be the position of a particle in the swarm at the tth iteration, and $pbest_i$ be the *pbest* of this particle. The flipping probability $\mathbf{P}_i = (p_{i,1}, p_{i,2}, ..., p_{i,N})$ can be obtained via comparing $\mathbf{X}_i$ with $pbest_i$ and *gbest*. Specifically, each $p_{i,j}$ $(j = 1, 2, ..., N)$ is calculated as

$$p_{i,j} = (\alpha \cdot |x_{i,j}^t - pbest_{i,j}| + \beta \cdot |x_{i,j}^t - gbest_j|) \cdot p_l, \tag{12}$$

where $pbest_{i,j}$ and $gbest_j$ denote the jth element in $pbest_i$ and *gbest*, respectively, p_l denotes the overall learning rate regarding *pbest* and *gbest*, and α and β ($\alpha+\beta = 1$) are two parameters controlling the influence of *pbest* and *gbest* on the flipping probability. Then, we can update the current particle position $\mathbf{X}_i$ to a new position $\mathbf{X}_i^p = (x_{i,1}^p, x_{i,2}^p, ..., x_{i,N}^p)$ with the probability vector $\mathbf{P}_i$. Each $x_{i,j}^p$ $(j = 1, 2, ..., N)$ is obtained as

$$x_{i,j}^p = \begin{cases} 1 - x_{i,j}^t, & \text{if} \quad r < p_{i,j} \\ x_{i,j}^t, & \text{otherwise} \end{cases} \tag{13}$$

where r denotes a random value in $[0, 1]$. The modified PSU mechanism in Eq. (12) is consistent with the core idea in PSO, i.e., a particle moves following the *pbest* and *gbest* with different learning rates (known as acceleration constants c_1 and c_2 in PSO). Compared with the standard PSU mechanism, since the p_0 term is not used in Eq.(12), the negative effect of preventing the reduction of features can be avoided in the modified PSU mechanism. Moreover, the *gbest* should have more influence on the moving of a particle than *pbest* [40]. Thus, we set $\alpha = 1/3$ and $\beta = 2/3$ (i.e., $\beta = 2\alpha$) for simplicity. The overall learning rate p_l will be further tuned with experiments.

3.3.3. Mutation operator

We propose a novel mutation operator to further update particles after the modified PSU mechanism to enhance the exploration ability of MPBPSO. Generally, three basic operations can be used to update a feature subset (solution) for an FS task [25]. They are "add a new feature (add operation), "eliminate a feature (eliminate operation)", and "add a feature and eliminate another feature (interchange operation)". The proposed mutation operator is designed using these three operations. The main idea of this mutation operator is one of the three operations is randomly chosen for updating the particle position each time. A detailed description of this mutation operator is given below.

We consider the particle $\mathbf{X}_i^p = (x_{i,1}^p, x_{i,2}^p, ..., x_{i,N}^p)$. Let $\Gamma = \{j | x_{i,j}^p = 1, j = 1, 2, ..., N\}$ and $\Theta = \{j | x_{i,j}^p = 0, j = 1, 2, ..., N\}$ be the two sets of indexes denoting the elements of 1 and 0 in $\mathbf{X}_i^p$. The mutation operator

updates $\mathbf{X}_i^p$ to $\mathbf{X}_i^m = (x_{i,1}^m, x_{i,2}^m, ..., x_{i,N}^m)$ with a mutation rate p_m as

$$\mathbf{X}_i^m = \begin{cases} add\_op(\mathbf{X}_i^p), & \text{if} \quad r < p_a \\ eliminate\_op(\mathbf{X}_i^p), & \text{if} \quad p_a \leq r < p_a + p_e \\ interchange\_op(\mathbf{X}_i^p), & \text{if} \quad p_a + p_e \leq r \leq p_a + p_e + p_i \end{cases} \tag{14}$$

where r is a random value in $[0, 1]$, and p_a, p_e, and p_i ($p_a + p_e + p_i = 1$) denote the probabilities for conducting the add (*add_op*), eliminate (*eliminate_op*) and interchange (*interchange_op*) operations, respectively. The three operations are defined as follows.

- ***Add operation*** randomly selects an index $\theta \in \Theta$ and updates $\mathbf{X}_i^p$ with the following equation:

$$x_{i,j}^m = \begin{cases} 1 - x_{i,j}^p, & \text{if} \quad j = \theta \\ x_{i,j}^p, & \text{otherwise} \end{cases}, j = 1, 2, ..., N. \tag{15}$$

- ***Eliminate operation*** randomly selects an index $\gamma \in \Gamma$ and updates $\mathbf{X}_i^p$ with the following equation:

$$x_{i,j}^m = \begin{cases} 1 - x_{i,j}^p, & \text{if} \quad j = \gamma \\ x_{i,j}^p, & \text{otherwise} \end{cases}, j = 1, 2, ..., N. \tag{16}$$

- ***Interchange operation*** randomly selects an index $\gamma \in \Gamma$ and an index $\theta \in \Theta$ and updates $\mathbf{X}_i^p$ with the following equation:

$$x_{i,j}^m = \begin{cases} 1 - x_{i,j}^p, & \text{if} \quad j \in \{\gamma, \theta\} \\ x_{i,j}^p, & \text{otherwise} \end{cases}, j = 1, 2, ..., N. \tag{17}$$

The p_a, p_e, and p_i in Eq. (14) are set as the same value of $1/3$, which means that the three basic operations have the same conduction probability. With this setting, the proposed mutation operator shows the following property:

Theorem 1. *Let $I(\mathbf{X})$ be the number of elements of 1 in the particle $\mathbf{X}$. The expected value of $I(\mathbf{X}^m)$ in the mutated particle $\mathbf{X}^m$ equals $I(\mathbf{X})$, i.e.,*

$$E[I(\mathbf{X}^m)] = I(\mathbf{X}). \tag{18}$$

Proof. For a solution $\mathbf{X}$, it is easily found that the add operation increases $I(\mathbf{X})$ by 1, the eliminate operation decreases $I(\mathbf{X})$ by 1, and the interchange operation does not change $I(\mathbf{X})$. Therefore, given

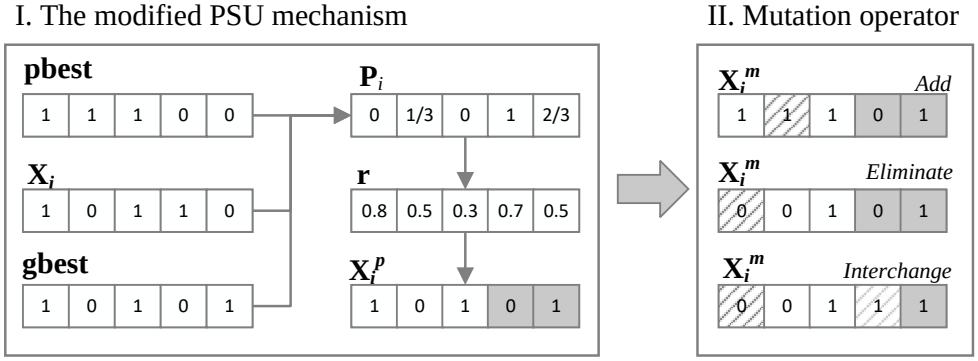


Fig. 1. Illustration of the proposed solution update mechanism combining the modified PSU mechanism with the mutation operator. For simplicity, we assume the length of the particle $N = 5$, $\alpha = 1/3$, $\beta = 2/3$ and $p_l = 1$.

$p_a = p_e = p_i$ and $p_a + p_e + p_i = 1$, the expected number of elements of 1 in $\mathbf{X}^m$ after the mutation operation is $E[I(\mathbf{X}^m)] = p_a \cdot (I(\mathbf{X}) + 1) + p_e \cdot (I(\mathbf{X}) - 1) + p_i \cdot I(\mathbf{X}) = I(\mathbf{X})$.

According to Theorem 1, the proposed mutation operator does not change the expected number of selected features for the particles in the swarm. This successfully addresses the issue of "preventing the elimination of features" caused by the p_0 term in the standard PSU mechanism of PBPSO as mentioned in Section 3.3.2.

Fig. 1 shows an illustration of the proposed solution update mechanism, which first utilizes the modified PSU mechanism and then uses the proposed mutation operator to update particle positions. The PSU mechanism inherits the basic idea of PSO that learns from the *pbest* and *gbest* to update particle positions, while the mutation operator guarantees the exploration performance of the algorithm. The mutation operator has an effect similar to p_0 in standard PSU. However, the p_0 term in the standard PSU mechanism applies the flipping operation with the same probability on each element in particles, which does not consider the impact on the change of the number of selected features for particles. Thus, it shows the negative effect of "preventing the elimination of features" in the late phase of the evolutionary process. In comparison, since the mutation operator considers the number of elements of 1 and 0 in particles, it maintains the number of selected features of the particles after the mutation, which is a good property for solving FS problems.

3.4. Distance-based pbest update (DPU) strategy

In PSO, the *pbest* of each particle needs to be updated if the new position (denoted by $\mathbf{X}^m$) of the particle has a better fitness than *pbest*. The Pareto dominance concept is an option to determine if the new particle position is better than *pbest* in MPBPSO. However, different from that in the single objective scenario, there is a much higher probability that $\mathbf{X}^m$ and *pbest* are not dominated by each other, i.e., equally good, in the multi-objective scenario. In this case, we are not able to differentiate between $\mathbf{X}^m$ and *pbest* based on the Pareto dominance concept. The overall chances for updating *pbest* will be considerably reduced if

we do not update *pbest* with $\mathbf{X}^m$ when *pbest* and $\mathbf{X}^m$ are not dominated by each other. This can affect the evolutionary performance of MPBPSO.

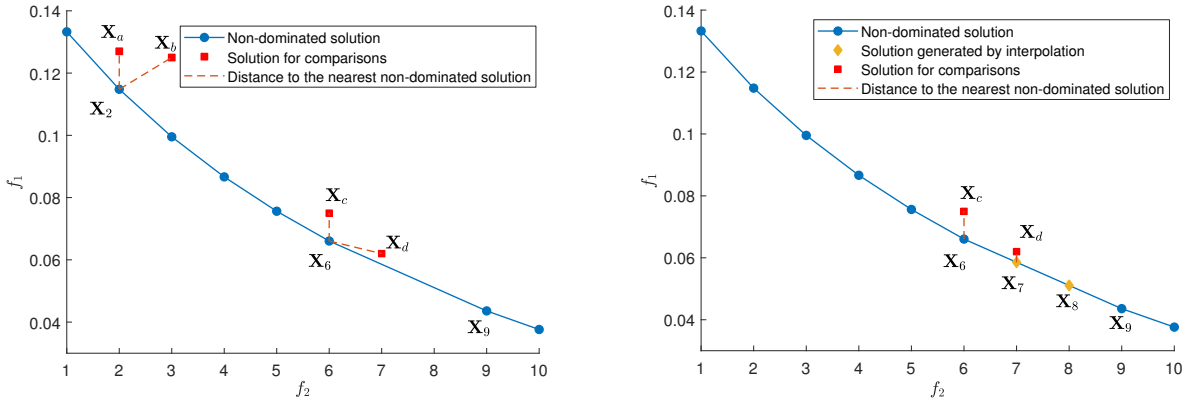
To cope with the above-mentioned issue, we propose a distance-based measure to compare two solutions that do not dominate each other. This measure is based on the idea of generational distance (GD) [34], which can determine the similarity between a solution and the non-dominated set. Given a solution $\mathbf{X}$, the GD value $GD(\mathbf{X})$ is calculated as its distance to the nearest non-dominated solution. For two solutions $\mathbf{X}_a$ and $\mathbf{X}_b$, we say $\mathbf{X}_a$ is better than $\mathbf{X}_b$ if $GD(\mathbf{X}_a) < GD(\mathbf{X}_b)$. Fig. 2a shows an illustration of the distance-based measure. We can find that the nearest non-dominated solution for $\mathbf{X}_a$ and $\mathbf{X}_b$ is the non-dominated solution $\mathbf{X}_2$. $\mathbf{X}_a$ is better than $\mathbf{X}_b$ because $\mathbf{X}_a$ is closer to $\mathbf{X}_2$. However, two issues should be addressed when using this distance measure to update *pbest*:

1. Directly using the GD measure to compare two solutions may lead to a biased comparison result. As shown in Fig. 2a, $\mathbf{X}_d$ is closer to the non-dominated front than $\mathbf{X}_c$. However, $\mathbf{X}_c$ is determined to be better than $\mathbf{X}_d$ because it has a shorter distance to the nearest non-dominated point $\mathbf{X}_6$ ("point" refers to a solution in the objective space). The reason for this result is that some non-dominated points are missing between $\mathbf{X}_6$ and $\mathbf{X}_9$, which yields a non-dominated front with unevenly spread non-dominated points.

   To deal with this problem, we add synthetic non-dominated points to the original non-dominated points to achieve a non-dominated front with uniformly spread points. Specifically, a *linear interpolation based solution generating* strategy is proposed in MPBPSO to generate the synthetic points. Suppose that the maximum and minimum values of objective f_2 (the number of selected features) obtained by the non-dominated set Ω are n_s and n_l respectively, which are two integer values. Then, the synthetic non-dominated points are generated as $\{(f_1, f_2) | f_2 \in \{n_s, n_s + 1, ..., n_l\} \quad and \quad f_2 \notin \mathbb{F}_2(\Omega)\}$, where $\mathbb{F}_2(\Omega)$ is the set of f_2 values obtained by Ω. The f_1 value of the new solution is obtained by linear interpolation based on the non-dominated solutions in Ω on its two sides. For example, in Fig. 2b, the two new points $\mathbf{X}_7$ and $\mathbf{X}_8$ are generated by the linear interpolation based on points $\mathbf{X}_6$ and $\mathbf{X}_9$. With the new non-dominated front after the interpolation, $\mathbf{X}_d$ is decided to be better than $\mathbf{X}_c$ because $\mathbf{X}_d$ has a shorter distance to the nearest non-dominated point than $\mathbf{X}_c$ dose.

2. The ranges of the two objectives in the KQF identification problem are quite different, this can affect the distance values used to compare two solutions. Therefore, before calculating the distance of a solution to the nearest non-dominated point, it is required to normalize the objective function values. For a solution $\mathbf{X}$, we use the min-max normalization method to obtain the normalized objective function values, i.e.,

$$f_i^n(\mathbf{X}) = f_i^{min} + (f_i(\mathbf{X}) - f_i^{min})/(f_i^{max} - f_i^{min}), \forall i \in \{1, 2\}, \tag{19}$$



(a) No interpolation. **(b)** With interpolation.

Fig. 2. Illustration of distance-based solution comparison mechanism.

Algorithm 2: Algorithmic procedure of the distance-based *pbest* update (DPU) strategy.

Input: A particle position $\mathbf{X}$, the *pbest*, the non-dominated set Ω;
Output: The updated *pbest*;

if $\mathbf{X} \prec pbest$ **then** `/* X dominates pbest */`
  $pbest \leftarrow \mathbf{X}$;
else if $pbest \not\preceq \mathbf{X}$ **then** `/* pbest do not dominate or equals X */`
  $\mathcal{F}_s \leftarrow$ Generate synthetic non-dominated points using the *linear interpolation based solution generating* strategy;
  $\mathcal{F} \leftarrow$ Combine the synthetic non-dominated points $\mathcal{F}_s$ with current non-dominated points in Ω;
  Obtain the normalized objective function values for $\mathbf{X}$, *pbest*, and each point in $\mathcal{F}$ with Eq. (19);
  $GD(\mathbf{X}) \leftarrow$ Calculate the distance between $\mathbf{X}$ and the nearest point in $\mathcal{F}$;
  $GD(pbest) \leftarrow$ Calculate the distance between *pbest* and the nearest point in $\mathcal{F}$;
  if $GD(\mathbf{X}) < GD(pbest)$ **then**
    $pbest \leftarrow \mathbf{X}$;
  end
return The *pbest* ;

where f_i^{max} and f_i^{min} denote the maximum and minimum values obtained by solutions in Ω.

Based on the above analysis, we develop the DPU strategy as shown in Algorithm 2. We apply two conditions to determine the update of *pbest*. The *pbest* is updated with $\mathbf{X}$ if $\mathbf{X}$ dominates *pbest*. Otherwise, the distance-based measure is used to compare $\mathbf{X}$ and *pbest* if they do not dominate each other. Specifically, we first generate a non-dominated front $\mathcal{F}$ with more uniformly spread points by the aforementioned linear interpolation based solution generating strategy. Then, we obtain the normalized objective function values for $\mathbf{X}$, *pbest*, and the points in $\mathcal{F}$, and obtain the distances between $X/pbest$ and the nearest points in $\mathcal{F}$. Finally, we update *pbest* with $\mathbf{X}$ if $\mathbf{X}$ has a smaller distance value than *pbest* does.

3.5. Roulette wheel selection based gbest selection (RWGS) strategy

At line 7 of Algorithm 1, a solution from the non-dominated set Ω is selected as the *gbest* for the current iteration. This *gbest* is used for updating particle positions by the modified PSU mechanism. Properly selecting the *gbest* during the iterations is essential to MPBPSO.

The solutions in Ω are spreading in different regions of the non-dominated front. A solution in Ω being selected as *gbest* means MPBPSO will allocate computational resources to search around it. To make the whole non-dominated front efficiently improve during the iterations, it is required to averagely allocate the limited computational resources to different regions of the non-dominated front. Therefore, we propose the RWGS strategy to uniformly select solutions in Ω as the *gbest* for MPBPSO.

In the RWGS strategy, a parameter φ is used to record the number of times being selected as the *gbest* for each solution $\mathbf{X}$. First, for a solution $\mathbf{X} \in \Omega$, the fitness value of selection is defined as

$$fit(\mathbf{X}) = 1/\exp(5\varphi(\mathbf{X})) \tag{20}$$

where $\varphi(\mathbf{X})$ denotes the φ value of $\mathbf{X}$. Then, the selection probability for each $\mathbf{X} \in \Omega$ is computed as

$$p(\mathbf{X}) = \frac{fit(\mathbf{X})}{\sum_{\mathbf{X}_i \in \Omega} fit(\mathbf{X}_i)}. \tag{21}$$

Finally, a solution $\mathbf{X} \in \Omega$ is selected as *gbest* with the probability $p(\mathbf{X})$. According to Eq. (20), if the difference of the φ values of two solutions is 1, the fitness of the solution with a smaller φ would be e^5 times (which is a very large value) to the fitness of the other solution. Therefore, the solutions with the smallest φ value have priority over other solutions being selected as the *gbest* as determined by Eq.(21). We can achieve the goal of uniformly selecting the solutions in Ω as *gbest* during the evolutionary process.

3.6. Sorting solutions

At line 17 of Algorithm 1, the particles in the union swarm $\mathcal{S}^u$ are sorted based on the goodness of particle positions in descending order. Hence, MPBPSO can maintain high-quality particles for the next iteration. In this paper, we adopt the idea of non-dominated sorting used in NSGA-II [5] to rank the particles in $\mathcal{S}^u$. Specifically, the fast non-dominated sorting strategy is first used to divide the solutions (positions of particles) in $\mathcal{S}^u$ into groups of different non-dominated front levels according to the Pareto dominance concept. A solution with a lower front level is said to have better quality. Then, the crowding distances for solutions at each front are calculated. The crowding distances can be used to further sort the solutions at the same front, and a greater crowding distance value denotes a worse quality of the solution. In [19], a modified non-dominated sorting strategy was proposed in a modified NSGA-II (MNSGA-II) for FS. This modified strategy further applies a process to detect and increase the front level of duplicate solutions at

each front after the standard non-dominated sorting process of NSGA-II. It has been shown to improve the performance of NSGA-II in FS tasks. Therefore, the modified sorting strategy in [19] is used in MPBPSO to sort solutions during the evolutionary process.

4. Experimental design

4.1. Datasets

Four real-world manufacturing process datasets, i.e., ADPN [7], LATEX [7], SPIRA [7], and PAPER [38], are used to verify the performance of the proposed MPBPSO-IPM method for KQF identification. ADPN was collected from the manufacturing process of adiponitrile. The manufacturing process is described by 100 input QFs (features), such as pressure, temperature, and flow, and a quality variable (response variable) "nickel loss" that reflects the product quality. LATEX was collected from the manufacturing process for latex. The manufacturing process is described by 117 input QFs, such as temperature, reactive concentration, monomer input rate, catalyst level, and a quality variable "insoluble products". SPIRA was collected from the fermentation process for manufacturing the antibiotic, i.e., Spiramycine. The manufacturing process is described by 96 input QFs, including the temperature level, stirring power, and oxygen consumption peaks, and a quality variable "content of Spiramycine". PAPER was collected from a paper recycling process. The manufacturing process is described by 54 QFs, including flow, concentration, and temperature measures at various time points, and a response variable "chemical oxygen demand load" evaluating the quality of wastewater from the recycling process. The original quality variables of these four datasets are continuous. In [2], the quality variables of these datasets were converted into discrete class labels with two levels according to the cut points provided by Gauchi and Chagnon [7] and Wold et al [38]. Such a conversion divides the products into two quality levels, i.e., premium (minority class) and regular (majority class), and thus classification models can be used. These converted datasets have been widely used in literature for verifying the performance of KQF identification methods [2, 19, 20]. The details of the four converted datasets used in the experiments are shown in Table 1.

Table 1: Details of the manufacturing process datasets.

Dataset	Number of instances	Number of features (QFs)	Number of minority class (premium quality) instances	Number of majority class (regular quality) instances
ADPN	71	100	20	51
LATEX	262	117	78	184
SPIRA	145	96	50	95
PAPER	384	54	33	351

4.2. Benchmark methods

We use eight benchmark FS methods to verify the proposed MPBPSO-IPM method. They are SFS [17], SBS [17], CMDPSOFS [39], NSPSOFS [39], NSGAII-IPM [19], MOFS-BDE [47], IDMS-IPM [20], and

GADMS-IPM [21]. SFS and SBS are two wrapper methods based on hill-climbing search strategies. SFS sequentially selects the informative features into the final feature subset, while SBS sequentially eliminates the non-informative features from the full feature set and the remaining features compose the final feature subset. CMDPSOFS, NSPSOFS, NSGAII-IPM, MOFS-BDE, IDMS-IPM, and GADMS-IPM are wrapper methods based on multi-objective optimization algorithms. CMDPSOFS and NSPSOFS are adapted from two multi-objective PSO algorithms, CMDPSO and NSPSO. NSGAII-IPM is an FS method based on MNSGA-II for KQF identification. MOFS-BDE applies a multi-objective binary DE algorithm to FS, where a local search step called one bit purifying was used to improve FS performance. IDMS-IPM utilizes a multi-objective optimization algorithm called IDMS to solve a KQF identification model for unbalanced production data. GADMS-IPM combines a GA with the DMS algorithm for KQF identification. The six multi-objective FS methods as well as our proposed MPBPSO-IPM method treat FS as a multi-objective optimization problem, i.e., to maximize the classification performance and minimize the number of selected features. The difference among these methods is that CMDPSOFS, NSPSOFS, and MOFS-BDE adopt the accuracy rate to measure classification performance while the other multi-objective FS methods utilize the GM measure to evaluate classification performance. It should be noted that NSGAII-IPM, IDMS-IPM, GADMS-IPM, and MPBPSO-IPM adopt IPM to select the best compromise solution (final KQF set) from the non-dominated solutions found by these multi-objective optimization algorithms, while CMDPSOFS, NSPSOFS, and MOFS-BDE do not provide a multi-objective decision method to determine the best compromise solution. For comparison purposes, we adopt IPM for CMDPSOFS, NSPSOFS, and MOFS-BDE to determine the final best compromise solution.

4.3. Parameter settings

In MPBPSO-IPM, the learning rate used in the modified PSU mechanism is set as $p_l = 0.60$ and the mutation rate is set as $p_m = 0.85$, which are determined with a set of parameter-tuning experiments (see Appendix A). The parameters in CMDPSOFS and NSPSOFS are set based on [39]. In CMDPSOFS, the inertia weight $w \in [0.1, 0.5]$ and mutation rate $p_m = 1/N$ (N is the number of original features). In NSPSOFS, the inertia weight $w = 0.7298$ and acceleration constants $c_1 = c_2 = 1.49618$ are used. CMDPSOFS and NSPSOFS use a real-coded particle $\mathbf{X} = (x_1, x_2, ..., x_N)$ to represent a feature subset. A threshold parameter $\theta = 0.6$ is used in CMDPSO and NSPSOFS to determine if the jth ($j = 1, 2, ..., N$) feature is selected ($x_j > \theta$) or not ($x_j \leq \theta$). In NSGAII-IPM, the crossover rate and mutation rate are set as $p_c = 0.9$ and $p_m = 1/N$ as suggested in [19]. In MOFS-BDE, the scale factor is set as $F = 0.5 \times rand$, the basic crossover probability is set as $CR = 0.4$, the parameter defining the frequency of the local search is $T_{loc} = 5$, and the turbulence coefficient is $\sigma = 0.01$ as suggested in [47]. In IDMS-IPM, the parameters controlling the search behavior are set as $\alpha^0 = 2$, $\beta_1 = \beta_2 = 0.95$ and $\gamma = 1$, $p_m = 1/N$ as suggested in [20]. In GADMS-IPM, the crossover and mutation rates used in the GA process are $p_c = 0.9$ and $p_{m1} = 1/N$,

and the two parameters used in the DMS process are $\beta = 0.9$ and $\alpha_0 = 1$ as suggested in [21]. In MPBPSO-IPM, CMDPSOFS, NSPSOFS, and NSGAII-IPM, the swarm/population size and the maximum number of iterations are set as $K = 100$ and $T = 100$, which results in 10,000 function evaluations. Moreover, the population size of MOFS-BDE and GADMS-IPM is set as 100 and the stopping criterion in MOFS-BDE, GADMS-IPM, and IDMS-IPM is set as $10,000$ function evaluations, which makes a fair comparison. In SFS and SBS, default settings in the Waikato Environment for Knowledge Analysis (Weka) [10] package are utilized.

4.4. Experimental configuration

Fig. 3 illustrates the experimental configuration for verifying the FS methods, which includes dataset division, FS, and test phases. First, 10-fold stratified cross validation (CV) [30] is used to divide the original dataset into training and test sets. Then, the training set is fed to the FS method to select features (KQFs). Finally, the training set with selected features (by eliminating other features) can be further used to establish a learning model. The model's classification performance on the test set shows the FS effectiveness. It is worth noting that the 10-fold CV generates 10 pairs of the training and test sets by dividing the dataset into 10 folds with equal sample sizes. For each pair, one fold is used as the test set and the other nine folds are combined as the training set. Therefore, the experiment to evaluate an FS method repeats 10 times in a 10-fold CV process. The FS methods except for SFS and SBS are based on stochastic search strategies. To comprehensively verify the performance of these methods, 3 repetitions of the 10-fold CV are implemented as suggested in [20], which means 30 (10×3) repetitions/runs of the experiment for each method on each dataset are implemented. We adopt the Wilcoxon signed-rank test [37] to compare MPBPSO-IPM with each benchmark method. Specifically, the 30 experimental repetitions obtain 30 observations of a performance metric for a method, which are used to compare the FS performance between two methods. Furthermore, being wrapper-based FS methods, MPBPSO-IPM and the benchmark methods described in Section 4.2 require a method to evaluate the goodness (i.e., classification performance) of a feature subset during the FS phase. In this paper, we apply 5-fold CV for feature subset evaluation to avoid FS bias [17]. Different from the above 10-fold CV, the 5-fold CV is an inner part of the FS method. It aims to further divide the training set fed to the FS method into 5 folds and generate 5 pairs of training and test sets, which are called inner-training and validation sets respectively in this paper for clarification. Then, 5 evaluation processes are performed with the 5 pairs of inner-training and validation sets to evaluate a feature subset. In each evaluation process, a learning model is first built with the inner-training set (only the features in the feature subset are used), and then, the classification results of the learning model on the validation set can be obtained. Finally, we can obtain the average value of a classification performance measure (e.g., GM defined in Eq. (1)) over the 5 evaluation processes to estimate the goodness of the feature subset. Moreover, we adopt a caching strategy to improve the time efficiency of MPBPSO as that used in [20]. This strategy

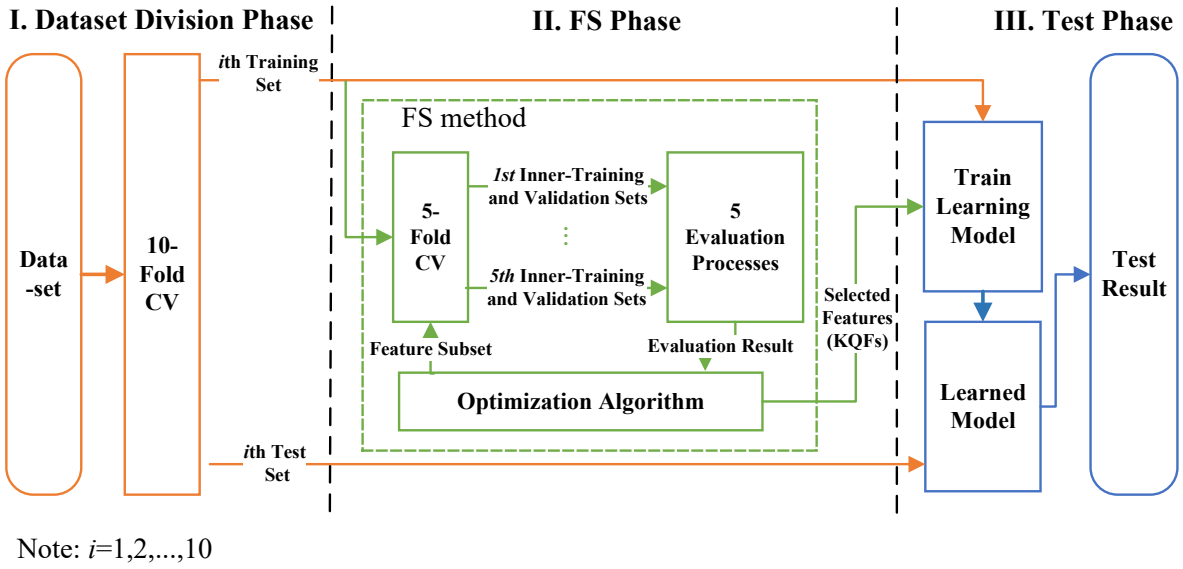


Fig. 3. Illustration of the experimental configuration.

uses a cache to store the objective function values of solutions. For a "new" solution generated during the iterations, we obtain the objective function values by the cache if this solution has been reached previously by the algorithm, otherwise, we adopt the inner 5-fold CV mentioned above to obtain the objective function values of this solution.

The experiments are implemented on a PC with a 3.6 GHz CPU and 16 GB RAM. The naive Bayesian (NB) classifier [14], which uses the kernel density estimator to estimate the distribution of continuous variables, is used as the learning algorithm in the training and test processes for the FS methods for its conciseness and effectiveness. A comparison between NB and other typical learning algorithms, including K-nearest neighbors (KNN), decision trees, and logistic regression classifiers, can be found in the supplementary material of this paper. The multi-objective FS methods including MPBPSO, CMDPSOFS, NSPSOFS, NSGAII-IPM, MOFS-BDE IDMS-IPM, and GADMS-IPM are implemented in the Matlab platform. The NB classifier, SFS, and SBS are implemented in Weka 3.7 [10]. The learning algorithm (i.e., NB classifier) used in these multi-objective FS methods is invoked from Weka 3.7. In the NB classifier, the option "-K, using kernel density estimator" is used. Note that the PAPER dataset is much more unbalanced than other datasets as the proportion of the number of majority class instances (denoted by $\#majIns$) to the number of minority class instances $\#minIns$ on PAPER is over 10 (351/33). This high imbalance rate may lead to the trained NB classifier being biased. To address this problem, we adopt a modified NB classifier for PAPER as suggested in [21]. This modified classifier uses an up-sampling strategy to balance the training set by duplicating the minority class instances $[\#majIns/\#minIns - 1]$ times. The balanced training set is then employed to train the original NB classifier.

5. Analysis of KQF identification performance

In this section, the KQF identification performance of MPBPSO-IPM is compared with that of SFS/SBS and benchmark multi-objective FS methods. The performance metrics including accuracy, TPR, TNR, GM, the macro-averaged F_1 score (F_1), and the number of selected KQFs (#KQFs) are used to evaluate the KQF identification performance. Accuracy, TPR, TNR, GM, and the macro-averaged F_1 score are typical classification performance measures. They range $[0, 1]$, the larger the better, reflecting the classification performance of selected KQFs. Specifically, the macro-averaged F_1 score is the average value of the F_1 scores for the minority and majority classes, where the F_1 score is defined as the harmonic mean of precision and recall in terms of a class. In the following text, F_1 refers to the macro-averaged F_1 score unless otherwise stated. The number of selected KQFs (smaller is better) denotes the reduction capability for irrelevant or redundant QFs of the FS methods. Moreover, the computational time of the FS methods is compared.

5.1. Comparison between MPBPSO-IPM and SFS/SBS

Table 2 shows the results of the KQF identification performance metrics obtained by MPBPSO-IPM, SFS, and SBS. Note that, for MPBPSO-IPM, the performance metric results of the best compromise solution selected by IPM from the non-dominated solutions returned by MPBPSO are recorded in the table. In the table, the mean and standard deviation (mean $\pm$ standard deviation) results of these performance metrics over the 30 repetitions of experiments are shown. The method that obtains the best mean value of each performance metric is highlighted in bold for each dataset. We use the Wilcoxon signed-rank test to compare MPBPSO-IPM with SFS/SBS. The "$\Uparrow$" ("$\uparrow$") or "$\Downarrow$" ("$\downarrow$") denotes MPBPSO-IPM obtains a significantly better or worse result than SFS/SBS at a significance level of $\alpha = 0.05$ ($\alpha = 0.1$).

Compared with SFS, MPBPSO-IPM obtains significantly better KQF identification performance on the datasets except for SPIRA. Specifically, MPBPSO-IPM obtains significantly better results on most classification performance metrics (including ACC, TPR, GM, and F_1) on ADPN, LATEX, and PAPER. Meanwhile, on the three datasets, MPBPSO-IPM generally obtains TNR values similar to SFS, and it selects significantly fewer KQFs than SFS. On SPIRA, it is shown that MPBPSO-IPM obtains significantly lower ACC, TNR, GM, and F_1 values than SFS, which indicates that the KQFs selected by SFS can better predict the quality of the products in the test set than MPBPSO-IPM.

Compared with SBS, MPBPSO-IPM obtains better KQF identification performance on all the four datasets. Specifically, MPBPSO-IPM obtains significantly better ACC, TPR, GM, and F_1 results than SBS at the significance level of $\alpha = 0.05$ or $\alpha = 0.1$ on ADPN and SPIRA. MPBPSO-IPM also obtains significantly better TPR and GM results than SBS at $\alpha = 0.05$ or $\alpha = 0.1$ on LATEX and PAPER. No results indicate MPBPSO-IPM obtains significantly worse classification performance than SBS. These results reveal that the KQFs selected by MPBPSO-IPM have a better predictive ability than SBS. From the perspective of

Table 2: Comparison of KQF identification performance between MPBPSO-IPM and SFS/SBS.

Dataset	Metric	MPBPSO-IPM	SFS	SBS
ADPN	ACC (%)	$\mathbf{83.83 \pm 8.00}$	77.32 ± 13.22 ↑↑	75.71 ± 14.36 ↑↑
	TPR (%)	$\mathbf{88.33 \pm 21.15}$	75.00 ± 25.00 ↑↑	65.00 ± 32.02 ↑↑
	TNR (%)	$\mathbf{82.00 \pm 13.65}$	78.00 ± 20.88	80.00 ± 12.65
	GM (%)	$\mathbf{83.52 \pm 10.05}$	73.78 ± 12.54 ↑↑	67.51 ± 27.25 ↑↑
	F_1 (%)	$\mathbf{81.44 \pm 8.49}$	73.87 ± 12.94 ↑↑	70.52 ± 18.27 ↑↑
	#KQFs	$\mathbf{2.2 \pm 0.4}$	5.3 ± 1.3 ↑↑	25.8 ± 9.9 ↑↑
LATEX	ACC (%)	$\mathbf{79.82 \pm 7.94}$	78.62 ± 6.72	76.75 ± 8.72
	TPR (%)	$\mathbf{73.15 \pm 16.26}$	49.82 ± 17.79 ↑↑	63.93 ± 15.23 ↑↑
	TNR (%)	82.59 ± 10.06	$\mathbf{90.70 \pm 5.63}$ ↓↓	82.13 ± 9.36
	GM (%)	$\mathbf{76.90 \pm 10.00}$	66.12 ± 12.12 ↑↑	71.92 ± 10.54 ↑
	F_1 (%)	$\mathbf{76.63 \pm 9.12}$	71.27 ± 9.40 ↑↑	72.63 ± 10.10
	#KQFs	$\mathbf{4.0 \pm 0.7}$	7.7 ± 2.9 ↑↑	93.8 ± 15.8 ↑↑
SPIRA	ACC (%)	78.48 ± 10.99	$\mathbf{82.81 \pm 5.38}$ ↓↓	73.38 ± 11.53 ↑↑
	TPR (%)	70.67 ± 16.92	$\mathbf{74.00 \pm 15.62}$	62.00 ± 24.41 ↑
	TNR (%)	82.70 ± 14.51	$\mathbf{87.56 \pm 8.90}$ ↓	79.22 ± 11.32
	GM (%)	75.55 ± 11.60	$\mathbf{79.72 \pm 7.00}$ ↓	66.09 ± 23.91 ↑
	F_1 (%)	76.27 ± 11.50	$\mathbf{80.62 \pm 5.86}$ ↓	69.70 ± 15.25 ↑↑
	#KQFs	$\mathbf{3.5 \pm 0.7}$	3.5 ± 1.0	51.7 ± 16.5 ↑↑
PAPER	ACC (%)	87.94 ± 4.11	82.06 ± 7.40 ↑↑	$\mathbf{87.99 \pm 5.59}$
	TPR (%)	$\mathbf{91.11 \pm 15.20}$	58.33 ± 30.28 ↑↑	80.83 ± 20.43 ↑↑
	TNR (%)	87.65 ± 4.07	84.36 ± 7.60	$\mathbf{88.88 \pm 7.17}$
	GM (%)	$\mathbf{89.00 \pm 8.38}$	65.36 ± 25.44 ↑↑	83.68 ± 9.58 ↑↑
	F_1 (%)	$\mathbf{75.10 \pm 7.66}$	63.29 ± 10.88 ↑↑	73.95 ± 7.72
	#KQFs	$\mathbf{2.9 \pm 0.4}$	4.0 ± 1.8 ↑↑	18.2 ± 9.8 ↑↑

feature reduction, MPBPSO-IPM selects a substantially smaller number of KQFs than SBS. On the ADPN, LATEX, and SPIRA datasets with more than 100 original QFs, MPBPSO-IPM selects less than 10% KQFs than that of SBS.

Overall, MPBPSO-IPM generally obtains significantly better classification performance while selecting a smaller number of KQFs than SFS and SBS on the tested manufacturing process datasets, showing that MPBPSO-IPM obtains better KQF identification performance than SFS and SBS.

5.2. Comparison between MPBPSO-IPM and benchmark multi-objective FS methods

Table 3 shows the results of the KQF identification performance metrics obtained by MPBPSO-IPM and benchmark multi-objective FS methods. Note that, for all these multi-objective methods, the performance metric results of the final solution selected by IPM from the non-dominated solutions found by the multi-objective optimization algorithms are recorded in the table for comparisons. The mean and standard deviation results over the 30 repetitions of experiments on each performance metric are shown for each dataset, and the Wilcoxon signed-rank test is used to compare MPBPSO-IPM with each of the benchmark methods. The notations used in Table 3 are the same as that in Table 2.

Table 3: Comparison of KQF identification performance between MPBPSO-IPM and the benchmark multi-objective FS methods.

Dataset	Metric	MPBPSO-IPM	CMDPSOFS	NSPSOFS	NSGAII-IPM	MOFS-BDE	IDMS-IPM	GADMS-IPM
ADPN	ACC (%)	**83.83 ± 8.00**	79.13 ± 12.54	79.87 ± 11.14	79.21 ± 15.16 ↑	82.01 ± 8.68	75.56 ± 8.53 ⇈	82.00 ± 8.13
	TPR (%)	**88.33 ± 21.15**	66.25 ± 32.21 ⇈	63.89 ± 32.01 ⇈	73.08 ± 28.54 ⇈	69.11 ± 28.68 ⇈	65.28 ± 29.67 ⇈	83.33 ± 23.57
	TNR (%)	82.00 ± 13.65	84.07 ± 15.44	85.98 ± 13.26	81.66 ± 16.45	**86.89 ± 11.34** ⇊	79.43 ± 13.56 ↑	81.44 ± 13.30
	GM (%)	**83.52 ± 10.05**	68.61 ± 26.74 ⇈	67.72 ± 26.50 ⇈	73.52 ± 23.14 ⇈	72.79 ± 20.84 ⇈	66.80 ± 21.39 ⇈	80.59 ± 10.99
	F_1 (%)	**81.44 ± 8.49**	73.17 ± 16.84 ⇈	73.49 ± 15.94 ↑	75.07 ± 17.12 ↑	76.35 ± 12.62	69.51 ± 11.88 ⇈	79.05 ± 8.92
	#KQFs	**2.2 ± 0.4**	6.8 ± 2.1 ⇈	3.6 ± 1.2 ⇈	3.1 ± 0.9 ⇈	2.7 ± 0.5 ⇈	14.1 ± 5.2 ⇈	2.2 ± 0.4
LATEX	ACC (%)	79.82 ± 7.94	80.24 ± 8.53	76.75 ± 7.91 ↑	79.98 ± 6.74	**80.87 ± 6.46**	78.11 ± 7.97	80.32 ± 7.18
	TPR (%)	73.15 ± 16.26	65.60 ± 21.11 ↑	55.62 ± 21.48 ⇈	65.48 ± 21.63 ↑	65.33 ± 21.22	70.36 ± 14.69	**74.35 ± 18.72**
	TNR (%)	82.59 ± 10.06	86.37 ± 9.39 ⇊	85.58 ± 8.49	85.91 ± 7.69 ⇊	**87.23 ± 8.58** ⇊	81.35 ± 9.44	82.81 ± 8.94
	GM (%)	76.90 ± 10.00	74.00 ± 12.98	67.34 ± 13.84 ⇈	73.54 ± 12.44	73.75 ± 12.88	75.10 ± 9.30	**77.39 ± 10.91**
	F_1 (%)	76.63 ± 9.12	75.68 ± 10.88	70.52 ± 10.88 ⇈	75.18 ± 9.43	75.91 ± 9.52	74.70 ± 8.96	**76.99 ± 8.99**
	#KQFs	4.0 ± 0.7	10.9 ± 2.4 ⇈	5.0 ± 1.5 ⇈	5.8 ± 1.9 ⇈	**4.0 ± 0.7**	29.9 ± 4.9 ⇈	4.2 ± 0.7
SPIRA	ACC (%)	**78.48 ± 10.99**	78.34 ± 8.02	73.96 ± 10.96 ⇈	73.95 ± 8.57 ⇈	74.73 ± 8.40 ↑	75.18 ± 10.82	77.95 ± 9.67
	TPR (%)	70.67 ± 16.92	**71.22 ± 18.04**	55.78 ± 23.08 ⇈	61.00 ± 19.89 ⇈	61.89 ± 19.10 ⇈	68.27 ± 21.27	68.00 ± 13.27
	TNR (%)	82.70 ± 14.51	82.15 ± 11.57	**83.56 ± 12.13**	80.78 ± 13.06	81.61 ± 11.21	78.81 ± 10.95	83.30 ± 12.58
	GM (%)	**75.55 ± 11.60**	75.33 ± 9.58	64.75 ± 21.10 ⇈	68.54 ± 11.07 ⇈	68.86 ± 15.54 ⇈	72.26 ± 13.55	74.73 ± 9.49
	F_1 (%)	**76.27 ± 11.50**	75.94 ± 8.66	69.22 ± 14.14 ⇈	70.40 ± 9.51 ⇈	71.19 ± 10.41 ⇈	72.66 ± 12.21	75.65 ± 9.81
	#KQFs	3.5 ± 0.7	6.3 ± 1.6 ⇈	3.7 ± 1.2	4.1 ± 1.0 ⇈	**3.3 ± 0.8**	16.0 ± 4.8 ⇈	3.8 ± 0.9 ↑
PAPER	ACC (%)	87.94 ± 4.11	88.56 ± 3.60	**89.79 ± 4.01** ⇊	89.48 ± 4.10 ⇊	89.32 ± 4.89 ↓	87.51 ± 5.04	87.68 ± 4.18
	TPR (%)	**91.11 ± 15.20**	69.91 ± 25.98 ⇈	68.19 ± 24.02 ⇈	68.47 ± 25.31 ⇈	65.94 ± 26.46 ⇈	88.06 ± 17.17	90.00 ± 14.34
	TNR (%)	87.65 ± 4.07	90.37 ± 3.87 ⇊	**91.92 ± 4.28** ⇊	91.49 ± 4.01 ⇊	91.56 ± 4.91 ⇊	87.46 ± 5.26	87.46 ± 4.27
	GM (%)	**89.00 ± 8.38**	76.01 ± 20.68 ⇈	76.75 ± 18.47 ⇈	75.81 ± 22.54 ⇈	73.30 ± 25.59 ⇈	87.21 ± 9.76	88.40 ± 7.74
	F_1 (%)	**75.10 ± 7.66**	71.78 ± 8.89	73.80 ± 9.91	73.41 ± 10.42	73.07 ± 11.77	74.48 ± 9.46	74.67 ± 7.52
	#KQFs	2.9 ± 0.4	4.3 ± 1.3 ⇈	5.7 ± 2.0 ⇈	4.8 ± 1.1 ⇈	5.1 ± 0.7 ⇈	3.1 ± 0.9	**2.8 ± 0.5**

5.2.1. Comparison with CMDPSOFS, NSPSOFS, MOFS-BDE, and NSGAII-IPM

Different from MPBPSO-IPM, CMDPSOFS, NSPSOFS, NSGAII-IPM, and MOFS-BDE adopt a KQF identification model that maximizes the accuracy and minimizes the number of selected features/QFs. According to Table 3, we can obtain the following results comparing MPBPSO-IPM with these benchmark methods.

First, MPBPSO-IPM generally obtains better classification performance than the four benchmark methods on all the four datasets. Specifically, MPBPSO-IPM obtains ACC results better than or similar to the four benchmark methods on the datasets except for PAPER. Moreover, MPBPSO-IPM generally obtains significantly better TPR, GM, or F_1 values than NSPSOFS and NSGAII-IPM on the four datasets, obtains significantly better TPR values than CMDPSOFS on the datasets except for SPIRA, and obtains significantly better TPR and GM values than MOFS-BDE on the datasets except for LATEX. This denotes that MPBPSO-IPM performs better than or similar to the four benchmark methods on classification performance metrics except for TNR in most cases. In terms of TNR, MPBPSO-IPM obtains significantly worse results than the four benchmark methods in some cases. However, these benchmark methods generally obtain substantially worse TPR results than the TNR results. This denotes that the performance of these benchmark methods is affected by data imbalance as they tend to classify many more products as regular quality products (i.e., the majority class), which reduces the classification performance for premium quality products (i.e., the minority class).

Second, in terms of the $\#KQF$ metric, MPBPSO-IPM selects significantly fewer KQFs than CMDPSOFS and NSGAII-IPM on all the four datasets, selects significantly fewer KQFs than NSPSOFS on three of the

four datasets, and selects significantly fewer KQFs than MOFS-BDE on two of the four datasets. No results indicate that MPBPSO-IPM selects significantly more KQFs than the four benchmark methods. This indicates that MPBPSO-IPM has a better feature reduction capability than these benchmark methods. Overall, the above analysis indicates that MPBPSO-IPM generally obtains better classification performance while selecting a smaller number of KQFs than the four benchmark methods, showing that MPBPSO-IPM outperforms these methods in KQF identification.

5.2.2. Comparison with IDMS-IPM

According to Table 3, we can find the following results comparing MPBPSO-IPM with IDMS-IPM. First, MPBPSO-IPM generally obtains better classification performance than IDMS-IPM. Specifically, MPBPSO-IPM obtains significantly better ACC, TPR, GM, and F_1 values on ADPN at the significance level of $\alpha = 0.05$, and obtains a significantly better TNR value on ADPN at $\alpha = 0.1$. In other cases, even though no evidence shows the difference of comparison results between the two methods is statistically significant, MPBPSO-IPM still shows better performance since the mean results on the five classification performance metrics obtained by it are better than IDMS-IPM.

Second, MPBPSO-IPM has better feature reduction performance than IDMS-IPM as MPBPSO-IPM obtains a significantly smaller number of KQFs on ADPN, LATEX, and SPIRA. On the three datasets, IDMS-IPM obtains more than 10 KQFs, while MPBPSO-IPM obtains fewer than 5 KQFs. The above results show that MPBPSO-IPM has a better KQF identification performance than IDMS-IPM, as MPBPSO-IPM can select fewer KQFs while obtaining better classification performance.

5.2.3. Comparison with GADMS-IPM

According to Table 3, we can obtain the following results comparing MPBPSO-IPM with GADMS-IPM. First, MPBPSO-IPM obtains slightly better classification performance than GADMS-IPM in most cases. Specifically, MPBPSO-IPM generally obtains better mean ACC, TPR, TNR, GM, and F_1 values than GADM-IPM on ADPN and PAPER, and obtains better mean ACC, TPR, GM, and F_1 values than GADMS-IPM on SPIRA. On LATEX, MPBPSO-IPM obtains slightly lower mean ACC, TPR, TNR, GM, and F_1 values than GADMS-IPM. Second, in terms of the #KQF metric, MPBPSO-IPM and GADMS-IPM generally select similar numbers of KQFs on the ADPN, LATEX, and PAPER datasets. On SPIRA, MPBPSO-IPM obtains significantly fewer KQFs than GADMS-IPM at the significance level of $\alpha = 0.1$.

It is worth noting that, the test results comparing MPBPSO-IPM with GADMS-IPM are not significantly different. This reveals that the overall KQF identification performance of the two methods is similar. However, as MPBPSO-IPM obtains better mean results on the classification performance metrics and similar #KQFs results compared to GADMS-IPM on three of the four datasets (i.e., ADPN, SPIRA, and PAPER), it is shown that MPBPSO-IPM slightly outperforms GADMS-IPM. In Section 6, we will compare the search

performance of the optimization algorithms, i.e., MPBPSO and GADMS, used in these two methods to further verify the proposed method.

5.3. Computational time

Fig. 4 shows the computational time of each FS method on the four datasets. In the figure, the average computational time consumed by each method over the 30 repetitions of experiments is shown for each dataset. Note that, we also record the computational time of a new version of MBPSO-IPM (denoted by MPBPSO-IPM-NC), which does not use the caching strategy to improve time efficiency. Thus, the effectiveness of the caching strategy can be analyzed. The following findings can be obtained from Fig. 4:

- First, the caching strategy is effective for improving the time efficiency because the computational time of MPBPSO-IPM is less than that of MPBPSO-IPM-NC on each dataset. The difference is not huge, which is probably because the size of the feature subsets selected by MPBPSO-IPM is small. So, the feature subsets are fast to evaluate, making the reduction of the computational time relatively small.

- Second, SFS consumes the least time and SBS consumes the most time on each dataset. The efficiency of SFS is because the forward search scheme keeps the feature subsets (solutions) being evaluated during the optimization process to be with small sizes. Seeing that the feature subset evaluation time in a wrapper approach increases with the increase of the feature subset size, the overall evaluation time in SFS can be substantially saved. Since SBS uses a backward search scheme and the sizes of feature subsets to be evaluated during the optimization process are large, SBS requires a large amount of time for evaluating feature subsets which increases the overall computational time.

- Third, the computational time of MPBPSO-IPM on each dataset is not significantly different from the benchmark multi-objective FS methods, i.e., CMDPSOFS, NSPSOFS, NSGAII-IPM, MOFS-BDE, IDMS-IPM, and GADMS-IPM, which shows that these methods have similar time complexities.

5.4. Summary

The comparisons in Sections 5.1 - 5.3 have shown that MPBPSO-IPM is effective and efficient for KQF identification. Summarizing these comparison results, we obtain the following findings.

First, the multi-objective FS methods generally perform more effectively in reducing the number of KQFs (features) than the conventional SFS and SBS. According to the #KQFs results in Tables 2 and 3, the multi-objective FS methods (except for IDMS-IPM) generally obtain a smaller number of KQFs than SFS and SBS. The following possible reasons can explain this result. On one hand, the multi-objective FS methods tend to optimize two objectives: a) maximizing the feature subset importance (measured by

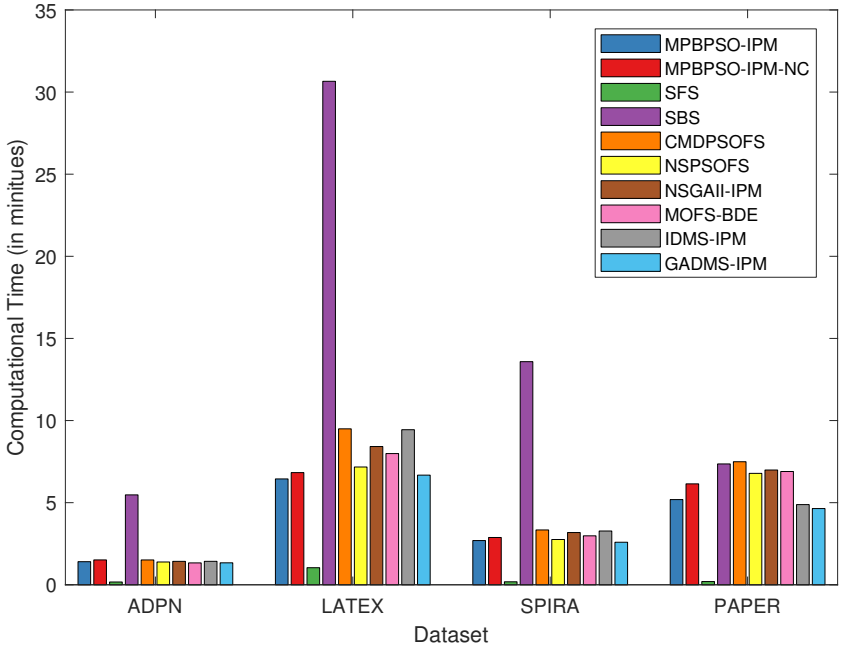


Fig. 4. Comparison of CPU time (in minutes) consumed by FS methods.

classification performance metrics) and b) minimizing the number of selected features. By adopting the second optimization objective, the performance for reducing the number of selected features is improved for these multi-objective FS methods. On the other hand, SFS and SBS are actually based on hill-climbing search strategies. From the optimization perspective, the optimization performance of hill-climbing search strategies is limited, especially when the search space is very large with many local optima. Thus, SFS and SBS cannot obtain feature subsets with both good classification performance and a small sizes. Note that, although IDMS-IPM and MPBPSO-IPM adopt the same multi-objective FS model, IDMS-IPM selects more KQFs than SFS. The reason is that IDMS-IPM evolves too slowly for obtaining good enough FS results with the given number of function evaluations (see Section 6 for the detailed analysis of the search performance of IDMS).

Second, the multi-objective FS (KQF identification) model adopted by MPBPSO-IPM performs effectively for identifying KQFs on unbalanced manufacturing process data. MPBPSO-IPM, IDMS-IPM, and GADMS-IPM adopt the same FS model, which adopts GM to evaluate the classification performance of the feature subsets. In comparison, CMDPSOFS, NSPSOFS, MOFS-BDE, and NSGAII-IPM adopt accuracy to evaluate feature subsets in the FS model. The comparisons between the methods adopting the two types of FS models reveal that the GM model can select KQFs with the better overall performance for predicting the quality of products (especially for the premium quality products) in the test set than the accuracy model. This is because GM is a better measure for evaluating the classification performance of different feature

subsets for unbalanced data than accuracy.

Third, MPBPSO-IPM obtains better KQF identification results than the benchmark FS methods (i.e., IDMS-IPM and GADMS-IPM) with the same FS model. MPBPSO-IPM generally obtains better mean results of the classification performance metrics than both IDMS-IPM and GADMS-IPM. Moreover, MPBPSO-IPM selects a substantially smaller number of KQFs than IDMS-IPM on ADPN, LATEX, and SPIRA. Because IDMS-IPM, GADMS-IPM, and MPBPSO-IPM adopt the same FS model, the difference in KQF identification performance for these methods is due to the search performance of the adopted optimization algorithms. In other words, a possible reason that MPBPSO-IPM obtains better final KQF identification results is that the optimization algorithm used by MPBPSO-IPM has better search performance than that of IDMS-IPM and GADMS-IPM. In Section 6, we will further analyze the search performance of these multi-objective optimization algorithms used in the FS methods.

Finally, MPBPSO-IPM consumes a similar time to the benchmark multi-objective FS methods. This is because all the multi-objective wrapper-based FS methods evaluate the objective functions for the same 10,000 times, which leads to similar overall function evaluation time in these methods since function evaluation is the most time-consuming part in a wrapper method. Moreover, the caching strategy that stores the already evaluated objective functions to save evaluation time is effective. This strategy can be considered a trick for saving computational time in wrapper-based FS methods.

6. Further analysis of the search performance of MPBPSO

The comparison results in the previous section have shown that MPBPSO-IPM performs effectively for KQF identification on unbalanced manufacturing process data. As analyzed, one reason for the effectiveness of MPBPSO-IPM is that the adopted FS model is effective. To comprehensively verify the effectiveness of MPBPSO-IPM, it is required to further analyze the search performance of MPBPSO-IPM. All the multi-objective FS methods including MPBPSO-IPM adopt a two-phase optimization framework where a multi-objective optimization algorithm is used first for finding a set of candidate KQF sets and then IPM is used for selecting the final KQF set. Therefore, the optimization algorithm adopted in each FS method is a key factor influencing the final KQF identification results. In this section, the search performance of MPBPSO, the optimization algorithm used in MPBPSO-IPM, will be verified.

6.1. Experimental setting

Eight benchmark multi-objective optimization algorithms are used. The first six benchmark algorithms are CMDPSO, NSPSO, MNSGA-II, MOBDE (multi-objective binary DE), IDMS, and GADMS, which are the multi-objective optimization algorithms used in CMDPSOFS, NSPSOFS, NSGAII-IPM, MOFS-BDE, IDMS-IPM, and GADMS-IPM. The last two benchmark algorithms are MOEA/D [46] and SPEA2 [50],

typical MOEAs widely used in many optimization tasks. The search performance of MPBPSO can be directly compared with IDMS and GADMS based on the first phase results of MPBPSO-IPM, IDMS-IPM, and GADMS-IPM from the 30 repetitions of experiments designed in Section 4.4 since they optimize the same FS model. To compare CMDPSO/NSPSO/MNSGA-II/MOBDE with MPBPSO, we further conduct experiments by applying CMDPSO, NSPSO, MNSGA-II, and MOBDE to the same FS model as MPBPSO shown in Eq. (1). Similarly, we conduct experiments for MOEA/D and SPEA2 by solving the FS model in Eq. (1) with these two algorithms. The configurations of the additional experiments for CMDPSO, NSPSO, MNSGA-II, MOBDE, MOEA/D, and SPEA2 are the same as that described in Section 4.4, i.e., we repeat 10-fold CV three times which results in 30 repetitions of the experiments. The parameter settings of CMDPSO, NSPSO, MNSGA-II, and MOBDE are the same as that described in Section 4.3. Similar to MNSGA-II, the population size, the maximum number of iterations, crossover rate, and mutation rate are set as 100, 100, 0.9, and $1/N$ in MOEA/D and SPEA2. Moreover, in MOEA/D, the Tchebycheff approach is utilized to convert the multi-objective optimization problem into a number of scalar optimization problems and the number of neighbors is set as 10 as suggested in [46].

6.2. Search performance metrics

Generally, the search performance of a multi-objective optimization algorithm can be evaluated in terms of three criteria (i.e., convergence, uniformity, and spread) [33]. Convergence measures how close the found non-dominated solutions are to the Pareto front, uniformity measures how uniformly the non-dominated solutions are distributed, and spread measures how well the non-dominated solutions cover the Pareto front. There are many search performance metrics proposed in terms of at least one of the three criteria. However, using only one measure may provide misleading information about the quality of the found solutions in some cases [33]. Therefore, to comprehensively compare the search performance between the proposed MPBPSO algorithm and the benchmark algorithms, we adopt four performance metrics, i.e., hypervolume (HV) [45], inverted generational distance (IGD) [4], set coverage (SC) [46], and the convergence distance (CD) [20]. The IGD and CD metrics are the smaller the better, and the HV and SC metrics are the larger the better.

HV, IGD, and SC are used for comparing the final search results of these algorithms. HV is defined as the hypervolume dominated by the non-dominated solutions found by an optimization algorithm in the objective space. It measures the convergence, uniformity, and spread properties of the found non-dominated solutions simultaneously. IGD evaluates the overall distance between non-dominated solutions obtained by an optimization algorithm and the Pareto-optimal solutions in the objective space. It considers the convergence and uniformity of the found non-dominated solutions. SC can be used to compare the goodness between two non-dominated sets. Given two non-dominated sets Ω and Φ. $SC(\Omega, \Phi)$ is defined as the percentage of solutions in Φ that are dominated by or equal to (i.e., covered by) a solution in Ω . In this paper, we let Φ be the Pareto-optimal solutions and use SC to evaluate the performance of the non-

dominated set Ω found by an optimization algorithm. Thus, SC can measure the convergence property of a multi-objective optimization algorithm. CD is used to draw the convergence curves of the optimization algorithms as that used in [20]. It is defined as the average of IGD and GD.

As suggested in [21], we first normalize the objective function values $f_i(X)$ $(i = 1, 2)$ for each obtained solution and then calculate HV, IGD, SC, and CD values based on the normalized objective function values. The min-max normalization method is used where the maximum and minimum values for each objective function are obtained based on the solutions obtained by all the compared optimization algorithms. The calculation of IGD, SC, and CD requires a given Pareto-optimal set. The non-dominated solutions based on the solutions output by all the compared optimization algorithms are used as the approximating Pareto-optimal set for the calculation as suggested in [21]. Moreover, we set the reference point as $\mathbf{r} = (1.1, 1.1)$ to calculate $HV(\Omega, \mathbf{r})$ as suggested in [45].

6.3. Comparison between MPBPSO and benchmark algorithms

Table 4 shows the results of HV, IGD, and SC metrics of MPBPSO and the benchmark multi-objective optimization algorithms. In the table, the mean and standard deviation of each performance metric over the 30 repetitions as well as the statistical test results using the Wilcoxon signed-rank test are shown. The "$\uparrow\uparrow$" ("$\uparrow$") or "$\downarrow\downarrow$" ("$\downarrow$") denotes MPBPSO outperforms or loses to the benchmark algorithm at a significance level of $\alpha = 0.05$ ($\alpha = 0.1$). In terms of HV, MPBPSO obtains significantly higher HV values than the benchmark algorithms except for MOBDE and GADMS on all the four datasets. Compared with MOBDE and GADMS, MPBPSO obtains significantly higher HV values on ADPN and PAPER. In terms of IGD, MPBPSO obtains significantly lower IGD values than CMDPSO, NSPSO, IDMS, MOEA/D, and SPEA2 on all the four datasets, obtains significantly lower IGD values than MNSGA-II on three datasets ADPN, LATEX, and SPIRA, and obtains significantly lower IGD values than GADMS on two datasets ADPN and PAPER. Compared with MOBDE, MPBPSO obtains a significantly lower IGD value on ADPN, but obtains a significantly higher IGD value on SPIRA. In terms of SC, MPBPSO obtains significantly higher SC values than the benchmark algorithms except for MOBDE and GADMS on all the four datasets. MPBPSO obtains significantly higher SC values than MOBDE and GADMS on the two datasets ADPN and PAPER. According to the above results, MPBPSO obtains significantly better HV, IGD, and SC results than the benchmark algorithms in most cases. Except that MPBPSO obtains a significantly worse IGD value than MOBDE on SPIRA, no results show that MPBPSO performs significantly worse than the benchmark algorithms on the three search performance metrics. Overall, the results in Table 4 indicate that MPBPSO has competitive search performance.

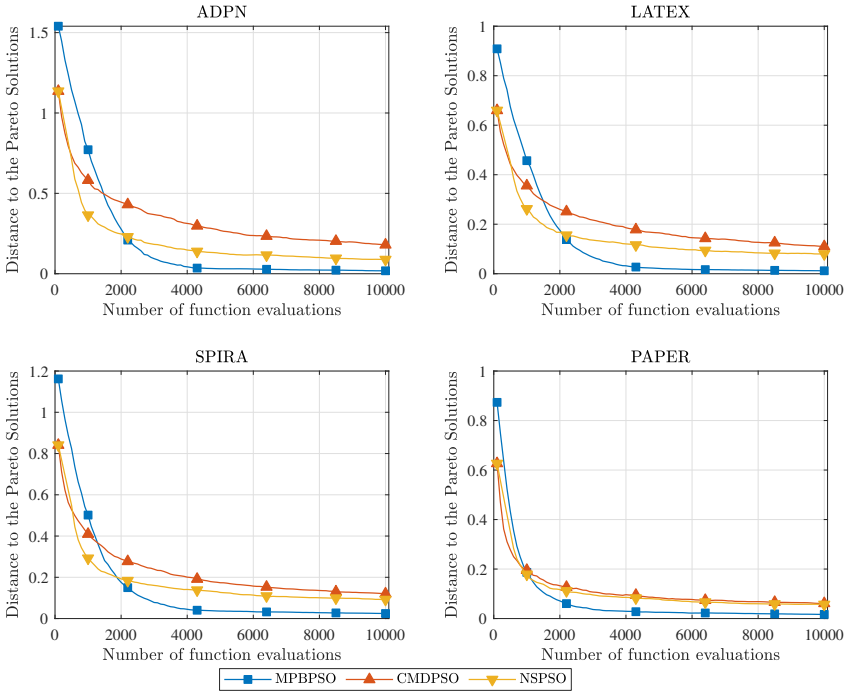
Fig. 5 shows the convergence curves of MPBPSO and the benchmark algorithms drawn with the CD metric. Fig. 5a shows the convergence curves of MPBPSO and the two PSO algorithms, i.e., CMDPSO and NSPSO. According to the figure, the convergence curves of MPBPSO decrease quickly and become

Table 4: Comparison of the search performance between MPBPSO and benchmark optimization algorithms.

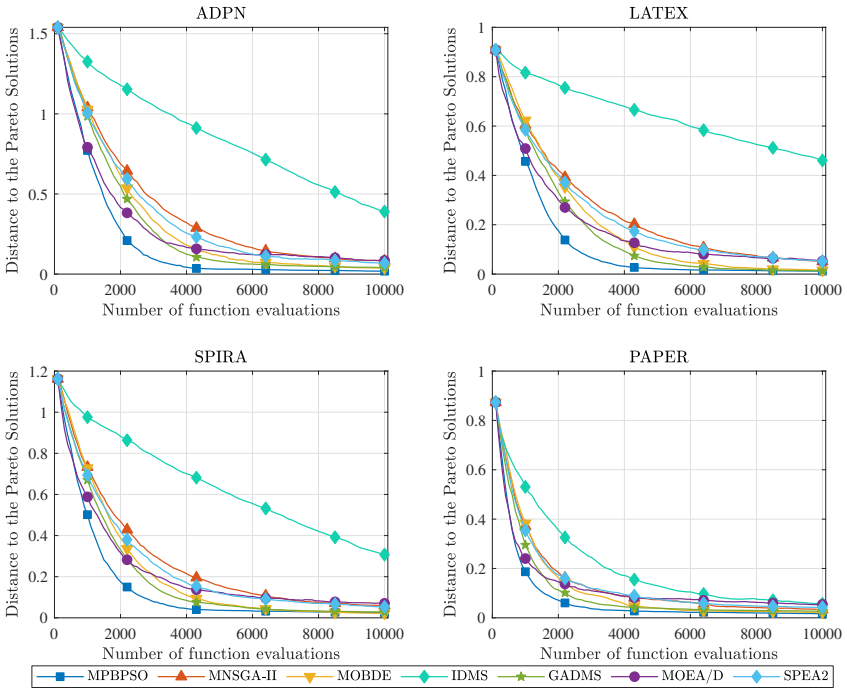
Metric	Dataset	MPBPSO	CMDPSO	NSPSO	MNSGA-II	MOBDE	IDMS	GADMS	MOEA/D	SPEA2
HV	ADPN	$\mathbf{1.160 \pm 0.023}$	0.953 ± 0.093 ⇈	1.091 ± 0.061 ⇈	1.083 ± 0.058 ⇈	1.151 ± 0.030 ⇈	0.645 ± 0.202 ⇈	1.139 ± 0.029 ⇈	1.117 ± 0.031 ⇈	1.121 ± 0.036 ⇈
	LATEX	1.169 ± 0.024	0.971 ± 0.053 ⇈	1.086 ± 0.059 ⇈	1.099 ± 0.041 ⇈	$\mathbf{1.172 \pm 0.021}$	0.506 ± 0.128 ⇈	1.165 ± 0.018	1.132 ± 0.034 ⇈	1.100 ± 0.049 ⇈
	SPIRA	1.120 ± 0.035	0.945 ± 0.083 ⇈	1.021 ± 0.074 ⇈	1.075 ± 0.043 ⇈	$\mathbf{1.128 \pm 0.032}$	0.614 ± 0.162 ⇈	1.114 ± 0.036	1.076 ± 0.044 ⇈	1.087 ± 0.039 ⇈
	PAPER	$\mathbf{1.128 \pm 0.025}$	1.081 ± 0.034 ⇈	1.085 ± 0.051 ⇈	1.114 ± 0.030 ⇈	1.125 ± 0.026 ↑	1.089 ± 0.052 ⇈	1.122 ± 0.028 ⇈	1.098 ± 0.043 ⇈	1.111 ± 0.031 ⇈
IGD	ADPN	$\mathbf{0.022 \pm 0.035}$	0.168 ± 0.053 ⇈	0.076 ± 0.042 ⇈	0.091 ± 0.034 ⇈	0.041 ± 0.038 ⇈	0.393 ± 0.166 ⇈	0.045 ± 0.037 ⇈	0.075 ± 0.026 ⇈	0.064 ± 0.033 ⇈
	LATEX	0.016 ± 0.009	0.112 ± 0.035 ⇈	0.061 ± 0.028 ⇈	0.050 ± 0.021 ⇈	$\mathbf{0.015 \pm 0.007}$	0.481 ± 0.112 ⇈	0.016 ± 0.009	0.041 ± 0.014 ⇈	0.052 ± 0.024 ⇈
	SPIRA	0.033 ± 0.018	0.123 ± 0.043 ⇈	0.091 ± 0.040 ⇈	0.060 ± 0.022 ⇈	$\mathbf{0.025 \pm 0.015}$ ⇊	0.334 ± 0.120 ⇈	0.036 ± 0.018	0.068 ± 0.024 ⇈	0.058 ± 0.020 ⇈
	PAPER	0.026 ± 0.018	0.062 ± 0.017 ⇈	0.062 ± 0.029 ⇈	0.033 ± 0.019	$\mathbf{0.024 \pm 0.015}$	0.062 ± 0.029 ⇈	0.037 ± 0.021 ⇈	0.051 ± 0.025 ⇈	0.049 ± 0.026 ⇈
SC	ADPN	$\mathbf{0.622 \pm 0.240}$	0.000 ± 0.000 ⇈	0.115 ± 0.186 ⇈	0.057 ± 0.129 ⇈	0.437 ± 0.219 ⇈	0.005 ± 0.026 ⇈	0.409 ± 0.242 ⇈	0.121 ± 0.116 ⇈	0.163 ± 0.180 ⇈
	LATEX	0.377 ± 0.158	0.000 ± 0.000 ⇈	0.015 ± 0.050 ⇈	0.026 ± 0.066 ⇈	0.318 ± 0.182	0.000 ± 0.000 ⇈	$\mathbf{0.411 \pm 0.163}$	0.088 ± 0.102 ⇈	0.023 ± 0.066 ⇈
	SPIRA	$\mathbf{0.369 \pm 0.143}$	0.000 ± 0.000 ⇈	0.042 ± 0.078 ⇈	0.056 ± 0.078 ⇈	0.358 ± 0.163	0.000 ± 0.000 ⇈	0.334 ± 0.148	0.079 ± 0.100 ⇈	0.119 ± 0.125 ⇈
	PAPER	$\mathbf{0.505 \pm 0.124}$	0.074 ± 0.109 ⇈	0.134 ± 0.076 ⇈	0.258 ± 0.151 ⇈	0.386 ± 0.139 ⇈	0.142 ± 0.101 ⇈	0.379 ± 0.139 ⇈	0.180 ± 0.106 ⇈	0.280 ± 0.125 ⇈

lower than that of CMDPSO and NSPSO when the number of function evaluations is larger than around 2,000. CMPSO and NSPSO suffer from the premature convergence problem as their convergence curves decrease much slower than MPBPSO in the late phase of the evolutionary process. Moreover, we can find that the convergence curves of CMDPSO and NSPSO start with a lower point than that of MPBPSO on each dataset. This is because CMDPSO and NSPSO use a threshold parameter $\theta = 0.6$, which is larger than 0.5, to determine the selection of features of a particle. The expected number of selected features in the initial swarm of CMDPSO and NSPSO is around $40\%N$ (N is the number of original features), which is smaller than that of MPBPSO (around $50\%N$). This leads to relatively good initial solutions in CMDPSO and NSPSO than MPBPSO. Overall, the results in Fig. 5a show that MPBPSO has a better convergence property than CMDPSO and NSPSO.

Fig. 5b shows the convergence curves of MPBPSO and other non-PSO algorithms, i.e., MNSGA-II, MOBDE, IDMS, GADMS, MOEA/D, and SPEA2. Compared with MOEA/D, MPBPSO can obtain lower convergence curves when the number of function evaluations becomes larger than around 1,000. The convergence curves of MPBPSO and MOEA/D in the early phase of the evolutionary process (before 1,000 function evaluations) are similar. However, the convergence curves of MOEA/D decrease slower than MPBPSO in the late phase of the evolutionary process. This denotes that MOEA/D suffers from the premature convergence problem. MNSGA-II and SPEA2 have close convergence speeds and their convergence curves are substantially higher than that of MPBPSO during the whole evolutionary process on the four datasets. This denotes that MPBPSO has substantially better convergence performance than MNSGA-II and SPEA2. For IDMS, the obtained convergence curves are much higher than other algorithms, especially on the ADPN, LATEX, and SPIRA datasets, which have more features than PAPER. This shows that IDMS has the lowest convergence speed among the optimization algorithms. Finally, it is worth noting that MOBDE and GADMS obtain close convergence curves to MPBPSO in the late phase of the evolutionary process. However, the convergence curves of these two algorithms are substantially higher than MPBPSO in the early phase of the evolutionary process. This means that given a smaller number of allowed function evaluations, MOBDE and GADMS will obtain substantially worse optimization results than MPBPSO. In other words, for the FS problems on high dimensional data, MOBDE and GADMS could show substantially worse search results



(a) Comparison with PSO algorithms



(b) Comparison with non-PSO algorithms

Fig. 5. Comparison of convergence performance between MPBPSO and benchmark multi-objective optimization algorithms.

than MPBPSO if limited computational resources are given because the two algorithms have a significantly lower convergence speed than MPBPSO. Overall, the results in Fig. 5b indicate that MPBPSO shows the best convergence performance among the non-PSO algorithms.

6.4. Performance evaluation of new components in MPBPSO

The results in Section 6.3 have shown that MPBPSO obtains better search performance than the benchmark algorithms. In MPBPSO, we have adopted a new solution update mechanism that combines the modified PSU mechanism with the proposed mutation operator. Moreover, the DPU strategy is used to update *pbest*, and the RWGS strategy is adopted to determine the *gbest* at each iteration. To verify the effectiveness of the new components proposed for MPBPSO, we propose four MPBPSO variants to compare with MPBPSO. The variants include:

- MPBPSO-U, which adopts the standard PSU mechanism of PBPSO [40] without using the mutation operator;
- MPBPSO-M, which uses the standard bitwise mutation operator with a mutation rate of 1/N instead of the proposed mutation operator used in MPBPSO;
- MPBPSO-P, which updates *pbest* based on the Pareto dominance concept; and
- MPBPSO-G, which selects the *gbest* randomly at each iteration instead of using the RWGS strategy.

Similar to MPBPSO, the search results of the variants are collected from 3 repetitions of the 10-fold CV. The performance metrics to compare between MPBPSO and each variant are the HV, IGD, and SC metrics described in Section 6.2.

Table 5 shows the results of the HV, IGD, and SC metrics comparing MPBPSO with each variant. The mean and standard deviation of each performance metric over the 30 repetitions and the statistical significance test results with the Wilcoxon signed-rank test are shown in the table. Similar to the notations in Table 4, the "$\uparrow\uparrow$" ("$\uparrow$") or "$\downarrow\downarrow$" ("$\downarrow$") denotes MPBPSO-IPM outperforms or lose to the variant at a significance level of $\alpha = 0.05$ ($\alpha = 0.1$). Several findings can be obtained from Table 5:

- First, compared with MPBPSO-U, MPBPSO obtains significantly better HV, IGD, and SC results on all the four datasets at the significance level of $\alpha = 0.05$. This denotes that the newly proposed solution update mechanism combining the modified PSU mechanism with the mutation operator is effective in improving the search performance of MPBPSO. The effectiveness of this mechanism can be explained as follows. The particles can be attracted to the area of the current *pbest* and *gbest* with the modified PSU mechanism, which maintains the fast convergence property of MPBPSO. Furthermore, the particles can avoid being easily trapped in the local optima with the mutation operator, which guarantees the exploration performance of particles.

Table 5: Comparison of the search performance between MPBPSO and its variants.

Metric	Dataset	MPBPSO	MPBPSO-U	MPBPSO-M	MPBPSO-P	MPBPSO-G
HV	ADPN	$\mathbf{1.117 \pm 0.026}$	0.884 ± 0.125 ↑↑	1.087 ± 0.038 ↑↑	1.102 ± 0.030 ↑↑	1.111 ± 0.025
	LATEX	$\mathbf{1.063 \pm 0.044}$	0.762 ± 0.077 ↑↑	1.041 ± 0.034 ↑↑	1.049 ± 0.048 ↑↑	1.048 ± 0.045 ↑↑
	SPIRA	$\mathbf{1.046 \pm 0.048}$	0.874 ± 0.089 ↑↑	1.042 ± 0.058	1.042 ± 0.050	1.039 ± 0.056
	PAPER	$\mathbf{1.084 \pm 0.045}$	1.055 ± 0.044 ↑↑	1.074 ± 0.043 ↑↑	1.082 ± 0.046	1.080 ± 0.047 ↑
IGD	ADPN	$\mathbf{0.022 \pm 0.027}$	0.174 ± 0.081 ↑↑	0.068 ± 0.034 ↑↑	0.041 ± 0.032 ↑↑	0.028 ± 0.025
	LATEX	$\mathbf{0.041 \pm 0.032}$	0.201 ± 0.044 ↑↑	0.054 ± 0.028 ↑↑	0.048 ± 0.032	0.049 ± 0.028
	SPIRA	$\mathbf{0.037 \pm 0.014}$	0.139 ± 0.049 ↑↑	0.042 ± 0.025	0.041 ± 0.022	0.040 ± 0.021
	PAPER	0.025 ± 0.019	0.050 ± 0.024 ↑↑	0.031 ± 0.022	$\mathbf{0.023 \pm 0.015}$	0.028 ± 0.019
SC	ADPN	$\mathbf{0.624 \pm 0.241}$	0.010 ± 0.039 ↑↑	0.325 ± 0.207 ↑↑	0.428 ± 0.243 ↑↑	0.572 ± 0.209
	LATEX	$\mathbf{0.386 \pm 0.171}$	0.000 ± 0.000 ↑↑	0.233 ± 0.166 ↑↑	0.347 ± 0.168	0.335 ± 0.138
	SPIRA	0.381 ± 0.140	0.000 ± 0.000 ↑↑	0.296 ± 0.183 ↑	0.367 ± 0.105	$\mathbf{0.402 \pm 0.185}$
	PAPER	$\mathbf{0.526 \pm 0.128}$	0.231 ± 0.138 ↑↑	0.451 ± 0.198 ↑	0.473 ± 0.153 ↑	0.474 ± 0.182

- Second, compared with MPBPSO-M, MPBPSO obtains significantly better HV results on ADPN, LATEX, and PAPER, obtains significantly better IGD results on ADPN and LATEX, and obtains significantly better SC results on all the four datasets at the significance level of $\alpha = 0.05$ or 0.1. No results indicate MPBPSO obtains significantly worse search performance than MPBPSO-M. These results denote that the proposed mutation operator in MPBPSO is effective in improving its performance. The reason for the effectiveness of the proposed mutation operator can be explained as follows. One drawback of the standard mutation operator in MPBPSO-M is that more elements in a particle will change from 0 to 1 than that from 1 to 0 when the elements of 0 are more than the elements of 1 in a solution because the same flipping probability is applied to each element in a particle. This means that during the late phase of the evolutionary process, the mutation operator has a negative effect on reducing the number of selected features/QFs. A similar effect is shown in the standard PSU mechanism of PBPSO, as the term p_0 in Eq. (9) contributes to the flipping probability of each element in a particle. In comparison, the proposed mutation operator has the property that maintains the expected number of selected features after the mutation, which is an advantage compared with the standard mutation operator for FS problems.

- Third, compared with MPBPSO-P, MPBPSO obtains significantly better HV results on ADPN and LATEX, obtains a significantly better IGD value on ADPN, and obtains significantly better SC results on ADPN and PAPER. Considering the mean values of HV, IGD, and SC, MPBPSO obtains better results than MPBPSO-P in 11 of the 12 cases. These results show that MPBPSO has better search performance than MPBPSO-P, which indicates that the proposed DPU strategy is effective in improving the search performance. Compared with the Pareto dominance concept, the proposed DPU strategy can compare the current position and the *pbest* of a particle even if they do not dominate each other. This makes the update of *pbest* more reasonable with the DPU strategy in the multi-objective scenario, which improves the search performance.

- Finally, compared with MPBPSO-G, MPBPSO obtains significantly better HV results on LATEX and PAPER. Moreover, MPBPSO obtains better mean values of HV, IGD, and SC than MPBSO-G in 11 of the 12 cases. This indicates the RWGS strategy can perform more effectively than the random selection strategy because the computational resources can be more uniformly allocated to different regions of the non-dominated front by RWGS. Therefore, it is preferred to use the RWGS strategy in MPBPSO.

6.5. Discussion

The experimental results in Section 5 have indicated that the proposed MPBPSO-IPM method is effective for identifying KQFs in complex manufacturing processes. Specifically, the results have shown that the KQFs identified by MPBPSO-IPM obtain good predictive performance for both premium and regular quality products, which indicates good KQF identification performance is obtained by MPBPSO-IPM. Further analysis in this section has shown that the proposed multi-objective optimization algorithm MPBPSO has better search performance than benchmark optimization algorithms. These results have shown that both the optimization algorithm and the KQF identification model applied in MPBPSO-IPM are effective.

The proposed KQF identification method, MPBPSO-IPM, can be very beneficial for effective quality control and improvement in real manufacturing processes. On the one hand, KQF identification determines the key factors required to be strictly monitored and controlled by quality control tools (such as statistical quality control (SPC)) in the manufacturing processes. Therefore, KQF identification is a pre-processing step for effective quality control in complex manufacturing processes. On the other hand, KQF identification is beneficial for continuous product quality improvement. Quality improvement tools such as response surface methodology (RSM) can be applied to optimize the identified KQFs, which can improve the quality of products.

The MPBPSO-IPM method is also beneficial for building an effective and efficient quality prediction model. Quality prediction models are widely required in modern industries. Quality prediction models built based on the manufacturing process data can timely predict the quality of products so that quality engineers can take accurate and timely measures to control the quality of final products [29]. However, the massive industrial data collected from the manufacturing processes may contain noisy and redundant QFs for predicting product quality. The application of a KQF identification method (such as the proposed MPBPSO-IPM) becomes an indispensable step for building a concise and effective learning model for quality prediction because it can select a small number of informative KQFs with good discriminative power for product quality.

7. Conclusions

In this paper, we have proposed an FS method called MPBPSO-IPM for identifying KQFs in complex manufacturing processes that significantly affect the final quality of products. In MPBPSO-IPM, MPBPSO is used to find a set of non-dominated solutions (candidate KQF sets), and then, the IPM is used to select the final KQF set. Specifically, MPBPSO is adapted from a probability-based BPSO algorithm to the multi-objective KQF identification model with several new components, including the solution update mechanism combining the modified PSU mechanism with the mutation operator, the DPU strategy, and the RWGS strategy. The experimental results on four unbalanced datasets show that MPBPSO-IPM is effective for identifying KQFs in the complex manufacturing processes. Specifically, MPBPSO-IPM identifies a small number of KQFs that can well predict the quality of both premium and regular quality products, which means the irrelevant and redundant QFs are substantially eliminated. We have also conducted experiments to verify the search performance of MPBPSO. The experimental results indicate that MPBPSO has better search performance than several typical multi-objective optimization algorithms including CMDPSO, NSPSO, MNSGA-II, MOBDE, IDMS, GADMS, MOEA/D, and SPEA2. Further analysis reveals that the proposed solution update mechanism for MPBPSO is more effective than the standard PSU mechanism in optimizing multi-objective FS problems. Moreover, the proposed DPU and RWGS strategies are also effective in improving the search performance, as they can properly update *pbest* and select *gbest* for MPBPSO in the multi-objective scenario.

The proposed MPBPSO-IPM is an FS method based on the wrapper framework. It is worth extending the application of MPBPSO-IPM based on the filter framework of FS, as a filter method is more time efficient than a wrapper method. This extension would be beneficial for large scale datasets with even more features and instances than that used in the experiments of this study. Moreover, building a regression model based FS method for KQFs identification for the data with a continuous response variable (i.e., quality variable) is one of our future research interests.

Appendix A. Tuning parameters in MPBPSO

In MPBPSO, two parameters, learning rate p_l and mutation rate p_m, are potential key factors that affect its search performance, which further affects the final KQF identification performance of MPBPSO-IPM. In this section, we design a set of experiments to tune p_l and p_m for MPBPSO. We will examine a set $\Lambda_l = \{0.5, 0.6, ..., 1.0\}$ of 6 parameter settings for p_l and a set $\Lambda_m = \{0.80, 0.05, ..., 1.00\}$ of 5 parameter settings for p_m, which yields 30 combinations of p_l and p_m (i.e., $\{(p_l, p_m)|\forall p_l \in \Lambda_l, \forall p_m \in \Lambda_m\}$). The ADPN dataset is used in the tuning experiments because it requires less computational time than other datasets and also has relatively high data dimensionality. With each parameter setting, MPBPSO is applied to the KQF identification model defined in Eq. (1) to obtain a set of non-dominated solutions. The non-dominated

solutions are used to evaluate the performance of each parameter setting. The HV metric described in Section 6.2 is adopted as the measure to decide a desirable parameter setting because it is a Pareto-compliant metric and takes into account the convergence, diversity, and spread properties of the obtained non-dominated solutions simultaneously [33]. Note that, similar to the experimental setting described in Section 4.4, we repeat the 10-fold CV process 3 times with each parameter setting as MPBPSO is a stochastic search method, which yields 30 repetitions of the experiments. Thus, 30 HV values can be obtained with each parameter setting, which comprehensively evaluates the performance and stability of the algorithm. Furthermore, considering both the mean and variance of the obtained 30 HV values for each parameter setting, we convert the 30 HV values into the Taguchi's signal-to-noise (S/N) ratio [31], which is defined as

$$S/N = -10 \log \left(\frac{1}{N_r} \sum_{r=1}^{N_r} \left(\frac{1}{HV_r} \right)^2 \right) \tag{A.1}$$

where $N_r = 30$ is the number of obtained HV values, HV_r is the HV value of the rth repetition. A larger S/N ratio denotes a better overall quality of the obtained HV values for a parameter setting.

The S/N ratio with each parameter setting of p_l and p_m is shown in Table A.1. We can find that the learning rate p_l should not be set as a too large value as the parameter settings with $p_l > 0.7$ generally yield worse results than that with $p_l \leq 0.7$. Moreover, we can find that setting p_m as either 0.85, 0.90, 0.95, or 1.00 given $p_l = 0.60$ will give relatively good S/N results. So, $p_l = 0.6$ is recommended for MPBPSO to yield stable and good search results. The best S/N ratio is 0.5059, which is from the parameter setting $p_l = 0.60$ and $p_m = 0.85$. Therefore, we use $p_l = 0.60$ and $p_m = 0.85$ for MPBPSO in the experiments in this paper.

Table A.1: The S/N ratios obtained by different parameter settings of p_l and p_m.

p_l \ p_m	0.80	0.85	0.90	0.95	1.00
0.50	0.4634	0.4483	0.4555	0.4679	0.4767
0.60	0.4348	**0.5059**	0.4812	0.4525	0.4856
0.70	0.4512	0.4583	0.4567	0.4371	0.4042
0.80	0.3542	0.4461	0.4260	0.4564	0.4445
0.90	0.4287	0.3481	0.4302	0.4344	0.4075
1.00	0.3938	0.4155	0.4391	0.4083	0.4428

Acknowledgments

The authors would like to thank the editor and anonymous referees for the constructive comments and suggestions. This work was supported in part by the National Natural Science Foundation of China (NSFC) [grant numbers 72101182 and 72231005]; State Administration of Science, Technology and Industry for National Defense of China [grant number JSZL2021204B001]; the Humanities and Social Sciences Youth

Fund of Ministry of Education of China [grant numbers 19YJC630071 and 19YJC630221]; the Marsden Fund of New Zealand Government [contract numbers VUW1913, VUW1914, VUW2115]; MBIE Data Science SSIF Fund [contract number RTVU1914]; Huayin Medical [grant number E3791/4165]; and MBIE Endeavor Research Programme [contract numbers C11X2001 and UOCX2104].

References

[1] E. Amaldi, V. Kann, On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems, Theoretical Computer Science 209 (1) (1998) 237 – 260.

[2] M. J. Anzanello, S. L. Albin, W. A. Chaovalitwongse, Multicriteria variable selection for classification of production batches, European Journal of Operational Research 218 (1) (2012) 97 – 105.

[3] U. Bhowan, M. Johnston, M. Zhang, Developing new fitness functions in genetic programming for classification with unbalanced data, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 42 (2) (2012) 406–421.

[4] K. Deb, H. Jain, An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints, IEEE Transactions on Evolutionary Computation 18 (4) (2014) 577–601.

[5] K. Deb, S. Agrawal, A. Pratap, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6 (2) (2002) 182–197.

[6] A. Ekbal, S. Saha, Multiobjective optimization for classifier ensemble and feature selection: an application to named entity recognition, International Journal on Document Analysis and Recognition (IJDAR) 15 (2) (2012) 143–166.

[7] J.-P. Gauchi, P. Chagnon, Comparison of selection methods of explanatory variables in PLS regression with application to manufacturing process data, Chemometrics and Intelligent Laboratory Systems 58 (2) (2001) 171 – 193.

[8] B. Guan, Y. Zhao, Y. Yin, Y. Li, A differential evolution based feature combination selection algorithm for high-dimensional data, Information Sciences 547 (2021) 870–886.

[9] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, Journal of Machine Learning Research 3 (2003) 1157–1182.

[10] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, The weka data mining software: an update, ACM SIGKDD explorations newsletter 11 (1) (2009) 10–18.

[11] F. Han, W.-T. Chen, Q.-H. Ling, H. Han, Multi-objective particle swarm optimization with adaptive strategies for feature selection, Swarm and Evolutionary Computation 62 (2021) 100847.

[12] H. Ishibuchi, N. Tsukamoto, Y. Nojima, Evolutionary many-objective optimization: A short review, in: Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2008, June 1-6, 2008, Hong Kong, China, 2008, pp. 2419–2426.

[13] Y. Ji, S. Liu, M. Zhou, Z. Zhao, X. Guo, L. Qi, A machine learning and genetic algorithm-based method for predicting width deviation of hot-rolled strip in steel production systems, Information Sciences 589 (2022) 360–375.

[14] G. H. John, P. Langley, Estimating continuous distributions in bayesian classifiers, in: Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence, UAI'95, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1995, pp. 338–345.

[15] J. Kennedy, R. C. Eberhart, A discrete binary version of the particle swarm algorithm, in: 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation, volume 5, 1997, pp. 4104–4108.

[16] M. A. Khanesar, M. Teshnehlab, M. A. Shoorehdeli, A novel binary particle swarm optimization, in: 2007 Mediterranean Conference on Control Automation, 2007, pp. 1–6.

[17] R. Kohavi, G. H. John, Wrappers for feature subset selection, Artificial Intelligence 97 (1) (1997) 273 – 324.

[18] N. Kozodoi, S. Lessmann, K. Papakonstantinou, Y. Gatsoulis, B. Baesens, A multi-objective approach for profit-driven feature selection in credit scoring, Decision Support Systems 120 (2019) 106 – 117.

[19] A.-D. Li, Z. He, Y. Zhang, Bi-objective variable selection for key quality characteristics selection based on a modified NSGA-II and the ideal point method, Computers in Industry 82 (2016) 95 – 103.

[20] A.-D. Li, Z. He, Q. Wang, Y. Zhang, Key quality characteristics selection for imbalanced production data using a two-phase bi-objective feature selection method, European Journal of Operational Research 274 (3) (2019) 978 – 989.

[21] A.-D. Li, B. Xue, M. Zhang, Multi-objective feature selection using hybridization of a genetic algorithm and direct multisearch for key quality characteristic selection, Information Sciences 523 (2020) 245 – 265.

[22] A.-D. Li, B. Xue, M. Zhang, Improved binary particle swarm optimization for feature selection with new initialization and search space reduction strategies, Applied Soft Computing 106 (2021) 107302.

[23] S. Liu, H. Wang, W. Peng, W. Yao, A surrogate-assisted evolutionary feature selection algorithm with parallel random grouping for high-dimensional classification, IEEE Transactions on Evolutionary Computation 26 (5) (2022) 1087–1101. doi:10.1109/TEVC.2022.3149601.

[24] K. Mistry, L. Zhang, S. C. Neoh, C. P. Lim, B. Fielding, A micro-GA embedded PSO feature selection approach to intelligent facial emotion recognition, IEEE Trans. Cybernetics 47 (6) (2017) 1496–1509.

[25] H. B. Nguyen, B. Xue, I. Liu, P. Andreae, M. Zhang, New mechanism for archive maintenance in PSO-based multi-objective feature selection, Soft Computing 20 (10) (2016) 3927–3946.

[26] B. H. Nguyen, B. Xue, P. Andreae, M. Zhang, A new binary particle swarm optimization approach: Momentum and dynamic balance between exploration and exploitation, IEEE Transactions on Cybernetics 51 (2) (2021) 589–603.

[27] J. A. Pacheco, S. Casado, F. Ángel-Bello, A. M. Alvarez, Bi-objective feature selection for discriminant analysis in two-class classification, Knowledge-Based Systems 44 (2013) 57 – 64.

[28] G. Patterson, M. Zhang, Fitness functions in genetic programming for classification with unbalanced data, in: M. A. Orgun, J. Thornton (Eds.), AI 2007: Advances in Artificial Intelligence, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 769–775.

[29] L. Ren, Z. Meng, X. Wang, L. Zhang, L. T. Yang, A data-driven approach of product quality prediction for complex production systems, IEEE Transactions on Industrial Informatics 17 (9) (2021) 6457–6465.

[30] J. D. Rodríguez, A. P. Martínez, J. A. Lozano, Sensitivity analysis of k-fold cross validation in prediction error estimation, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (3) (2010) 569–575.

[31] A. Sadeghi, A. Alem-Tabriz, M. Zandieh, Product portfolio planning: a metaheuristic-based simulated annealing algorithm, International Journal of Production Research 49 (8) (2011) 2327–2350.

[32] X.-F. Song, Y. Zhang, D.-W. Gong, X.-Z. Gao, A fast hybrid feature selection based on correlation-guided clustering and particle swarm optimization for high-dimensional data, IEEE Transactions on Cybernetics 52 (9) (2022) 9573–9586. doi:10.1109/TCYB.2021.3061152.

[33] R. Tanabe, H. Ishibuchi, An analysis of quality indicators using approximated optimal distributions in a 3-D objective space, IEEE Transactions on Evolutionary Computation 24 (5) (2020) 853–867.

[34] D. A. Van Veldhuizen, Multiobjective evolutionary algorithms: Classifications, analyzes, and new innovations, Technical Report, Air Force Institute of Technology, 1999.

[35] X. Wang, T. Hu, L. Tang, A multiobjective evolutionary nonlinear ensemble learning with evolutionary feature selection for silicon prediction in blast furnace, IEEE Transactions on Neural Networks and Learning Systems 33 (5) (2022) 2080–2093.

[36] P. Wang, B. Xue, J. Liang, M. Zhang, Differential evolution-based feature selection: A niching-based multiobjective approach, IEEE Transactions on Evolutionary Computation 27 (2) (2023) 296–310. doi:10.1109/TEVC.2022.3168052.

[37] F. Wilcoxon, Individual comparisons by ranking methods, Biometrics Bulletin 1 (6) (1945) 80 – 83.

[38] S. Wold, M. Sjöström, L. Eriksson, PLS-regression: a basic tool of chemometrics, Chemometrics and Intelligent Laboratory

Systems 58 (2) (2001) 109 – 130.

[39] B. Xue, M. Zhang, W. N. Browne, Particle swarm optimization for feature selection in classification: A multi-objective approach, IEEE Transactions on Cybernetics 43 (6) (2013) 1656–1671.

[40] B. Xue, S. Nguyen, M. Zhang, A new binary particle swarm optimisation algorithm for feature selection, in: A. I. Esparcia-Alcázar, A. M. Mora (Eds.), Applications of Evolutionary Computation, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014, pp. 501–513.

[41] B. Xue, M. Zhang, W. N. Browne, Particle swarm optimisation for feature selection in classification: Novel initialisation and updating mechanisms, Applied Soft Computing 18 (2014) 261 – 276.

[42] B. Xue, M. Zhang, W. N. Browne, X. Yao, A survey on evolutionary computation approaches to feature selection, IEEE Transactions on Evolutionary Computation 20 (4) (2016) 606–626.

[43] H. Yang, S. Kumara, S. T. Bukkapatnam, F. Tsung, The internet of things for smart manufacturing: A review, IISE Transactions 51 (11) (2019) 1190–1216.

[44] L. Yu, H. Liu, Efficient feature selection via analysis of relevance and redundancy, Journal of Machine Learning Research 5 (Oct.) (2004) 1205–1224.

[45] Y. Yuan, H. Xu, B. Wang, X. Yao, A new dominance relation-based evolutionary algorithm for many-objective optimization, IEEE Transactions on Evolutionary Computation 20 (1) (2016) 16–37.

[46] Q. Zhang, H. Li, MOEA/D: A multiobjective evolutionary algorithm based on decomposition, IEEE Transactions on Evolutionary Computation 11 (6) (2007) 712–731.

[47] Y. Zhang, D. Gong, X. Gao, T. Tian, X. Sun, Binary differential evolution with self-learning for multi-objective feature selection, Information Sciences 507 (2020) 67–85.

[48] Y. Zhang, Y.-H. Wang, D.-W. Gong, X.-Y. Sun, Clustering-guided particle swarm feature selection algorithm for high-dimensional imbalanced data with missing values, IEEE Transactions on Evolutionary Computation 26 (4) (2022) 616–630. doi:10.1109/TEVC.2021.3106975.

[49] Y. Zhu, J. Liang, J. Chen, Z. Ming, An improved NSGA-III algorithm for feature selection used in intrusion detection, Knowledge-Based Systems 116 (2017) 74 – 85.

[50] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: Improving the strength pareto evolutionary algorithm, in: Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems. Proceedings of the EUROGEN'2001. Athens. Greece, September 19-21, 2001, pp. 95–100.