Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Transfer Learning for Gaussian Process Assisted Evolutionary Bi-objective Optimization for Objectives with Different Evaluation Times

Xilu Wang, Sebastian Schmitt, Markus Olhofer, Yaochu Jin

2020

Preprint:

Copyright ACM 2020. This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record was published in GECCO 2020, https://doi.org/10.1145/3377930.3390147.

Transfer Learning for Gaussian Process Assisted Evolutionary Bi-objective Optimization for Objectives with Different Evaluation Times*

Xilu Wang
Department of Computer Science of University of Surrey
Surrey, United Kingdom
xw00616@surrey.ac.uk

Yaochu Jin
Department of Computer Science of University of Surrey
Surrey, United Kingdom
yaochu.jin@surrey.ac.uk

Sebastian Schmitt
Honda Research Institute Europe, GmbH
Offenbach/Main, Germany
Sebastian.Schmitt@honda-ri.de

Markus Olhofer
Honda Research Institute Europe, GmbH
Offenbach/Main, Germany
markus.olhofer@honda-ri.de

ABSTRACT

Despite the success of evolutionary algorithms (EAs) for solving multi-objective problems, most of them are based on the assumption that all objectives can be evaluated within the same period of time. However, in many real-world applications, such an assumption is unrealistic since different objectives must be evaluated using different computer simulations or physical experiments with various time complexities. To address this issue, a surrogate assisted evolutionary algorithm along with a parameter-based transfer learning (T-SAEA) is proposed in this work. While the surrogate for the cheap objective can be updated on sufficient training data, the surrogate for the expensive one is updated by either the training data set or a transfer learning approach. To find out the transferable knowledge, a filter-based feature selection algorithm is used to capture the pivotal features of each objective, and then use the common important features as a carrier for knowledge transfer between the cheap and expensive objectives. Then, the corresponding parameters in the surrogate models are adaptively shared to enhance the quality of the surrogate models. The experimental results demonstrate that the proposed algorithm outperforms the compared algorithms on the bi-objective optimization problems whose objectives have a large difference in computational complexities.

CCS CONCEPTS

• **Theory of computation** → **Evolutionary algorithms**;
• **Computing methodologies** → *Genetic algorithms*;

KEYWORDS

Transfer learning, delayed objective functions, Gaussian processes, surrogate models, evolutionary algorithms, Bayesian optimization

ACM Reference Format:
Xilu Wang, Yaochu Jin, Sebastian Schmitt, and Markus Olhofer. 2020. Transfer Learning for Gaussian Process Assisted Evolutionary Bi-objective Optimization for Objectives with Different Evaluation Times. In *Proceedings of the Genetic and Evolutionary Computation Conference 2020 (GECCO '20).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3377930.3390147

1 INTRODUCTION

In many multi-objective optimization problems (MOPs), the evaluation of some objectives involves very costly physical experiments or computationally intensive numerical simulations. This class of problems is common in real-world applications, where, e.g., computationally intensive computational fluid dynamics (CFD) simulations must be conducted to evaluate the performance in structural design optimization [14] and in engineering design optimization [11]. In this context, surrogate assisted evolutionary algorithms (SAEAs) are proposed for reducing the number of objective evaluations [12]. Despite their success, most existing SAEAs share a common assumption that it takes similar computation time to evaluate different objectives so that environmental selection can be carried out once all objectives are evaluated within the same period of time. However, such an assumption does not take into account the common real-life situations where the evaluation time of different objectives may vary a lot. For example, it may take minutes to evaluate the aerodynamics of a vehicle, while it can take dozens of hours to evaluate the vehicle's crash performance. Although such optimization problems can still be handled using a canonical multi-objective evolutionary algorithm (MOEA) by simply waiting until all objectives are evaluated, the efficiency will become extremely low. Therefore, here we consider the multi-objective optimization problems where at least one of the objectives requires a long period of evaluation time, or all

*Produces the permission block, and copyright information

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
GECCO '20, July 8–12, 2020, Cancún, Mexico
© 2020 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-7128-5/20/07…$15.00
https://doi.org/10.1145/3377930.3390147

objectives are expensive but their time complexities are very different. Such MOPs are referred as multi-objective optimization for objectives with non-uniform latencies or evaluation times [1]. For the sake of simplicity, this work focuses on bi-objective optimization problems and in the following, we define the related notations and assumptions.

- The expensive (or delayed/slow) objective is denoted as y^{ex} while the cheap (or non-delayed/fast) one is denoted as y^{ch}. Subsequently, the ratio of the evaluation time of y^{ex} to the evaluation time of y^{ch} is presented as τ.
- Evaluations of the objectives can be parallelized. Firstly, each candidate solution can be evaluated on both the fast and slow objective functions in parallel; secondly, each objective function can be evaluated on a number of candidate solutions at the same time.
- It is assumed that the computational time for building surrogates and for implementing the genetic operators is negligible compared to that for evaluating the expensive objectives. Consequently, the total computational time available for solving the problem is defined as the total budget of objective function evaluations. Specifically, the total budget is defined as the maximum number of objective function evaluations for y^{ex}, denoted as $MaxFE^{ex}$; then correspondingly the limited budget for y^{ch} is $MaxFE^{ex} * \tau$.

To the best of our knowledge, very little work has been done on optimization problems discussed above. Allmendinger and Knowles [2] solved such a problem by assigning pseudovalues instead of the true evaluated values to delayed objectives. Another idea is to fully use the evaluations for cheap objectives instead of making approximation of the expensive objectives [1], which suggests three distinct delayed-handling methods: waiting, fast-first and interleaving strategies. Specifically, the waiting strategy simply waits for all the objectives to be evaluated completely before continuing the optimization process; the fast-first strategy firstly leverages all the budget of evaluations for the fast objective via a single-objective evolutionary algorithm (EA), and then chooses some individuals to be evaluated by the slow one; the interleaving strategies, including brood and speculative interleaving methods, try to exchange some information of the two objectives. More recently, Tinkle *et al.* [8] proposed an algorithm called HK-RVEA to tackle MOPs for objectives with different evaluation times from the perspective of surrogate models. In HK-RVEA, two strategies are introduced for maximizing the utilization of the fast objective evaluation, a single-objective EA to optimize the fast objective in the initial stage and the genetic operators to sample more data points for the fast objective. HK-RVEA shows promising performance when the latencies are not too big. It is worthy noting that although SAEAs have been successfully applied to many optimization problems, tackling MOPs for delayed objectives by SAEA is challenging [2] since many factors must be taken into consideration, such as how to assign the evaluation budget and how to update the surrogates.

The ideas discussed above provide some preliminary ideas for solving MOPs with delayed objectives, but most of them concentrate on taking advantage of the evaluations within the framework of evolutionary algorithms without using any advanced machine learning techniques. Motivated by the fact that only a small amount of data is available for training the surrogate of y^{ex}, while a relatively large amount of data can be collected for training the surrogate of y^{ch}, this work aims to leverage the abundant solutions evaluated by y^{ch} to improve the learning of the surrogate for y^{ex} with the help of a transfer learning method. Specifically, similar to HK-RVEA, the per-objective evaluation budget is fully utilized in our work. When waiting for the expensive objective evaluations on the initial population and the new selected samples, a single-objective EA and the Latin hypercube sampling (LHS) method are adopted to sample more solutions to be evaluated by y^{ch}, respectively. Besides, we make efforts to transfer the knowledge obtained from y^{ch} to improve the learning of the surrogate for y^{ex}. To identify the transferable knowledge, a feature selection algorithm is used to capture the pivot features (a subset of decision variables) of each objective. Subsequently, the corresponding parameters in the surrogate model of y^{ex} will learn from the surrogate model of y^{ch}.

In the following section, a brief introduction to the related techniques is presented, including the Gaussian processes (GPs), acquisition functions (AFs) in Bayesian optimization, feature selection techniques, and transfer learning based on GPs. Section 3 gives the details of the proposed algorithm. The experimental results on two suites of benchmarks and analysis are given in Section 4. Section 5 presents a summary of the paper and discussions of the future work.

2 RELATED TECHNIQUES

Here, a brief review of Gaussian processes is given first, followed by an introduction to acquisition functions, which serve as the model management strategy for sampling new data to update the GP. Then, the feature selection method is reviewed. At last, a briefly discussion about transfer learning methods based on GPs is given.

2.1 Gaussian Processes

Gaussian processes are commonly used as surrogate models in SAEAs due to their ability for providing both predictions and a confidence level of its predictions. Note that the confidence level is essential in model management to balance the exploration and exploitation in SAEAs. The idea behind GPs [21] is that there is a multivariate Gaussian distribution on R^n for any finite set of N n-dimensional inputs $\mathbf{X} = \{\mathbf{x}^1, \mathbf{x}^2, ...\mathbf{x}^N\}^T$. Generally, GPs work on an underlying assumption of smooth domains, i.e., the input $\mathbf{X}$ is continuous so is the output $\mathbf{Y}$ [20]. Given a set of inputs $\mathbf{x}$ and their associated function values $\mathbf{Y} = \{y^1, y^2, ...y^N\}^T$ as the training data, a GP model can be expressed by:

$$y(\mathbf{x}) = \mu + \varepsilon(\mathbf{x}) \tag{1}$$

where μ is the mean of the stochastic process, and $\varepsilon(\mathbf{x})$ is drawn from a stochastic process with a zero mean and a standard deviation σ.

$$\varepsilon(\mathbf{x}) \sim N(0, \sigma^2) \tag{2}$$

The correlation between the error terms of any two data points, $\mathbf{x}^i$ and $\mathbf{x}^j$, is strongly related to their distance. In general, the squared exponential function with additional hyperparameters, rather than an Euclidean distance function, is employed to calculate the correlation:

$$d(\mathbf{x}^i, \mathbf{x}^j) = \sum_{k=1}^{m} \theta_k |x_k^i - x_k^j|^{p_k} \tag{3}$$

$$\mathrm{Corr}(\mathbf{x}^i, \mathbf{x}^j) = \exp[-d(\mathbf{x}^i, \mathbf{x}^j)] \tag{4}$$

where $p_k \in (0, 1)$ controls the smoothness of the function in terms of the k-th dimension, and $\theta_k > 0$ denotes the importance of this dimension. When there are N training data, an $N \times N$ correlation matrix $\mathbf{C}$ will be obtained,

$$\mathbf{C} = \begin{pmatrix} \mathrm{Corr}(\mathbf{x}^1, \mathbf{x}^2) & \cdots & \mathrm{Corr}(\mathbf{x}^1, \mathbf{x}^N) \\ \vdots & \ddots & \vdots \\ \mathrm{Corr}(\mathbf{x}^N, \mathbf{x}^1) & \cdots & \mathrm{Corr}(\mathbf{x}^N, \mathbf{x}^N) \end{pmatrix} \tag{5}$$

As a result, $2N + 2$ hyperparameters will determine a GP model, which can be estimated by maximizing the following likelihood function,

$$\psi(\theta_1, ..., \theta_N, p_1, ..., p_N) = -\frac{1}{2}\left(N \ln \sigma^2 + \ln det(\mathbf{C})\right) \tag{6}$$

Therefore, the estimates $\hat{\mu}$ and $\hat{\sigma}^2$ for the true values μ and σ will be obtained

$$\hat{\mu} = \frac{\mathbf{1}^T \mathbf{C}^{-1} \mathbf{Y}}{\mathbf{1}^T \mathbf{C}^{-1} \mathbf{1}} \tag{7}$$

$$\hat{\sigma}^2 = \frac{(\mathbf{Y} - \mathbf{1}\hat{\mu})^T \mathbf{C}^{-1} (\mathbf{Y} - \mathbf{1}\hat{\mu})}{N} \tag{8}$$

where $\mathbf{1}$ denotes a $N \times 1$ column vector of ones. Based on the given parameters, the GP model can predict the mean value together with a variance for a new data point $\mathbf{x}^{new}$,

$$f(\mathbf{x}^{new}) = \hat{\mu} + \mathbf{r}^T \mathbf{C}^{-1} (\mathbf{Y} - \mathbf{1}\hat{\mu}) \tag{9}$$

$$\hat{\sigma}(\mathbf{x}^{new})^2 = \hat{\sigma}^2 [1 - \mathbf{r}^T \mathbf{C}^{-1} \mathbf{r} + \frac{(1 - \mathbf{r}^T \mathbf{C}^{-1} \mathbf{r})^2}{\mathbf{1}^T \mathbf{C}^{-1} \mathbf{1}}] \tag{10}$$

where $\mathbf{r} = (\mathrm{Corr}(\mathbf{x}^{new}, \mathbf{x}^1), ..., \mathrm{Corr}(\mathbf{x}^{new}, \mathbf{x}^N))^T$ presents a correlation vector between $\mathbf{x}^{new}$ and each element $\mathbf{x}^i$ in $\mathbf{X}$.

2.2 Acquisition Functions

In SAEAs, it is non-trivial to build high-quality surrogate models based on limited training data. Therefore, it is very important to manage the surrogates properly. The concept of model management is introduced to identify a limited number of new samples to be evaluated using the real objective functions for updating the surrogates [13]. In the context of GP-based SAEAs, acquisition functions (AFs) used in Bayesian optimization, such as expected improvement (EI) [15] and lower confidence bound (LCB) [23], have been used for model management. The LCB is designed to balance the exploration and exploitation by combining the uncertainty with the predicted objective values. To achieve better balance between exploration and exploitation, an adaptive acquisition function (ALCB) based on LCB is proposed [24], defined as follows:

$$AF = (1 - \alpha) \cdot (\boldsymbol{\mu}_{\mathbf{x}}./\boldsymbol{\mu}_{max}) + \alpha \cdot (\boldsymbol{\sigma}_{\mathbf{x}}./\boldsymbol{\sigma}_{max}) \tag{11}$$

where

$$\alpha = -0.5 \cdot \cos(\frac{FE^{ex}}{MaxFE^{ex}} \cdot \pi) + 0.5 \tag{12}$$

where './' denotes element-by-element division for two vectors of the same size; $\boldsymbol{\mu}_{\mathbf{x}}$ and $\boldsymbol{\sigma}_{\mathbf{x}}$ denote the mean and variance values of each sample $\mathbf{x}$, respectively; $\boldsymbol{\mu}_{max}$ and $\boldsymbol{\sigma}_{max}$ represent the maximum predictions provided by GP models; α is an adaptation parameter defined by a cosine function. FE denotes the current number of expensive objective function evaluations.

2.3 Feature Selection

Feature selection techniques are widely used in machine learning for dimensionality reduction by removing the irrelevant and redundant features. Feature selection methods can be roughly grouped into three categories: filters, wrappers and embedded methods [3]. Wrappers involve a learning algorithm to select a best feature subset. As a result, they require high computational costs to achieve good performance. On the contrary, filters are characterized by their low computational costs, although the performance may become slightly worse. In evolutionary computation, feature selection techniques are employed to tackle high-dimensional problems, including both high-dimensional decision space and objective space. For example, feature selection is adopted to select a subset of decision variables for constructing ensemble surrogates [10]. In many-objective optimization problems, a lot of algorithms have employed feature selection methods for reducing the number of objectives [17, 18]. In this work, we employ feature selection to identify a subset of decision variables that are most relevant to both expensive and cheap objectives for knowledge transfer.

2.4 Transfer Learning in Modeling of Gaussian Processes

Recently, many efforts have been dedicated to transfer learning in GPs. In general, transfer learning in GPs can be separated into two main categories. The first one is parameter transfer [5, 25], which transfers knowledge by sharing some parameters of GPs under the assumption that the smoothness of the learning models of source and target tasks are similar. For example, Yu *et al.* composed different tasks within a hierarchical Bayesian modeling, where a normal-inverse Wishart prior distribution is introduced to capture information extracted from each task [25]. The second category is called inductive transfer, where the tasks exchange information by correlating the outputs of different tasks [22]. Most attempts of transfer learning based on GPs focus on multi-task learning, which is different from our optimization problem and transfer learning for optimization [4]. Regarding the input of the GPs, different instances can be extracted from each

task in multi-task learning; however, for the bi-objective optimization problems considered in this work, most candidate solutions (inputs of surrogates) for both objective are the same due to the common initial population and the same decision space. As to the outputs of the GPs, it is assumed in multi-task learning that the outputs of a set of task are correlated, while in this work, the correlation of the two objectives becomes stronger when the solutions are closer to the Pareto front. In other words, the correlation between the two objectives may be weak in the beginning of optimization.

3 PROPOSED METHOD

To address MOPs for objectives with different evaluation times, we resort to the Bayesian approach to SAEA, where GPs are used as surrogates and an acquisition function is employed for model management. In addition, we want to address the following two questions. First, considering that the allowed maximum numbers of function evaluations for the two objectives are different, how to make full use of the given budget for each objective to explore promising solutions? Second, how to exploit the common knowledge between the two objectives so that the surrogate for the expensive objective function can benefit from the data of the cheap objective function? In this work, we propose to perform additional single-objective optimization of the cheap objective, while introducing parameter-based transfer learning to achieve the above two goals. The flowchart of the proposed algorithm is shown in Fig. 1 and the pseudocode is listed in Algorithm 1.

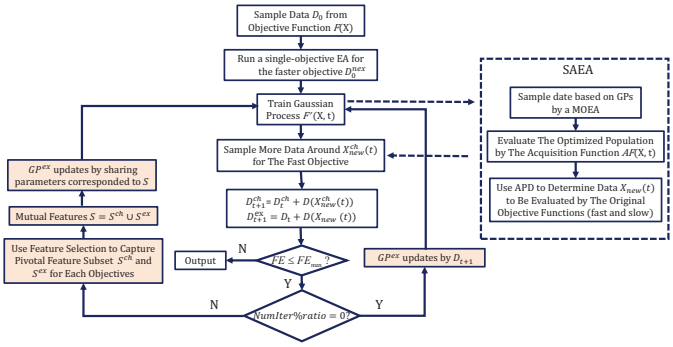

Figure 1: The framework of the proposed algorithm

In the first step, Latin hypercube sampling (LHS) is used to generate a predefined number of individuals as the initial population, where both objectives are evaluated using the real objective function. Since it takes much longer time for evaluating the expensive objective, a single-objective search is carried out on the cheap objective function while waiting for the evaluations of the expensive objective function (y^{ex}) to be completed. Here, the solutions evaluated on the expensive objective are stored in an archive D_{ex}, which is the training data set for constructing a GP model GP^{ex} for y^{ex}. Similarly, solutions evaluated only on the cheap objective function are stored in D_{ch} used to train a GP GP^{ch} for y^{ch}. Different from conventional SAEAs in which we always have the same number of training data for all objectives, in this work, we have much more data in D_{ch} than in D, due to the fact that more function evaluations can be done on the cheap objective function. This will apparently result in different degrees of accuracy for the two surrogates, if they are constructed completely separately. However, if there is a correlation between the two objectives, and if we are able to transfer knowledge of the cheap objective function to the surrogate of the expensive objective function, we will be able to enhance the quality of the surrogate for the expensive objective function trained on a very limited amount of training data. This is the main motivation to introduce transfer learning in GP-based modeling of the two objectives. Once the surrogates for the two objective functions are constructed, similar to the canonical SAEAs, a multiobjective EA (we use the RVEA [7]) is employed to search new samples based on

Algorithm 1: Framework of T-SAEA

Input : $MaxFE^{ex}$: the maximum number of fitness evaluations for y^{ex}; τ: the ratio of the evaluation times of the two objectives; u: the number of new samples to update the GPs; w_{max}: the maximum number of generations before updating GPs

Output : The final solution population in the archive D

Initialization: Use the LHS method to sample an initial population D_0 and set $D_0^{ch} = D_0$; Run a single-objective EA to optimize y^{ch} and save solutions in D_0^{ch}; set $FE^{ex} = size(D_0)$ and $IterNum = 1$.
while $FE^{ex} \leqslant MaxFE^{ex}$ **do**
  Train GP^{ch} with the training data D^{ch} for y^{ch};
  if $IterNummod\tau == 0$ **then**
    Train GP^{ex} with the training data D for y^{ex};
  else
    Use the filter-based feature selection to select the relevant features;
    Use the Eq.(17) to update GP^{ex};
  end
  while $w \leqslant w_{max}$ **do**
    `/* Using the surrogate in the EA */`
    Run an EA to find samples for updating GPs;
    $w = w + 1$;
  end
  Call ALCB to evaluate the individuals in the optimized population;
  Call APD to determine u points x_{new}^{ex} to be evaluated by y^{ex} and add x_{new}^{ex} to D;
  Use LHS to sample $\tau * u$ points x_{new}^{ch} around x_{new}^{ex} to be evaluated by y^{ch} and add solutions to D^{ch};
  Update $FE^{ex} = FE^{ex} + u$, $IterNum = IterNum + 1$;
end
Return The final solution population D;

two GP surrogates. After obtaining the optimal population via the RVEA, the above-mentioned ALCB is adopted to assess each individual and APD suggested in RVEA is then employed to determine predefined number of samples x_{new}^{ex}.

When evaluating the expensive objective function on the selected x_{new}^{ex}, additional new samples x_{new}^{ch} for the cheap objective can be collected by sampling around x_{new}^{ex} with the LHS method. While GP^{ch} is updated on D_{ch}, GP^{ex} is updated using two different methods alternately. Specifically, for τ (which is an integer larger than 1 defined in Section 1) iterations, GP^{ex} is updated once using the data in D_{ex}. Then, the parameters in GP^{ex} corresponding to the decision variables identified by the feature selection method are updated for $\tau-1$ iterations using the parameter sharing method in (16).

In the following, we first describe the filter-based feature selection methods adopted to select the most relevant parameters to share between the two surrogates. Then an adaptive aggregation method is suggested to share the parameters of the two GP models. A more detailed description of the feature selection algorithm and the parameter sharing method will be given below.

3.1 Filter-based Feature Selection

As mentioned above, for a bi-objective optimization problem with delayed objectives, the limited training data samples for the expensive objective function are usually insufficient for constructing a high-quality surrogate GP^{ex}. By contrast, the training data for GP^{ch} is relatively sufficient; hence a more accurate surrogate model is available for y^{ch}. With this in mind, we introduce a transfer learning method to improve the accuracy of GP^{ex} by transferring knowledge from GP^{ch} to GP^{ex}. However, it is worth noting that knowledge transfer is beneficial only if there is a strong similarity or correlation between the source and target tasks; otherwise, negative transfer may occur [19]. To avoid negative transfer, a filter-based feature selection is employed to identify common decision variables related to both y^{ex} and y^{ch}, working as a carrier to transfer knowledge between the two surrogates. To this end, a distance-based correlation is used to measure the relevance R_1 of the selected features $\mathbf{X_s} = \{\mathbf{x_s}^1, \mathbf{x_s}^2, ...\mathbf{x_s}^N\}^T$ to objectives $\mathbf{Y} = \{\mathbf{y}^1, \mathbf{y}^2, ...\mathbf{y}^N\}^T$ and the redundancy R_2 of $\mathbf{X_s}$ [10]. Our aim is to detect features (decision variables) with the maximum relevance while minimizing the redundancy. Therefore, the following objective function is adopted for feature selection [6]

$$\min\ Fitness = \alpha * R2 - (1-\alpha) * R1 \tag{13}$$

where $\alpha \in [0,1]$ is a constant, indicating the relative importance of the two components. Given $\mathbf{X_s}$ and $\mathbf{Y}$, two $N \times N$ matrices based on the distance can be calculated by

$$\begin{aligned} &a_{i,j} = \|\mathbf{x_s^i} - \mathbf{x_s^j}\|^2, b_{i,j} = \|\mathbf{y^i} - \mathbf{y^j}\|^2 \\ &A_i^j = a_i^j - \bar{a}_{i.} - \bar{a}_{.j} - \bar{a}_{..} \\ &B_i^j = b_i^j - \bar{b}_{i.} - \bar{b}_{.j} - \bar{b}_{..} \\ &i,j = 1,2,...,N \end{aligned} \tag{14}$$

where $\bar{a}_{i.}$, $\bar{a}_{.j}$ and $\bar{a}_{..}$ denotes the mean of ith row, jth column and the whole matrix a, respectively. Subsequently, the distance covariance is obtained by

$$dCov(\mathbf{X}_s, \mathbf{y}) = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} A_i^j B_i^j \tag{15}$$

Thus, the distance correlation between $\mathbf{X_s}$ and $\mathbf{y}$ is the relevance R_2, defined as

$$dCor(\mathbf{X}_s, \mathbf{y}) = \frac{dCov(\mathbf{X}_s, \mathbf{y})}{\sqrt{dCov(\mathbf{X}_s, \mathbf{X}_s) dCov(\mathbf{y}, \mathbf{y})}} \tag{16}$$

where the $dCov(\mathbf{X}_s, \mathbf{X}_s)$ and $dCov(\mathbf{y}, \mathbf{y})$ are defined by Eq.(15). Redundancy R_2 can be defined in a similar way.

The binary particle swarm optimisation algorithm in [16] is utilized to identify the the relevant but distinct features, which paves the way for parameter-based transfer learning between the surrogate models.

3.2 Parameter-based Transfer Learning

From the model perspective, the (hyper-) parameters of a model reflect the knowledge learned by the model, which makes it possible to transfer knowledge by sharing parameters of different models [28]. Here, we adopt parameter-based transfer learning in building GP^{ex} by sharing some of its parameters with that of GP^{ch}. In the previous section, we apply a feature selection method to identify the most relevant and compact subset of the parameters to be shared to improve the performance of GP^{ex}. In the Gaussian process considered in this work, the hyperparameter θ_k in $\theta = [\theta_1, \theta_2, ..., \theta_n]$ given in the kernel function denotes the importance of the corresponding kth feature (decision variable). Intuitively, such a connection between the features and the hyperparameters in the GP models makes it possible to transfer knowledge in the selected features at the parameter level. More precisely, denoting the hyerparameters in GP^{ex} and GP^{ch} corresponded to the selected features at the tth iteration as $\boldsymbol{\theta}_t^{ex}$ and $\boldsymbol{\theta}_t^{ch}$, respectively. $\boldsymbol{\theta}_t^{ex}$ can then be calculated by

$$\boldsymbol{\theta}_{t+1}^{ex} = (1-\alpha)\cdot(\boldsymbol{\theta}_t^{ch}) + \alpha \cdot (\boldsymbol{\theta}_t^{ex}) \tag{17}$$

where

$$\alpha = -0.5 \cdot \cos(\frac{FE}{MaxFE^{ex}} \cdot \pi) + 0.5 \tag{18}$$

where './' denotes element-by-element division for two vectors of the same size; FE denotes the current number of the function evaluations using the real objective function. α is an adaptation parameter defined by a cosine function, aiming to assign adaptive weights to $\boldsymbol{\theta}^{ex}$ and $\boldsymbol{\theta}^{ch}$ in the aggregation function. Note that the parameter sharing method in the Eq.(17) allows GP^{ex} to learn knowledge from GP^{ch}; however, it does not mean there is a linear correlation between the two GPs, since only a selected subset of hyperparameters is shared. The main motivation behind the adaptive parameter sharing function is to transfer knowledge effectively and avoid negative transfer. In the beginning of the optimization, it is desirable to make use of knowledge from the relatively high-quality model GP^{ch} (since it is trained on much more data) by allocating a larger weight to $\boldsymbol{\theta}_t^{ch}$, as the training data

is insufficient for GP^{ex}. As the optimization proceeds, the number of training samples for GP^{ex} increases, and therefore, the weight for $\boldsymbol{\theta}_t^{ex}$ gradually becomes larger.

4 EXPERIMENTAL RESULTS AND DISCUSSIONS

4.1 Experimental settings

To demonstrate the effectiveness of the proposed algorithm for solving bi-objective optimization problems with delayed objectives, T-SAEA is compared with its two variants, an SAEA variant without the parameter-based transfer learning (termed as NT-SAEA) and an SAEA variant sharing the parameters on all decision variables without the feature selection (termed as NS-SAEA). In addition, we compare T-SAEA with HK-RVEA [8] and three algorithms presented in [1]. Specifically, HK-RVEA is the state-of-the-art method for problems we are addressing in this work; waiting, fast-first and speculative interleaving [1] are three distinct schemes identified as delayed-handling strategies. Note that we adopt RVEA as the multi-objective optimizer, and the canonical genetic algorithm (GA) as the single-objective optimizer in all the compared algorithms for the sake of fairness. The first set of test functions we used are from DTLZ [9], including DTLZ1-DTLZ7 and two modified counterparts of DTLZ1 and DTLZ3, where the number/dimension of decision variables (features) is set to $n = M + K - 1$: $K = 5$ is adopted for DTLZ1 and DTLZ1a, $K = 10$ is used for DTLZ2 to DTLZ6, and $K = 20$ is employed in DTLZ7. M represents the number of objectives, in this work, we set $M = 2$. The second set of test functions are seven bi-objective problems from the UF test suite [27], where the dimension is fixed as 30. In all the benchmarks, we assume f_2 (objective 2) is the expensive objective f^{ex}.

We suppose the total budget is limited to 200 fitness evaluations of the expensive objective function, i.e., $MaxFE^{ex} = 200$. In this way, we can determine the number of fitness evaluations we can afford, $MaxFE^{ch}$, as long as we know τ. Here, we set $\tau = 5, 10$. The initial population size is 100, the number of new samples to be collected for the expensive objective is 3, and each algorithm is run independently on each test problem for 20 times. The inverted generational distance (IGD) [26], which accounts for both convergence and diversity of the obtained set of solutions, is adopted to assess the performance of the compared algorithms. The standard deviation (std) and mean of IGD values over 20 times are collected and presented in Tables 1 and 2, and the mean of the results are adopted for comparison using the Wilcoxon rank sum test, where symbol "(+)", "(−)" and "(≈)" indicate that the proposed algorithm performs better than, worse than, or comparable to the compared algorithm, respectively. The best result of each benchmark function is highlighted.

4.2 Comparative experiments

In the first empirical study, we set $\tau = 5$ and $MaxFE^{ex} = 200$, thus we have $MaxFE^{ch} = 1000$. The results obtained by each compared algorithm in terms of convergence and diversity, indicated by IGD values (the lower the better) are presented in Table 1. Several observations can be made from the above results. Firstly, the proposed algorithm T-SAEA achieves the best overall results. Except on DTLZ1a, DTLZ4, DTLZ5, UF4 and UF7, T-SAEA significantly outperforms all compared algorithms on the rest eleven test instances due to its delay-handling strategy. In addition to the better result on DTLZ4 and UF4 yielded by waiting, it is important to observe that T-SAEA shows the superiority on all the benchmarks over waiting, fast-first and speculative interleaving strategies. Secondly, the efficiency of the proposed parameter-based transfer learning and the feature selection in T-SAEA can be demonstrated in comparison with NT-SAEA and NS-SAEA, respectively. Note that T-SAEA shows competitive performance on most test problems while NT-SAEA achieves good results on three test instances (DTLZ1a, DTLZ4 and UF4) only, indicating that T-SAEA can benefit from transferring the useful knowledge in surrogate modeling. Note that the effectiveness of transfer learning is sensitive to the relationship between the source and target tasks and it makes good sense to apply parameter-based transfer learning only to those most relevant features. Similarly, the proposed T-SAEA achieves better performance than NS-SAES on most test problems except for DTLZ1a and DTLZ5, which confirms the importance of identifying the transferable parameters. Compared with HK-RVEA, despite HK-RVEA can yield good results on some test problems, its performance degrades considerably in addressing the test problems with high complexity and ruggedness, such as DTLZ1, DTLZ3 and DTLZ3a. By contrast, T-SAEA can scale up its performance to complex problems. We want to point out that fastfirst, speculative interleaving and HK-RVEA can sort of utilize transfer learning by using the extra solutions evaluated on the cheap objective function to guide the search for the expensive objective. Due to the straightforward transformation of the instances from the data perspective, these strategies, as shown in the table, are not able to solve the problems as effectively as T-SAEA.

To further investigate how the knowledge transfer strategies are impacted as the delay length increases, we set $\tau = 10$, while the allowed computational budget remains unchanged, i.e., $MaxFE^{ex} = 200$ and $MaxFE^{ch} = 2000$. As a result, the knowledge gap between y^{ex} and y^{ch} may increase. Note that the correlation between the two objectives in the DTLZ and UF benchmarks is unknown and cannot be controlled, which reflects some real-world applications well. With the increase on the information transferred from the non-delayed objective to the delayed one, it is not surprising to observe from Table 2 that each algorithm performs better on some test problems and worse on others when compared with the results for $\tau = 5$. For example, when $MaxFE^{ch}$ increases from 1000 to 2000, the performance of HK-RVEA is affected, improving slightly on some test problems at the expense of degraded performance on over half of the test problems. Similar phenomena can be seen on fastfirst and speculative interleaving approaches. By contrast, T-SAEA performs consistently well on most problems, which confirms the efficiency of the proposed strategies. Overall, it is obvious that T-SAEA is

Table 1: Statistical results of the IGD values obtained by Waiting, Fastfirst, Speculative interleaving, HK-RVEA, NT-SAEA, NS-SAEA and T-SAEA with $MaxFE^{ex} = 200$ and $\tau = 5$

Test problem	Waiting		Fastfirst		Interleaving		HK-RVEA		NT-SAEA		NS-SAEA		T-SAEA	
	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
DTLZ1	30.21	+ 14.26	69.68	+ 24.14	48.06	+ 10.45	5.93e4	+ 1.20e4	46.26	+ 12.90	60.79	+ 14.76	**21.67**	11.92
DTLZ1a	14.24	+ 8.484	60.5	+ 31.52	32.59	+ 13.98	20.46	+ 24.82	**0.875**	≈ 0.138	0.678	– 0.723	1.062	0.998
DTLZ2	0.240	+ 0.049	0.803	+ 0.075	0.380	+ 0.033	0.244	+ 0.005	0.052	≈ 0.010	0.049	≈ 0.028	**0.046**	0.027
DTLZ3	309.2	+ 83.4	549.1	+ 141.8	462.3	+ 67.45	7.53e6	+ 1.99e6	321.5	+ 140.4	371.4	+ 208.4	**202.5**	100.4
DTLZ3a	227.2	+ 75.41	545.9	+ 86.86	406.3	+ 96.6	4.13e4	+ 3.51e4	10.08	+ 14.99	8.65	+ 33.9	**5.340**	37.54
DTLZ4	0.509	– 0.324	0.776	+ 0.114	0.652	+ 0.104	0.379	– 0.176	**0.285**	– 0.516	0.581	≈ 0.254	0.600	0.133
DTLZ5	0.270	+ 0.064	0.855	+ 0.101	0.391	+ 0.029	0.141	+ 0.051	0.240	+ 0.004	**0.036**	– 0.028	0.049	0.017
DTLZ6	7.306	+ 0.524	8.793	+ 0.11	8.258	+ 0.126	2.748	+ 0.002	6.597	+ 7.177	6.439	+ 0.807	**2.558**	1.207
DTLZ7	4.413	+ 0.623	7.532	+ 0.391	5.565	+ 0.681	1.705	+ 1.710	1.288	+ 0.401	4.676	+ 0.574	**1.147**	0.912
UF1	1.011	+ 0.135	0.494	+ 0.036	0.424	+ 0.045	0.244	+ 0.037	1.351	+ 0.026	0.676	+ 0.539	**0.189**	0.016
UF2	0.500	+ 0.073	0.582	+ 0.085	0.506	+ 0.026	0.161	+ 0.012	0.615	+ 0.052	0.302	+ 0.084	**0.140**	0.016
UF3	0.967	+ 0.078	1.224	+ 0.061	1.083	+ 0.067	0.591	+ 0.053	0.552	+ 0.012	0.595	+ 0.158	**0.189**	0.077
UF4	0.209	– 0.007	0.241	+ 0.018	0.232	+ 0.005	0.215	≈ 0.005	**0.189**	– 0.007	0.196	≈ 0.019	0.225	0.006
UF5	4.747	+ 0.417	3.529	+ 0.257	3.254	+ 0.158	4.661	+ 0.396	4.324	+ 1.069	4.664	+ 1.312	**2.494**	0.438
UF6	4.358	+ 0.629	2.277	+ 0.238	1.993	+ 0.167	1.521	+ 0.327	4.134	+ 3.061	2.084	+ 0.591	**1.009**	0.251
UF7	1.113	+ 0.176	0.580	+ 0.064	0.472	+ 0.053	**0.284**	– 0.034	0.500	+ 0.091	0.455	+ 0.142	0.365	0.063

Table 2: Statistical results of the IGD values obtained by Waiting, Fastfirst, Speculative interleaving, HK-RVEA, NT-SAEA, NS-SAEA and T-SAEA with $MaxFE^{ex} = 200$ and $\tau = 10$

Test problem	Waiting		Fastfirst		Interleaving		HK-RVEA		NT-SAEA		NS-SAEA		T-SAEA	
	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
DTLZ1	30.2	+ 14.26	102.7	+ 37.52	44.22	+ 15.49	6.10e4	+ 1.08e4	50.66	+ 20.35	35.62	+ 23.76	**11.23**	14.73
DTLZ1a	14.24	+ 8.484	68.43	+ 36.66	36.10	+ 12.65	38.20	+ 23.58	**0.751**	– 0.097	1.247	+ 2.509	1.616	4.057
DTLZ2	0.240	+ 0.049	0.966	+ 0.107	0.386	+ 0.037	0.239	+ 0.006	**0.026**	– 0.003	0.048	≈ 0.523	0.041	0.020
DTLZ3	309.2	+ 83.4	655.9	+ 122.2	414.7	+ 82.84	6.88e6	+ 1.52e6	381.62	+ 136.6	400.1	+ 99.45	**136.8**	99.33
DTLZ3a	227.2	+ 75.41	597.8	+ 139.3	429.8	+ 86.94	5.08e4	+ 7.18e4	11.563	+ 12.55	9.186	+ 19.98	**0.499**	6.401
DTLZ4	0.509	≈ 0.324	0.748	+ 0.133	0.663	+ 0.121	0.512	≈ 0.193	0.671	+ 0.302	**0.426**	– 0.169	0.544	0.208
DTLZ5	0.270	+ 0.064	0.968	+ 0.099	0.410	+ 0.031	0.239	+ 0.003	**0.033**	– 0.006	0.046	– 0.051	0.116	0.013
DTLZ6	7.306	+ 0.524	8.972	+ 0.177	8.261	+ 0.156	**2.974**	– 0.626	6.462	+ 1.177	6.539	+ 0.767	3.993	1.407
DTLZ7	4.413	+ 0.623	7.763	+ 0.701	5.757	+ 0.574	1.704	+ 0.001	0.129	+ 0.101	3.471	+ 0.474	**1.657**	1.053
UF1	1.011	≈ 0.135	0.541	+ 0.065	0.408	+ 0.038	0.227	+ 0.046	0.439	+ 0.258	0.568	+ 0.319	**0.193**	0.012
UF2	0.500	+ 0.073	0.760	+ 0.099	0.523	+ 0.035	0.161	+ 0.010	0.270	+ 0.012	0.316	+ 0.067	**0.151**	0.008
UF3	0.967	+ 0.078	1.214	+ 0.071	1.091	+ 0.075	0.607	≈ 0.072	0.590	+ 0.082	0.595	≈ 0.158	**0.602**	0.068
UF4	0.209	≈ 0.007	0.585	+ 0.015	0.233	+ 0.005	0.213	≈ 0.004	**0.188**	– 0.005	0.189	– 0.004	0.212	0.003
UF5	4.747	+ 0.417	3.456	+ 0.276	3.236	+ 0.189	4.858	+ 0.393	3.783	+ 0.864	4.949	+ 0.712	**2.949**	0.332
UF6	4.358	+ 0.629	2.201	+ 0.251	2.012	+ 0.181	1.761	+ 0.267	2.311	+ 0.065	1.258	≈ 0.191	**1.009**	0.157
UF7	1.113	+ 0.176	0.536	+ 0.056	0.460	+ 0.051	0.318	≈ 0.037	0.636	+ 0.128	0.497	+ 0.142	**0.317**	0.037

the most competitive approach when $\tau = 10$. Specifically, the proposed algorithm has achieved good performance on most test instances, especially the rugged and complex problems such as DTLZ1, DTLZ1a, DTLZ3 and DTLZ3a. These results confirm the validity of the parameter-based transfer learning and are consistent with the results presented in Table 1.

Based on the above results in the two tables, T-SAEA is the most competitive method among the above-mentioned algorithms for solving the optimization problems when objective functions have different evaluation times or latencies.

5 CONCLUSION

In this paper, we address multi-objective optimization problems where the objectives have very different computational complexities, i.e., very different evaluation times, which are commonly seen in simulation or experiment driven optimization. In particular, we consider bi-objective optimization

problems in which one objective is expensive and another is cheap, and propose a new framework based on a surrogate-assisted MOEA in combination with a parameter-based transfer learning method. The proposed method aims to maximize the use of the budget for the cheap objective to construct an accurate GP model, and a transfer learning technique is proposed to enhance the GP surrogate for the expensive objective function. To improve the effectiveness of knowledge transfer, a filter-based feature selection is used to capture a set of compact and relevant features between the two objectives. The corresponding parameters between the surrogates can thus be shared to improve the quality of the surrogate for the expensive objective function. Experimental results on two benchmark suites have shown that the proposed algorithm, T-SAEA, significantly outperforms the compared state-of-the-art algorithms on most benchmark problems, including the rugged and complex test problems.

As the budget for the cheap objective increases, it is necessary to avoid negative transfer due to a strong bias introduced by the much more intensive search on the cheap objective function. Although the proposed algorithm with the help of the feature selection is able to mitigate possible performance degradation, our future work will aim to gain more insights into transfer learning techniques to more effectively exploit useful knowledge and reduce the chance of negative transfer. In addition, we plan to investigate the impact of the number of expensive (delayed) objectives and the delay length, and extend our efforts to optimization problems with more objectives.

ACKNOWLEDGEMENTS

The work was supported in part by a Royal Society International Exchanges Program under No. IEC\NSFC\170279.

REFERENCES

[1] Richard Allmendinger, Julia Handl, and Joshua Knowles. 2015. Multiobjective optimization: When objectives exhibit non-uniform latencies. *European Journal of Operational Research* 243, 2 (2015), 497–513.

[2] Richard Allmendinger and Joshua Knowles. 2013. Hang On a Minute: Investigations on the effects of delayed objective functions in multiobjective optimization. In *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 6–20.

[3] Verónica Bolón-Canedo, Noelia Sánchez-Maroño, and Amparo Alonso-Betanzos. 2013. A review of feature selection methods on synthetic data. *Knowledge and information systems* 34, 3 (2013), 483–519.

[4] Edwin V Bonilla, Kian M Chai, and Christopher Williams. 2008. Multi-task Gaussian process prediction. In *Advances in neural information processing systems*. 153–160.

[5] Bin Cao, Sinno Jialin Pan, Yu Zhang, Dit-Yan Yeung, and Qiang Yang. 2010. Adaptive transfer learning. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*.

[6] Liam Cervante, Bing Xue, Mengjie Zhang, and Lin Shang. 2012. Binary particle swarm optimisation for feature selection: A filter based approach. In *2012 IEEE Congress on Evolutionary Computation*. IEEE, 1–8.

[7] Ran Cheng, Yaochu Jin, Markus Olhofer, and Bernhard Sendhoff. 2016. A reference vector guided evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation* 20, 5 (2016), 773–791.

[8] Tinkle Chugh, Richard Allmendinger, Vesa Ojalehto, and Kaisa Miettinen. 2018. Surrogate-assisted evolutionary biobjective optimization for objectives with non-uniform latencies. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 609–616.

[9] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. 2002. Scalable multi-objective optimization test problems. In *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02*, Vol. 1. IEEE, 825–830.

[10] Dan Guo, Yaochu Jin, Jinliang Ding, and Tianyou Chai. 2018. Heterogeneous ensemble-based infill criterion for evolutionary multiobjective optimization of expensive problems. *IEEE transactions on cybernetics* 49, 3 (2018), 1012–1025.

[11] Shinkyu Jeong, Mitsuhiro Murayama, and Kazuomi Yamamoto. 2005. Efficient optimization design method using kriging model. *Journal of Aircraft* 42, 2 (2005), 413–420.

[12] Yaochu Jin. 2011. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation* 1, 2 (2011), 61–70.

[13] Yaochu Jin, Markus Olhofer, and Bernhard Sendhoff. 2000. On evolutionary optimization with approximate fitness functions. In *Proceedings of the 2nd Annual Conference on Genetic and Evolutionary Computation*. Morgan Kaufmann Publishers Inc., 786–793.

[14] Yaochu Jin and Bernhard Sendhoff. 2009. A systems approach to evolutionary multiobjective structural optimization and beyond. *IEEE Computational Intelligence Magazine* 4, 3 (2009), 62–76.

[15] Donald R Jones, Matthias Schonlau, and William J Welch. 1998. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization* 13, 4 (1998), 455–492.

[16] James Kennedy and Russell C Eberhart. 1997. A discrete binary version of the particle swarm algorithm. In *1997 IEEE International conference on systems, man, and cybernetics. Computational cybernetics and simulation*, Vol. 5. IEEE, 4104–4108.

[17] Bingdong Li, Jinlong Li, Ke Tang, and Xin Yao. 2015. Many-objective evolutionary algorithms: A survey. *ACM Computing Surveys (CSUR)* 48, 1 (2015), 13.

[18] Antonio López Jaimes, Carlos A Coello Coello, and Debrup Chakraborty. 2008. Objective reduction using a feature selection technique. In *Proceedings of the 10th annual conference on Genetic and evolutionary computation*. ACM, 673–680.

[19] Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2009), 1345–1359.

[20] Chiwoo Park and Daniel Apley. 2018. Patchwork kriging for large-scale gaussian process regression. *The Journal of Machine Learning Research* 19, 1 (2018), 269–311.

[21] Carl Edward Rasmussen. 2003. Gaussian processes in machine learning. In *Summer School on Machine Learning*. Springer, 63–71.

[22] Grigorios Skolidis. 2012. Transfer learning with Gaussian processes. (2012).

[23] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. 2009. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995* (2009).

[24] Xilu Wang, Yaochu Jin, Sebastian Schmitt, and Markus Olhofer. 2020. An Adaptive Bayesian Approach to Surrogate-Assisted Evolutionary Multi-objective Optimization. *Information Sciences* (2020).

[25] Kai Yu, Volker Tresp, and Anton Schwaighofer. 2005. Learning Gaussian processes from multiple tasks. In *Proceedings of the 22nd International Conference on Machine Learning (ICML-05)*. 1012–1019.

[26] Qingfu Zhang, Aimin Zhou, and Yaochu Jin. 2008. RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm. *IEEE Transactions on Evolutionary Computation* 12, 1 (2008), 41–63.

[27] Qingfu Zhang, Aimin Zhou, Shizheng Zhao, Ponnuthurai Nagaratnam Suganthan, Wudong Liu, and Santosh Tiwari. 2008. *Multiobjective optimization test instances for the CEC 2009 special session and competition*. Technical Report.

[28] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2019. A Comprehensive Survey on Transfer Learning. *arXiv preprint arXiv:1911.02685* (2019).