Experience-based Computation:
Learning to Optimise

Project Number: 766186
Project Acronym: ECOLE
Project title: Experienced-based Computation: Learning to Optimise

Deliverable D3.4

Online learning for proactive dynamic and robust optimisation

Authors:
Gan Ruan, Leandro L. Minku, Xin Yao – University of Birmingham

Project Coordinator: Professor Xin Yao, University of Birmingham
Beneficiaries: Universiteit Leiden, Honda Research Institute Europe, NEC Laboratories Europe

H2020 MSCA-ITN
Date of the report: **31.03.2021**

Contents

Executive summary

The objective of WP3.4 is to study online learning for proactive dynamic and robust optimisation. The status of this work package within the ECOLE project is presented in this report. Our study on the investigation of when and how to transfer knowledge in dynamic multi-objective opitmisation is first introduced (in Section 2). The existing work in the literature transfers the knowledge whenever a change happens no matter what kinds of changes and what types of problems are involved. Through computational studies, we found that transfer learning fails on problems with fixed Pareto optimal solution sets and under small environmental changes. In addition, we also found that Gaussian kernel function used in the existing transfer learning-based method is not always adequate. Therefore, transfer learning should be avoided when dealing with problems for which transfer learning fails and other kernel functions should be used when the Gaussian kernel is inadequate. Novel strategies and kernel functions are proposed that can be used in such cases. The results show that our proposed approach is able to more proactively and more quickly adapt to changes soon after they happen, rather than spending a large amount of time adapting to changes. In addition to this, the efficiency of transfer learning in dynamic multi-objective optimisation is also studied (in Section 3). Through computation time analysis of transfer learning, we show that the optimisation problem solved within transfer learning ("inner optimization problem") is very time-consuming. In order to enhance the efficiency, two alternatives are computationally investigated on a number of dynamic bi- and tri-objective test problems. The results show that the greatly enhanced efficiency does not result in huge degeneration on the performance of transfer learning. The two proposals in Section 2 and Section 3 are more robust to changes as they are able to generate good solutions in the first generation right after the changes.

Major Achievements

Major scientific achievements concerning the research invested in this deliverable are presented. In particular, short answers to the most important research questions - practical issues - are described:

Research Questions	Discussion
When does transfer fail in dynamic multi-objective optimisation?	Through comparing the quality of transferred solutions and those without transfer on a set of benchmark problems with various environment changes, it is found that the transfer fails on problems with fixed Pareto optimal solution sets and under small environmental changes
Why transfer fails in dynamic multi-objective optimisation?	A mathematical proof on transfer learning-based dynamic multi-objective optimisation demonstrates that the Gaussian kernel function used in the existing transfer learning-based method is not always adequate.
How efficient is transfer learning in dynamic multi-objective optimisation? And why does it perform in such way?	Transfer learning is found to be very time-consuming as the ‘inner’ optimisation method in transfer learning is very costly.
How to improve the efficiency of transfer learning in dynamic multi-objective optimization? Does such improved efficiency negatively affect the quality of transferred solutions?	Two alternative optimisation methods can be used to replace the existing ‘inner’ optimisation method to improve the efficiency of the transfer learning. Experimental results show that the greatly enhanced efficiency does not result in large deterioration of the quality of transfer learning.
Is the computation time saved in optimisation cancelled out by computationally expensive transfer learning?	Experiments using the computational cost of transfer learning to optimise randomly generated solutions show that convergence and diversity of final solutions generated from the randomly generated solutions are significantly better than those generated from transferred solutions under the same total computational budget. Therefore, the high computational cost of transfer cancels out the benefits of the computational time that it can save in optimization.

1. Introduction

In the Experience-based Computation: Learning to Optimise (ECOLE) project, we aim to tackle sophisticated optimisation problems promoted by the emergence of large amounts of product and process data which arise from our increasingly dynamic and interconnected world. Naturally, these optimisation problems seldom exist in isolation as different problems or the same problem in different dynamic states may share some similarity or interconnectivity. Therefore, our program utilizes the notion of experience to develop novel and advanced machine learning and optimisation techniques to assist the optimisation of engineering and real-world problems.

In Work Package 3 (WP3) of the ECOLE project, the research aim is to develop, analyse and evaluate novel machine learning algorithms for more effective and efficient optimisation using system engineering and big data analytics. A sub-Work Package of WP3, WP3.4, focuses on using knowledge extracted from the optimisation process to forecast changes and reduce the search space, leading to novel optimisation algorithms able to find good and timely solutions in uncertain environments, under the assistance of advanced machine learning techniques. One of the representations of experience in solving dynamic optimisation problems are optimal solutions from past problem-solving.

Many real-world engineering and Information and Communications Technology (ICT) [1] [2] optimisation problems operate in dynamic and uncertain environments, requiring solutions to be updated when changes happen. In addition, these optimisation problems typically involve multiple and conflicting objective functions. For example, to support the successful and smooth operation of ICT systems, the role of power generating systems [3] is of great importance. Hydro-thermal power generation system is popular and widely used in modern society, where both the hydroelectric and thermal generating units are utilized to meet the total power demand. The power scheduling optimisation problem in this system involves the allocation of power to all concerned units, so that the total fuel cost of thermal generation and emission properties are minimized, also with the satisfaction of power demand that changes over time. This type of problem is referred to as dynamic multi-objective optimisation problems (DMOPs) [4], whose objectives change over time [5] [6].

The key challenge in DMO is how to constantly trace a changing Pareto optimal front (POF) and/or Pareto optimal set (POS) before the next environment change [9]. Aiming at this goal, researchers have proposed a prediction-based method [9] [10]. This kind of method predicts what the good solutions in the next environment are after learning the regularity of the environment changes. In most prediction-based approaches, it is implicitly assumed that the evolution of the solutions used to train and test the prediction model obeys a fixed independent and identical probability distribution. However, this is not always true under dynamic environments in optimisation, since the environmental changes may result in different evolution patterns over time. Consequently, the prediction model based on the incorrect assumption may cause inaccurate prediction of optimal solutions.

Transfer learning [7] is a kind of online learning method that is able to transfer the knowledge from a source task to a target task. It does not assume that all data used to train and test the prediction model obeys a fixed probability distribution, being a good candidate for solving DMOPs if it can learn and exploit the relationship among different problems. This inherent characteristic of transfer learning makes it intuitive to apply transfer learning to explore useful experience that have been obtained in one task/problem to solve another related task/problem. The reason is that some similar or related problems/tasks may share some common features, which help to transfer experience from one problem to help solving another problem. As a result, computational resources can be significantly saved when solving similar problems later. As we normally assume that the changing problems in DMOPs are related, there are good opportunities for the application of transfer learning in dynamic multi-objective optimisation (DMO). However, there has been so far only one published paper on DMO, introducing transfer learning-based dynamic multi-objective optimisation algorithms (Tr-DMOEAs) [8].

The main idea behind Tr-DMOEA is to transfer solutions in the Pareto front (POF) of the previous environment to generate an initial population for the next environment. Experimental studies [8] have shown the superiority of Tr-DMOEAs over the state-of-the-art in DMO. However, the results also showed that Tr-DMOEA does not always work well, and little is known on why. It is therefore important to understand why and when transfer learning does not work well. Only after understanding it can we make some improvements regarding transfer learning in DMO. In addition, it has been found that the transfer learning process in Tr-DMOEA is computationally expensive. Therefore, how to improve the efficiency is another important research focus in online learning for proactive dynamic optimisation, where new solutions need to be produced very quickly after changes in the environment.

In this report, we present our work on improving the performance of transfer learning in solving DMOPs. We propose novel approaches that are able to quickly enhancing the quality of transferred solutions after changes as well as increasing the efficiency over Tr-DMOEA. Section 2 first presents our in depth analysis of when and how to transfer knowledge in DMO and proposes to transfer when environment changes are smaller and problems have fixed POS, and replace the existing Gaussian kernel with a linear one. Section 3 presents our proposed approach to efficiently transfer knowledge from the past to the present environment state, enhancing solution quality.

2. When and How to Transfer Knowledge in Dynamic Multi-objective Optimisation

This section aims to answer two research questions when and how to transfer knowledge in dynamic multi-objective optimisation. The rest of this section is organized as follows: Section 2.1 answers the question when transfer learning works/fails in DMO and proposes to avoid transfer learning when it fails. Section 2.2 presents how transfer learning works in DMO and propose to replace the Gaussian kernel function with a linear one. Section 2.3 shows experimental studies of the proposal and comparison with the existing methods. Section 2.4 summarises the work of this section.

The following publication in the ECOLE is contributing to this section:

Ruan, G., Minku, L. L., Menzel, S., Sendhoff, B., & Yao, X. (2019, December). When and how to transfer knowledge in dynamic multi-objective optimisation. In 2019 IEEE Symposium Series on Computational Intelligence (SSCI) (pp. 2034-2041). IEEE.

Generally, whenever a new environment change happens, there is a population that has already been optimised to the previous environment. To check whether transfer learning works, we compare the quality of the following solutions on the new environment: (1) transferred solutions and (2) solutions copied from the previously optimised population. For transfer learning to be considered as successful, (1) should be of at least similar (and ideally better) quality than (2) on the new environment. Therefore, this section designs an experiment to compare the quality of the solutions (1) and (2).

Experimental setups of the experiments in this section:
In this report, the IEEE CEC 2015 Benchmark problems [11] are selected as the test problems, which have 12 bi- and tri-objective problems with different features. Inverted Generational Distance (IGD) [12] is used to compare the performance of two solutions sets. IGD can measure the diversity and convergence of a solution set found by an algorithm, so it can give us a comprehensive understanding about the performance of compared algorithms. MIGD [13] is a modified version of IGD, which is the average IGD values in all changes. The smaller the MIGD the better the algorithm. The Wilcoxon rank sum test [14] with the significance level 0.05 is carried out to indicate significance between results obtained by two compared algorithms. MIGD of two compared algorithms for each problem with one parameter is regarded as one observation for the test.

Parameter Settings: the population size is set as 200, as in previous work [8]. For the parameters of these test problems, there are 20 changes. Therefore, the MIGD is the average of IGDs of populations under 20 changes in the following tables. In addition, in these tables, the better values obtained by the algorithm are highlighted in bold face. In order to study the effectiveness of Tr-DMOEAs in different dynamics, there are three dynamics with different severity of change (i.e., n_t = 10, 1 and 20). They represent the environment changes are medium, large and small, respectively. Within each change, the population is forced to run 50 generations (i.e., τ_t = 50), which enables the population to converge. This corresponds to the parameter settings of C4, C6 and C8 in Table 1. At the beginning of the algorithm, the population iterates for 50 generations, which also enables the population to converge.

Table 1: Configurations of benchmark function parameters. n_t, τ_t and τ_T are the severity of change, frequency of change and maximum number of iterations, respectively.

	n_t	τ_t	τ_T
C1	10	5	150
C2	10	10	250
C3	10	25	550
C4	10	50	1050
C5	1	10	250
C6	1	50	1050
C7	20	10	250
C8	20	50	1050

2.1 When Does Transfer Learning Work/Fail in Dynamic Multi-objective Optimisation?

This section analyzes when transfer learning works or fails in DMO, such that some potential improvements can be put forward.

2.1.1 Computational Investigation into Transfer Learning in DMO

Generally, whenever a new environment change happens, there is a population that has already been optimised to the previous environment. To check whether transfer learning works, we compare the quality of the following solutions on the new environment: (1) transferred solutions and (2) solutions copied from the previously optimised population. For transfer learning to be considered as successful, (1) should be of at least similar (and ideally better) quality than (2) on the new environment. Therefore, an experiment is conducted to compare Tr-RMMEDA and RMMEDA where RMMEDA [15] is a regularity model-based multi-objective estimation of distribution algorithm. It is able to make the population converge quickly before the next change, avoiding the cases that unconverged solutions affect the results. The state-of-the-art Tr-DMOEA [8] is adopted.

In DMO, the key point is to find the optimal solutions as soon as possible before next change. In this case, the better the generated solutions after each change, the better the Tr-DMOEA. Therefore, if transferred solutions are better than copied solutions from the previous environment, we consider that transfer learning works well. In the contrary, if transferred solutions are worse than copied solutions, transfer learning fails.

Here, MIGD values for Tr-RMMEDA and RMMEDA are compared. After each environmental change, RMMEDA gets an initial population found in last generation of the previous environment, while Tr-RMMEDA obtains an initial population through transfer learning. These two initial populations are evaluated in the new environment and then used to calculate the IGD values. MIGD values for Tr-RMMEDA and RMMEDA are the average of IGDs under 20 environments. The comparison results are shown in Table 2, in which ‘Tr’. and ‘Copy’ mean MIGD values for Tr-RMMEDA and RMMEDA, respectively. It can be observed from the table that transferred

Table 2:
MIGD for Tr-RMMEDA and RMMEDA

Prob.	C4		C6		C8	
Methods	Tr.	Copy	Tr.	Copy	Tr.	Copy
FDA4	**0.082**	0.129	**0.098**	3.987	0.081	0.081
FDA5	0.389	**0.179**	**1.294**	5.211	0.325	**0.115**
FDA5$_{iso}$	0.280	**0.079**	0.708	**0.597**	0.294	**0.077**
FDA5$_{dec}$	0.645	**0.242**	**3.006**	4.901	0.583	**0.119**
DIMP2	**11.630**	19.361	**11.379**	19.621	**12.466**	22.124
DMOP2	1.531	**0.798**	**35.126**	40.256	1.165	**0.216**
DMOP2$_{iso}$	0.002	0.002	**0.082**	0.111	0.002	0.002
DMOP2$_{dec}$	**1.044**	2.533	**5.182**	61.288	1.028	**0.324**
DMOP3	0.751	**0.472**	**33.648**	37.936	0.656	**0.121**
HE2	0.365	**0.125**	0.187	**0.125**	0.266	**0.123**
HE7	0.309	**0.051**	0.285	**0.042**	0.311	**0.049**
HE9	0.288	**0.238**	0.273	**0.215**	0.263	**0.245**

Table 3:
MIGD of combined and transferred solutions

Prob.	C4		C6		C8	
Methods	Tr.	Comb.	Tr.	Comb.	Tr.	Comb.
FDA4	0.082	**0.070**	0.098	**0.057**	0.081	**0.070**
FDA5	0.389	**0.185**	1.294	**0.226**	0.325	**0.181**
FDA5$_{iso}$	0.280	**0.265**	0.708	**0.226**	0.294	**0.260**
FDA5$_{dec}$	0.645	**0.270**	3.006	**0.295**	0.583	**0.353**
DIMP2	11.630	**3.514**	11.379	**3.793**	12.466	**3.807**
DMOP2	1.531	**0.004**	35.126	**18.035**	1.165	**0.004**
DMOP2$_{iso}$	**0.002**	0.004	**0.082**	0.112	**0.002**	0.004
DMOP2$_{dec}$	1.044	**0.019**	5.182	**0.191**	1.028	**0.017**
DMOP3	0.751	**0.003**	33.648	**18.022**	0.656	**0.003**
HE2	0.365	**0.069**	0.187	**0.056**	0.266	**0.061**
HE7	0.309	**0.040**	0.285	**0.036**	0.311	**0.040**
HE9	0.288	**0.226**	0.273	**0.199**	0.263	**0.237**

solutions are all worse than those from the previous environment on problems HE2, HE7 and HE9, no matter what kinds of changes are present in these benchmark problems. All HE problems have fixed POS. Therefore, it can be concluded that transfer learning fails on problems with fixed POS. For other problems, transferred solutions are better than those copied from the previous environment when $n_t = 1$ (C6). Regarding other two kinds of changes (C4 and C8), cases that transferred solutions are better for C8 are more than those for C4. As a result, transfer learning works better when the environment change is smaller. To sum up, transfer learning works on problems with small changes and with fixed POS. In other words, transfer learning can make the population more proactively adapt to changes more quickly soon after they happen on problems with small changes and with fixed POS.

2.1.2 Avoiding Transfer on Problems with Fixed POS And Small Environmental Changes

Generally, it is unlikely that the error of transferring good solutions from a problem to another problem would be zero. Even if the existing Tr-DMOEA is improved, it is unlikely that the error would become zero. As the copied solutions from the previous environment are very good solutions for situations with small changes or problems with fixed POS, it would be very difficult for transferred solutions to beat the copied ones. Therefore, the most intuitive idea to prevent negative results obtained by transfer learning in such cases would be to prevent using transferred solutions. It has been computationally shown that transfer learning fails on problems with small changes and fixed POS in DMO. In order to improve the performance of transfer learning in DMO, the most intuitive idea is to avoid transfer learning when it fails. In addition, when transfer learning works well, it should be definitely used. However, it is impossible for algorithms to foresee which problem has fixed POS and when the environmental changes are small. To overcome this problem, after each change, transferred solutions and copied ones from the previous environment are firstly combined together. After that, nondominated sort and crowding distance in NSGA-II [16] are used to rank the combined solutions on the new environment. Lastly, solutions with the population size are selected from the combined population as the initial population.

2.1.3 Experimental Studies of the Proposed Strategy

This section analyzes the effectiveness of the strategy proposed in section 2.1.2 through experiments. Here, we compare the proposed strategy against the original transfer learning approach, to check whether it can improve the MIGD values of the initial populations after the changes.

The specific comparison results of transferred to combined solutions are shown in Table 3. The better value that the solution set has is highlighted in bold face. Combined solutions are termed as ‘Comb.’. It is clear that combined solutions are better than transferred ones (Tr.) on almost all test problems under different environments. The Wilcoxon rank sum test result with h = 1 and p = 1.2532e - 4 shows that the combined solutions are significantly better than transferred solutions in the first generation after change. In other words, the combined solutions can adapt to changes more proactively and quickly soon after they happen more than transferred solutions, which further proves the robustness of combined solutions.

2.2 How Does Transfer Learning Work in Dynamic Multi-objective Optimisation?

2.2.1 Mathematical Proof that Gaussian Kernels Are Not Ideal

In this section, we briefly introduce the foundations of Tr-DMOEA first. Then the weakness of using the Gaussian kernel will be highlighted. The detailed process of Tr-DMOEA can be found in [8].

In Domain Adaption Learning (DAL) [17], a transfer learning method, it is assumed that a transformation should be found to a latent space where the difference between the distributions of source and target domain is minimized. Once this transformation is found, it can act as a bridge to connect the source domain and the target domain. Then solutions that have been optimised in one domain can be transferred to be good solutions in another domain through this bridge.

The distance between the distributions of the source and target domain can be calculated through the Maximum Mean Discrepancy (MMD) [18], which evaluates the distance between two distributions in the Reproducing Kernel Hilbert Space (RKHS). Let p and q be two Borel probability distributions defined on a domain X . $FS=\{Fs_1,...,Fs_m\}$ and $FT=\{Ft_1,...,Ft_m\}$ are observations drawn from p and q. Let F be a class of functions $f:\mathrm{X}\rightarrow R$. f can be written as $f(x)=\langle\phi(x),f\rangle$ in a RKHS, where $\phi(x):\mathrm{X}\rightarrow H$. The estimated MMD in RKHS can be calculated as:

$$MMD(\mathrm{F}\ ,p,q):=\left\|\frac{1}{m}\sum_{i=1}^{m}\phi(Fs_i)-\frac{1}{n}\sum_{i=1}^{n}\phi(Ft_i)\right\|_H^2 \qquad (1)$$

In Tr-DMOEA, the distribution of the source and target domains in consideration is the distribution of the objective vectors of source and target solutions. Therefore, in Tr-DMOEA, FS and FT are the objective vectors of randomly generated solutions in the source environment s (i.e., the problem before a change) and target environment t (i.e., the problem after a change), respectively. The function ϕ is defined as $\phi(F) = W^T \kappa(F)$, where W is a transformation matrix which maps the objective vector into the latent space, and $\kappa(F)$ is defined as follows, where $\kappa(\cdot,\cdot)$ is a kernel function [19]:

$$\kappa(F) = [\kappa(Fs_1, F), ..., \kappa(Fs_m, F), \kappa(Ft_1, F), ..., \kappa(Ft_n, F)]^T \tag{2}$$

Once W is found, Tr-DMOEA will initialize the population in the target environment with solutions whose objective vectors are close to that of any good solution from the source environment in the latent space. For that, it needs to find solutions t_k whose objective vector is close to that of a solution s_l from POF of the problem in the source environment (POF_s), i.e., whose $\|\phi(Fs_l) - \phi(Ft_k)\|$ is minimal. We can expand the distance as follows:

$$\|\phi(Fs_l) - \phi(Ft_k)\| = \|W^T \kappa(Fs_l) - W^T \kappa(Ft_k)\| = \|W^T [\kappa(Fs_l) - \kappa(Ft_k)]\| \tag{3}$$

$$= \sum_{j=1}^{d} \left(\phi_j(Fs_l) - \phi_j(Ft_k)\right)^2 = \sum_{j=1}^{d} \left\{ \sum_{i=1}^{m+n} W_{ji} \times \left(\kappa(F_i, Fs_l) - \kappa(F_i, Ft_k)\right) \right\}^2 \tag{4}$$

in which d is the dimension of the latent space; $F = Fs$ when $i \in [1, m]$ and $F = Ft$ when $i \in [m+1, m+n]$.

From the above we can see that the more similar $\kappa(F_i, Fs_l)$ and $\kappa(F_i, Ft_k)$ are, the smaller the distance in the latent space. In the existing Tr-DMOEA, the used kernel function is the Gaussian kernel:

$$\kappa(F, Ft_k) = e^{-(F - Ft_k)^T (F - Ft_k)} \tag{5}$$

Here, we consider bi-objective problems: $Fs_l = (Fs_l^1, Fs_l^2)$; $Ft_k = (Ft_k^1, Ft_k^2)$; $F = (f^1, f^2)$; therefore:

$$\kappa(F_i, Fs_l) - \kappa(F_i, Ft_k) = e^{-(Fs_l^1 - F_i^1)^2 - (Fs_l^2 - F_i^2)^2} - e^{-(Ft_k^1 - F_i^1)^2 - (Ft_k^2 - F_i^2)^2} \tag{6}$$

Then, the difference between $\kappa(F_i, Fs_l)$ and $\kappa(F_i, Ft_k)$ is the difference between their exponent:

$$\left| \left((Fs_l^1 - F_i^1)^2 + (Fs_l^2 - F_i^2)^2\right) - \left((Ft_k^1 - F_i^1)^2 + (Ft_k^2 - F_i^2)^2\right) \right| \tag{7}$$

$$= \left| \left((Fs_l^1 - F_i^1)^2 - (Ft_k^1 - F_i^1)^2\right) + \left((Fs_l^2 - F_i^2)^2 - (Ft_k^2 - F_i^2)^2\right) \right| \tag{8}$$

To make eq. (7) minimal, both terms of eq (8) should be minimal. To simplify the analysis, only the first term is considered here. When the first term is equal to 0, it can be re-written as follows:

$$\left|Fs_l^1 - F_i^1\right| = \left|Ft_k^1 - F_i^1\right| \Rightarrow Fs_l^1 - F_i^1 = Ft_k^1 - F_i^1 \textbf{ OR } Fs_l^1 - F_i^1 = F_i^1 - Ft_k^1 \tag{9}$$

Therefore, solutions for Ft_k^1 are (Ft_k^2 is the similar as Ft_k^1):

$$Ft_k^1 = Fs_l^1 \textbf{ OR } Ft_k^1 = 2F_i^1 - Fs_l^1 \tag{10}$$

Therefore, objective values of found solutions are close to either those of the solution Fs_l from POF_s or twice over those of randomly generated solutions (F_i). The original intention of Tr-DMOEA was to find a solution in the target environment whose objective vector Ft_k^1 k is similar to that of a POF_s solution in the latent space (i.e., the first solution in Eq. (10)). However, there is no reason to believe that a solution in the target domain whose objective vector is similar to a random solution from the source domain in the latent space (i.e., the second solution in Eq. (10)) is going to be a good solution. In this case, the Gaussian kernel function is not the ideal in present version of Tr-DMOEA.

2.2.2 Replacing Gaussian Kernel with Linear Kernel

After reviewing present common kernel functions, we find that the linear kernel functions [20] [21] [22] overcome the problem presented by the Gaussian kernel function explained in section 2.2.1. In the following, the details why the linear kernel overcomes this problem are explained. Here, the simplest linear kernel is used: $\kappa(F, Ft_k) = F^T Ft_k$. Therefore,

$$\kappa(F_i, Fs_l) - \kappa(F_i, Ft_k) = F_i^1(Fs_l^1 - Ft_k^1) + F_i^2(Fs_l^2 - Ft_k^2) \tag{11}$$

It can be seen from eq (4) when the distance is minimal, $\kappa(F_i, Fs_l) - \kappa(F_i, Ft_k)$ would be close to 0. In this case, $Ft_k^1 = Fs_l^1$ and $Ft_k^2 = Fs_l^2$. Therefore, the searched solution $Ft_k = (Ft_k^1, Ft_k^2)$ will be close to the solution $Fs_l = (Fs_l^1, Fs_l^2)$ in last POF_s, being a potentially good solution to initialize the population in the target domain.

2.2.3 Experimental Evaluation of Different Kernels

In section 2.2.2, it has been mathematically proved that the linear kernel function overcomes the problem of the Gaussian kernel. In order to verify the effectiveness of it from the perspective of experiment, specific experiment will be conducted. In this experiment, two algorithms will be compared. One is Tr-DMOEA with the Gaussian kernel function. Another is Tr-DMOEA with the linear kernel function.

The comparison results of the Gaussian and linear kernel based Tr-DMOEA are shown in Table 4. In the Table 4, ‘Gauss.’ and ‘Lin.’ represent the Gaussian and linear kernel based Tr-DMOEA, respectively. It can be seen from the table that when change is small (C8) and medium (C4), Tr-DMOEA with linear kernel function is better than that with Gaussian kernel function except HE9 problem. Plus, when changes are large (C6), there are only 4 out of 12 cases that linear kernel-based Tr-DMOEA is worse than Gaussian kernel-based Tr-DMOEA. As a whole, the linear kernel greatly improves the performance of solution quality after change in the first generation, compared with the Gaussian kernel one. The Wilcoxon rank sum test result with h = 1 and p = 0.0132 shows that transferred solutions with the linear kernel are significantly better than those with the Gaussian one in the first generation after change. In other words, transferred solutions with the linear kernel can adapt to changes more proactively and quickly soon after they happen more than those with the Gaussian one, which further proves the robustness of transferred solutions with the linear kernel.

Table 4:
MIGD for Gaussian kernel-based and linear kernel-based Tr-RMMEDA

Prob.	C4		C6		C8	
Methods	Lin.	Gauss.	Lin.	Gauss.	Lin.	Gauss.
FDA4	**0.052**	0.082	**0.048**	0.098	**0.052**	0.081
FDA5	**0.238**	0.390	**0.825**	1.295	**0.163**	0.325
FDA5$_{iso}$	**0.204**	0.280	**0.628**	0.708	**0.185**	0.294
FDA5$_{dec}$	**0.464**	0.645	**0.641**	3.006	**0.265**	0.583
DIMP2	**10.909**	11.630	**10.527**	11.379	**12.076**	12.466
DMOP2	**0.006**	1.5313	**33.085**	35.126	**0.003**	1.166
DMOP2$_{iso}$	0.002	0.002	0.087	**0.082**	0.002	0.002
DMOP2$_{dec}$	**0.053**	1.044	9.107	**5.182**	**0.070**	1.028
DMOP3	**0.003**	0.751	35.076	**33.648**	**0.003**	0.656
HE2	**0.115**	0.365	**0.096**	0.187	**0.071**	0.266
HE7	**0.301**	0.308	**0.265**	0.285	**0.300**	0.311
HE9	0.303	**0.288**	0.297	**0.273**	0.294	**0.263**

Table 5:
MIGD for the original Tr-RMMEDA and the improved Tr-RMMEDA

Prob.	C4		C6		C8	
Methods	Tr.	ImTr.	Tr.	ImTr.	Tr.	ImTr.
FDA4	0.082	**0.070**	0.098	**0.058**	0.081	**0.071**
FDA5	0.389	**0.285**	1.294	**0.822**	0.325	**0.249**
FDA5$_{iso}$	**0.280**	0.344	0.708	**0.633**	**0.294**	0.336
FDA5$_{dec}$	0.645	**0.327**	3.006	**0.507**	0.583	**0.254**
DIMP2	**11.630**	12.093	**11.379**	**12.163**	12.466	11.860
DMOP2	1.531	**0.006**	35.126	**32.984**	1.165	**0.003**
DMOP2$_{iso}$	**0.002**	**0.004**	0.082	0.102	**0.002**	0.004
DMOP2$_{dec}$	1.044	**0.052**	**5.182**	10.026	1.028	**0.024**
DMOP3	0.751	**0.004**	33.648	**33.219**	0.656	**0.004**
HE2	0.365	**0.115**	0.187	**0.136**	0.266	**0.112**
HE7	0.309	0.309	**0.285**	0.292	**0.311**	0.319
HE9	0.288	**0.255**	**0.273**	0.275	0.263	**0.254**

2.3 Improved Transfer Learning in DMO

It has been mathematically and experimentally shown that when and how to transfer is vitally important in DOM. Transfer learning fails on problems with stationary POS and when changes are small. The kernel function also matters in Tr-DMOEA, which contributes the idea of replacing the Gaussian kernel with a linear one. Therefore, in order to make improvements on present Tr-DMOEA, this section proposes novel method for Tr-DMOEA.

Algorithm 1. **Responding strategy based on improved transfer learning.** (The highlighted text corresponds to the differences between the original and improved Tr-DMOEA.)

```
Input: Two DMOPs $F_s(\cdot)$ and $F_t(\cdot)$ in the source and target environments; $POS_s$ and $POF_s$
       of the DMOP in the source environment; linear kernel function $\kappa$.
Output: The responding initial population $\mathbf{P}_{init}$.
1. Initialization;
2. Randomly sample two solution sets in the search space of problems $F_s(\cdot)$ and $F_t(\cdot)$, as $X_s$ and $X_t$;
3. Evaluate $X_s$ and $X_t$ on their own objective functions to get their objective vectors $\mathbf{F}_s$ and $\mathbf{F}_t$;
4. Use $\mathbf{F}_s$ and $\mathbf{F}_t$ as well as the linear kernel function to get the transformation matrix W [8];
5. Determine target domain solutions $X_{Tr}$ whose fitness is similar to that of the POFs solutions in
   the latent space. The linear kernel function is used here to map solutions to the latent space;
6. Calculate the objective values of $X_{Tr}$ and $POS_s$ on problem $F_t(\cdot)$;
7. Sort on the combined populations $X_{Tr}$ and $POS_s$ through nondominated sort and crowding
   distance;
8. Select the top N solutions from the combined population as $\mathbf{P}_{init}$, where N is the population size;
9. Return $\mathbf{P}_{init}$.
```

2.3.1 Proposed Method to Generate An Initial Population

The main idea in a DMOEA is to generate an initial population such that the population can quickly reach the new optimum after a change. Therefore, in Tr-DMOEA, an initial population is produced through transferring good solutions from the previous environment. In this section, we present a method that makes use of the strategies proposed in sections 2.1.2 and 2.2.2 to improve Tr-DMOEA's solutions in the initial population after the changes. Firstly, in the process of transfer learning, the Gaussian kernel will be replaced with a linear one, as described in section 2.2.2. Secondly, considering that it is difficult to judge when changes are small and which problem has fixed POS for a DMOEA, transferred solutions and solutions copied from the previous environment are combined together, as described in section 2.1.2. The initial population is selected from the combined populations and then regarded as the initial population for the problem in next environment. The detailed procedures of improved Tr-DMOEA are shown in **Algorithm** 1. The differences with regarding to the original Tr-DMOEA are highlighted in the pseudocode. This algorithm can be embedded into any population based evolutionary algorithms.

2.3.2 Evaluation through Computational Studies

In order to evaluate the effectiveness of the proposed improved Tr-DMOEA, computational studies are conducted in this section. Firstly, we compare the performance of initial population produced by the original transfer learning and improved one in DMO following a similar procedure to that used in Sections 2.1.3 and 2.2.3. Then, the performance of populations obtained by several compared methods after optimisation is stated. The reason why we are now also comparing the results after optimisation is to verify whether the improved Tr-DMOEA helps to solve DMOPs, compared with other state-of-the-arts.

1) Solutions Quality Comparison in the First Generation:
Similar to previous experiments, comparison results of solutions after the change produced by the original and improved Tr-RMMEDA are shown in Table 5. It can be observed from the table that results of the improved Tr-RMMEDA are better than the original one on most of investigated problems. The Wilcoxon rank sum test result with $h = 1$ and $p = 0.0269$ shows that the transferred solutions through the improved Tr-DMOEA are significantly better than those through the original Tr-DMOEA in the first generation after change. In other words, transferred solutions through the improved Tr-DMOEA can adapt to changes more proactively and quickly soon after they happen more than those through the original Tr-DMOEA, which further proves the robustness of transferred solutions through the improved Tr-DMOEA.

2) Solutions Quality Comparison after Optimisation: Here, the experiment is conducted to verify the effect of the proposed method at the final generation after the changes. It should be noted that there are three investigated population-based algorithms, the fast and elitist multi-objective genetic algorithm (NSGA-II) [4], multi-objective particle swarm optimisation algorithm (MOPSO) [23] and regularity model-based multi-objective estimation of distribution algorithm (RMMEDA) [15],

which are regarded as the optimisation algorithms. The DMOEAs with the original transfer learning are termed as Tr-DMOEAs. Similarly, DMOEAs with proposed method are written as ImTr-DMOEAs.

For each ImTr-DMOEA, there are three compared DMOEAs: the original Tr-DMOEA, the static MOEA (COPY-DMOEA) and one with random response strategy (RND-DMOEA). The reason for using three different optimisation algorithms is to test the performance sensitivity of transfer learning-based method in three different type of algorithms. Parameters of different algorithms are set as the same in their original papers [8] [16] [23]. MIGD results of four compared DMOEAs with RMMEDA on all test problems are shown in Tables 6. Due to space limitations, the table showing MIGD results of four compared DMOEAs with MOPSO is omitted here and put in a supplementary material, which can be found in http://www.escience.cn/people/gruan/index.html. The results obtained by four DMOEAs with MOPSO are similar to those with RMMEDA. Additionally, in order to show the significant superiority of the proposed method to other algorithms, Friedman and Nemenyi statistical tests are conducted on all benchmark problems regarding MIGD of 12 algorithms (4 responding strategies with 3 EAs). The MIGD value obtained by a given algorithm on one problem with one parameter setting is regarded as an observation to compose that algorithm's group for the test, following Demsar's guidelines [14]. Therefore, there are 96 (12 problems and 8 parameters) observations in each group. Friedman detects significant differences in average accuracy with a p-value of 1.3726e-55. The Nemenyi post tests are shown in Figure 1, and are discussed over the next sub-sections. According to the test, average accuracy of the ImTr-RMMEDA is significantly better than that of the other approaches except Tr-RMMEDA and ImTr-MOPSO.

a) Impact of EAs on Different DMOEAs: Some DMOEAs with different EAs have different performance, while others have the similar performance, which can be seen from Figure 1. As a whole, MIGD values that RMMEDA obtains are better than those from MOPSO in most cases regarding all problems and different dynamics, while NSGA-II gets the worst MIGD results. For example, for problem FDA4 with the parameter setting C2, each DMOEA with RMMEDA has better MIGD than that of NSGA-II, no matter what the MIGD order of different DMOEAs with the same EA. This shows that RMMEDA are more suitable to solve these DMOPs than other two EAs. The above observations are confirmed by the Friedman and Nemenyi tests in Figure 1. When transferred solutions are not converged, solutions with Gaussian kernel-based transfer have better diversity, enabling better results to be achieved after the change than when using linear kernel. That is why the improved Tr-NSGA-II is worse than the original Tr-NSGA-II.

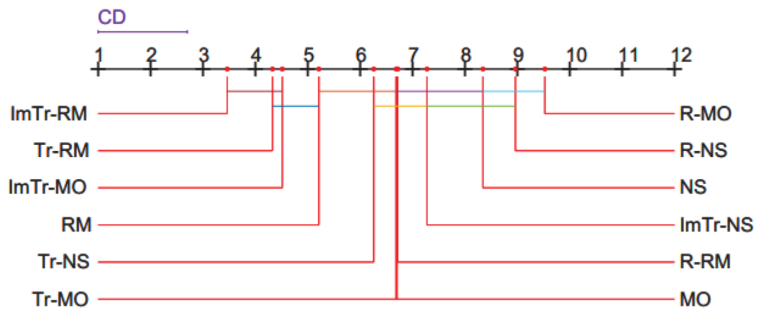
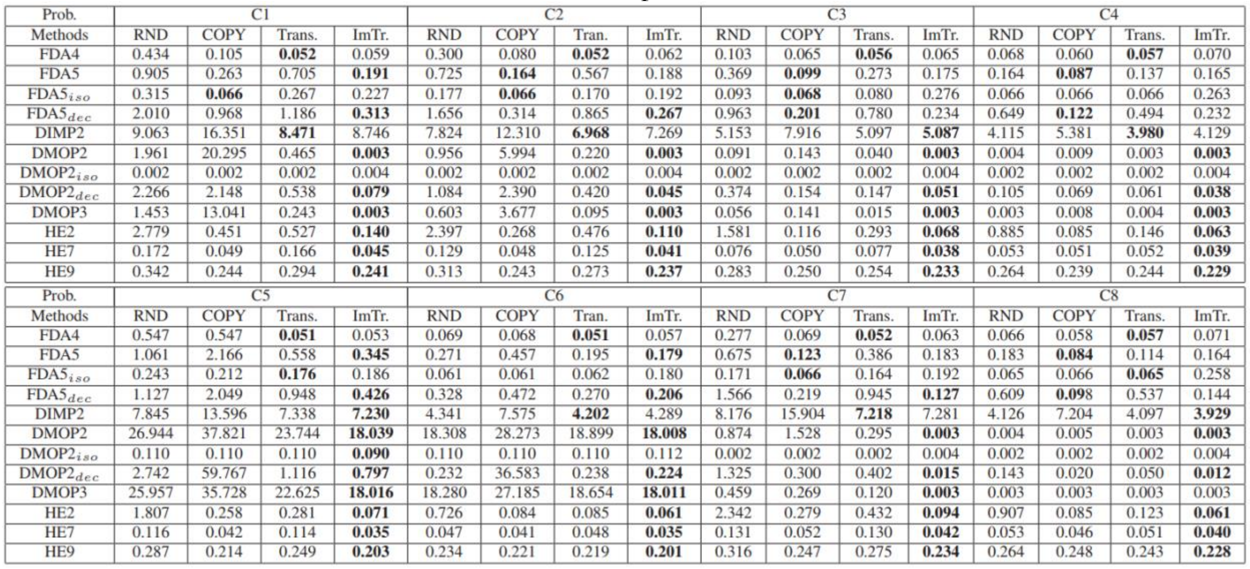


Figure 1: Friedman ranking among all DMOEAs from left to right, 4 responding strategies with respect to 3 EAs including RMMEDA, NSGA-II and MOPSO (referred as RM, NS and MO for simplicity). Any pair of approaches whose distance between them is larger than CD are considered to be different.

Table 6: MIGD values of four DMOEAs with the optimisation algorithm RMMEDA on all benchmark problems



Prob.	C1				C2				C3				C4			
Methods	RND	COPY	Trans.	ImTr.	RND	COPY	Tran.	ImTr.	RND	COPY	Trans.	ImTr.	RND	COPY	Trans.	ImTr.
FDA4	0.434	0.105	**0.052**	0.059	0.300	0.080	**0.052**	0.062	0.103	0.065	**0.056**	0.065	0.068	0.060	**0.057**	0.070
FDA5	0.905	0.263	0.705	**0.191**	0.725	**0.164**	0.567	0.188	0.369	**0.099**	0.273	0.175	0.164	**0.087**	0.137	0.165
FDA5$_{iso}$	0.315	**0.066**	0.267	0.227	0.177	**0.066**	0.170	0.192	0.093	**0.068**	0.080	0.276	0.066	0.066	0.066	0.263
FDA5$_{dec}$	2.010	0.968	1.186	**0.313**	1.656	0.314	0.865	**0.267**	0.963	**0.201**	0.780	0.234	0.649	**0.122**	0.494	0.232
DIMP2	9.063	16.351	**8.471**	8.746	7.824	12.310	**6.968**	7.269	5.153	7.916	5.097	**5.087**	4.115	5.381	**3.980**	4.129
DMOP2	1.961	20.295	0.465	**0.003**	0.956	5.994	0.220	**0.003**	0.091	0.143	0.040	**0.003**	0.004	0.009	0.003	**0.003**
DMOP2$_{iso}$	0.002	0.002	0.002	0.004	0.002	0.002	0.002	0.004	0.002	0.002	0.002	0.004	0.002	0.002	0.002	0.004
DMOP2$_{dec}$	2.266	2.148	0.538	**0.079**	1.084	2.390	0.420	**0.045**	0.374	0.154	0.147	**0.051**	0.105	0.069	0.061	**0.038**
DMOP3	1.453	13.041	0.243	**0.003**	0.603	3.677	0.095	**0.003**	0.056	0.141	0.015	**0.003**	0.003	0.008	0.004	**0.003**
HE2	2.779	0.451	0.527	**0.140**	2.397	0.268	0.476	**0.110**	1.581	0.116	0.293	**0.068**	0.885	0.085	0.146	**0.063**
HE7	0.172	0.049	0.166	**0.045**	0.129	0.048	0.125	**0.041**	0.076	0.050	0.077	**0.038**	0.053	0.051	0.052	**0.039**
HE9	0.342	0.244	0.294	**0.241**	0.313	0.243	0.273	**0.237**	0.283	0.250	0.254	**0.233**	0.264	0.239	0.244	**0.229**

Prob.	C5				C6				C7				C8			
Methods	RND	COPY	Trans.	ImTr.	RND	COPY	Tran.	ImTr.	RND	COPY	Trans.	ImTr.	RND	COPY	Trans.	ImTr.
FDA4	0.547	0.547	**0.051**	0.053	0.069	0.068	**0.051**	0.057	0.277	0.069	**0.052**	0.063	0.066	0.058	**0.057**	0.071
FDA5	1.061	2.166	0.558	**0.345**	0.271	0.457	0.195	**0.179**	0.675	**0.123**	0.386	0.183	0.183	**0.084**	0.114	0.164
FDA5$_{iso}$	0.243	0.212	**0.176**	0.186	0.061	0.061	0.062	0.180	0.171	**0.066**	0.164	0.192	0.065	0.066	**0.065**	0.258
FDA5$_{dec}$	1.127	2.049	0.948	**0.426**	0.328	0.472	0.270	**0.206**	1.566	0.219	0.945	**0.127**	0.609	**0.098**	0.537	0.144
DIMP2	7.845	13.596	7.338	**7.230**	4.341	7.575	**4.202**	4.289	8.176	15.904	**7.218**	7.281	4.126	7.204	4.097	**3.929**
DMOP2	26.944	37.821	23.744	**18.039**	18.308	28.273	18.899	**18.008**	0.874	1.528	0.295	**0.003**	0.004	0.005	0.003	**0.003**
DMOP2$_{iso}$	0.110	0.110	0.110	**0.090**	0.110	0.110	0.110	0.112	0.002	0.002	0.002	0.004	0.002	0.002	0.002	0.004
DMOP2$_{dec}$	2.742	59.767	1.116	**0.797**	0.232	36.583	0.238	**0.224**	1.325	0.300	0.402	**0.015**	0.143	0.020	0.050	**0.012**
DMOP3	25.957	35.728	22.625	**18.016**	18.280	27.185	18.654	**18.011**	0.459	0.269	0.120	**0.003**	0.003	0.003	0.003	0.003
HE2	1.807	0.258	0.281	**0.071**	0.726	0.084	0.085	**0.061**	2.342	0.279	0.432	**0.094**	0.907	0.085	0.123	**0.061**
HE7	0.116	0.042	0.114	**0.035**	0.047	0.041	0.048	**0.035**	0.131	0.052	0.130	**0.042**	0.053	0.046	0.051	**0.040**
HE9	0.287	0.214	0.249	**0.203**	0.234	0.221	0.219	**0.201**	0.316	0.247	0.275	**0.234**	0.264	0.248	0.243	**0.228**

For each problem with each parameter, all algorithms get optimised populations by RMMEDA in the last generation of each change. MIGD is the average of IGDs of these populations with one run under 20 changes. The best values that the algorithm obtains are highlighted in bold face.

b) Performance of DMOEAs on Different Benchmarks:

In general, it is clear from Table 6 that transfer learning based DMOEAs are all better than static MOEAs and DMOEAs with random initial population. This shows that transferred solutions can maintain the balance of convergence and diversity to some degree, compared with static MOEAs and random DMOEAs. When comparing the performance of Tr-DMOEAs and ImTr-DMOEAs, ImTr-DMOEAs are basically better than Tr-DMOEAs. Specifically, for HEs problems,
ImTr-DMOEAs are all better than Tr-DMOEAs except HE2 and HE7 when the EA is NSGA-II. This shows that transfer learning based DMOEA with the linear kernel and combined solutions is more capable of solving DMOPs with fixed POS. Regarding tri-objective optimisation problems, ImTrDMOEAs performs better than Tr-DMOEAs on FDA5iso and FDA5dec while worse than Tr-DMOEAs on FDA4 and FDA5. This implies that the improved Tr-DMOEAs can solve tri-objective problems with complex property such as isolated or deceptive POF. For DIMP2 whose decision variable has its own rate of change, it is difficult for any algorithm to solve, compared with other problems. The original Tr-DMOEA shows its superiority on this problem. In terms of other bi-objective problems, the Tr-DMOEAs with linear kernel function are all superior to those with the Gaussian kernel. This demonstrates that the improved Tr-DMOEA is better capable of solving tri-objective problems except the one with very complicated features like DIMP2, for which all algorithms struggle.

c) Influence of Dynamics on Tr-DMOEAs: Overall, it is clear from Table 6 that MIGD values obtained by all DMOEAs become better with the increase of changing frequency, when they are under same changing severity. The reason is obvious, which is that the more the iterations within each change, the better the performance. Additionally, no matter what kind of changing sizes are, two kinds of Tr-DMOEAs are all better than the random and static ones. The only difference between the original and the improved Tr-DMOEAs is their performance with different EAs, which has been introduced in section 2.3.2-2) a. Therefore, it can be concluded that the improved Tr-DMOEA can address most investigated problems no matter what kinds of dynamics are, as long as solutions before the change have been converged.

2.4 Summary

To summarise, this section studies transfer learning in DMO, analyzing when and how transfer learning works in DMO. It has been computationally observed that transfer learning works poorly on problems with fixed POS and when environmental changes are small. It has also been shown that the Gaussian kernel function in the existing method Tr-DMOEA [6] is inadequate for DMO. Based on these two observations, a new method has been proposed regarding avoiding transfer learning when it fails and replacing the Gaussian kernel with a linear one. Experimental results have shown that our proposed method is effective in solving DMOPs, compared with other state-of-the-art algorithms. In other words, the proposed method can adapt to changes more proactively and quickly soon after they happen, rather than spending a lot of time adapting to changes, which further proves the robustness of the proposed method.

3. Improve the Efficiency of Knowledge Transfer in Dynamic Multi-objective Optimisation

This section investigates how to improve the efficiency of knowledge transfer in dynamic multi-objective optimisation and whether the improved efficiency affects the performance of knowledge transfer. The rest of this section is organized as follows: Section 3.1 gives the time complexity of Tr-DMOEA and the computation time analysis of each component in Tr-DMOEA. Section 3.2 aims to investigate whether it is worthy to consume such long time taken by the transfer to use transfer learning in DMO. Section 3.3 studies whether the efficiency of transfer learning in DMO could be improved and whether the solution quality could be affected by the improved efficiency if the efficiency could be improved. Section 3.4 summarises the work of this section.

The following publication in the ECOLE is contributing to this section:

Ruan, G., Minku, L. L., Menzel, S., Sendhoff, B., & Yao, X. (2020, July). Computational Study on Effectiveness of Knowledge Transfer in Dynamic Multi-objective Optimisation. In 2020 IEEE Congress on Evolutionary Computation (CEC) (pp. 1-8). IEEE. (Best student paper finalist).

3.1 Efficiency of Transfer Learning in DMO

The time complexity of TCA and primal dual interior point has been analyzed in [8]. The major time cost of TCA is spent on the eigenvalue decomposition. It costs $O(d(m+n)^2)$ time when d nonzero eigenvectors are to be extracted, where m and n are the numbers of the solutions which are generated in the search space of source and target problems. For the primal dual interior point method, suppose the constraint matrix A has n rows and m columns, and $n < m$, it has $O(\sqrt{m}L)$ iterations and $O(m^3 L)$ arithmetic operations, where L is the total number of bits of the input. It is clear that the time complexity of the interior point is larger than that of TCA.

In order to verify the cost of TCA and interior point method from the perspective of computation time, an initial experiment is conducted regarding how much computation time these two parts consume in a single run. The experimental design is as follows: RMMEDA [15] is selected as the optimisation algorithm. Here, only one problem dMOP2 is used as the test problem as it is only used to reflect the proportion of computation time of each component in Tr-DMOEA. For the TCA parameters, the Gaussian kernel function is set as the default value and the expected dimensionality is set to be 20. The value of μ is set as 0.5. Computation time of each function of the whole algorithm will be recorded by the Profile environment of MATLAB 2018b. Computation time of each part of the Tr-RMMEDA is recorded and presented in Table 7. It is clear from the table that the interior point method consumes the most computation time, which confirms the complexity analysis of TCA and interior point in previous one section.

Table 7: Computation time of each process of Tr-RMMEDA with one run, one change and one parameter on problem DMOP2.

Process	Computation time	Proportion
Interior point	427.455 s	93.9%
RMMEDA	25.53 s	5.6 %
TCA	2.408 s	0.5%
Sum	455.393 s	100 %

3.2 Solutions Quality after Optimisation for Which Transfer Learning Cost Is Used

It has been experimentally shown that the existing Tr-DMOEA is extremely time-consuming. It is still unclear whether it is worthy to consume such long time to use transfer learning in DMO. This section is presented to explore this.

3.2.1 Computation Time of Transfer Learning Used for Optimisation

In order to figure out whether it is worthwhile to use transfer learning in DMO, this section designs an experiment to verify it. The main idea behind this experiment is to use the computation time of transfer learning to optimise randomly generated solutions. Whenever there is a change, transfer leaning is used to get the transferred initial population, while another initial population is randomly generated in the search space. The costs of transfer learning and random generation are recorded, termed as C_{tr} and C_{ran}. The cost is the running time determined based on the stopwatch timer in Matlab, where the Matlab command 'tic' and 'toc' starts and ends the timer, respectively. Then, the cost $C_{tr} - C_{ran}$ is used to optimise the randomly generated population. During the optimisation, both transferred population and random one will iterate for τ_t generations to get two optimised solutions.

3.2.2 Quality Comparison of Transferred And Random Solutions with Same Cost Budget

The specific experimental design has been introduced in the previous section. In this section we compare the quality of optimised solutions with transfer and optimised solutions from random generation, both of which are obtained under the same computation time budget.

a) Solution Quality Comparison in the First Generation: Here, we compare the quality of transferred solutions and those solutions which are optimised using the transfer learning computation time. MIGD is the average of these 20 IGD values under 20 changes for two solution sets. The mean and standard deviation of MIGD values are presented in Table 8, in which 'Transfer' and 'Random' refer to the MIGD values of these two solution sets. The better values that the method gets are highlighted in bold face. In order to indicate the significance between transferred solutions and optimised solutions using transfer learning computation time, the Wilcoxon rank sum test with the significance level 0.05 is carried out across problem instances, as recommended by Demsar [14]. MIGD value of 'Transfer' and 'Random' for each problem with one parameter is regarded as one observation data for the test. The result with h = 1 and p= 1.2624e-08 shows that random solutions optimised using transfer learning computation time are significantly better than transferred solutions. In other words, random solutions optimised using transfer learning computation time can adapt to changes more proactively and quickly soon after they happen more than transferred solutions, which further proves the robustness of random solutions optimised using transfer learning computation time.

Table 8: Mean and standard deviations of MIGD values of transferred solutions and those solutions optimised using the computation time of transfer learning after being randomly generated

Prob.	C1		C2		C3		C4	
Methods	Transfer	Random	Transfer	Random	Transfer	Random	Transfer	Random
FDA4	8.80e-02(2.50e-04)	**7.30e-02(2.45e-08)**	8.20e-02(2.05e-04)	**7.27e-02(2.01e-07)**	9.72e-02(3.95e-04)	**7.29e-02(5.38e-09)**	8.53e-02(2.38e-04)	**7.30e-02(4.06e-09)**
FDA5	2.62e-01(1.83e-03)	**1.77e-01(4.85e-06)**	2.03e-01(1.68e-03)	**1.73e-01(8.88e-07)**	1.94e-01(1.61e-03)	**1.75e-01(3.22e-06)**	1.89e-01(2.54e-03)	**1.74e-01(2.77e-06)**
FDA5$_{iso}$	**5.80e-02(6.97e-05)**	2.59e-01(1.57e-06)	**5.91e-02(3.24e-05)**	2.56e-01(1.67e-06)	**6.15e-02(5.93e-05)**	2.69e-01(2.35e-06)	**6.18e-02(5.33e-05)**	2.52e-01(1.85e-06)
FDA5$_{dec}$	5.41e-01(2.23e-02)	**2.06e-01(7.60e-06)**	5.79e-01(2.71e-02)	**1.98e-01(4.12e-06)**	5.20e-01(1.16e-02)	**2.00e-01(3.92e-07)**	5.13e-01(9.76e-03)	**1.90e-01(1.41e-06)**
DIMP2	9.61e+00(2.37e+00)	**3.57e+00(9.26e-04)**	9.97e+00(1.07e+00)	**3.53e+00(6.77e-05)**	9.03e+00(3.57e+00)	**3.54e+00(5.98e-04)**	9.77e+00(1.32e+00)	**3.56e+00(6.11e-04)**
DMOP2	4.05e-01(3.69e-02)	**1.46e-02(1.59e-04)**	3.02e-01(1.78e-02)	**1.81e-02(2.69e-04)**	2.34e-01(5.82e-03)	**5.12e-03(4.52e-06)**	2.29e-01(5.17e-03)	**1.15e-02(8.32e-05)**
DMOP2$_{iso}$	**2.19e-03(7.48e-09)**	4.45e-03(3.35e-10)	**2.32e-03(7.88e-08)**	4.44e-03(6.10e-10)	**2.49e-03(1.48e-07)**	4.44e-03(1.48e-10)	**2.55e-03(1.35e-07)**	4.44e-03(2.34e-10)
DMOP2$_{dec}$	6.45e-01(1.24e-01)	**2.54e-02(1.87e-05)**	5.12e-01(4.01e-02)	**2.28e-02(2.81e-05)**	3.48e-01(5.99e-03)	**2.35e-02(2.17e-05)**	2.94e-01(5.18e-03)	**1.48e-02(4.17e-06)**
DMOP3	3.14e-01(5.55e-02)	**3.28e-03(5.27e-11)**	1.38e-01(3.28e-03)	**3.28e-03(5.63e-11)**	1.39e-01(1.32e-02)	**3.29e-03(1.43e-11)**	7.20e-02(4.84e-03)	**3.30e-03(8.28e-11)**
HE2	4.59e-01(2.49e-02)	**5.98e-02(6.63e-08)**	4.97e-01(1.38e-02)	**5.89e-02(2.62e-08)**	3.23e-01(1.07e-03)	**5.77e-02(5.15e-08)**	2.70e-01(5.19e-03)	**5.66e-02(8.82e-10)**
HE7	2.24e-01(1.94e-04)	**3.70e-02(1.41e-08)**	2.29e-01(2.97e-04)	**3.72e-02(3.44e-09)**	2.48e-01(3.49e-04)	**3.70e-02(1.38e-08)**	2.64e-01(3.56e-04)	**3.71e-02(7.82e-10)**
HE9	3.30e-01(2.51e-04)	**2.34e-01(7.89e-08)**	3.06e-01(4.54e-04)	**2.34e-01(1.43e-07)**	2.83e-01(2.29e-04)	**2.33e-01(1.82e-07)**	2.80e-01(4.85e-04)	**2.32e-01(7.33e-08)**

Prob.	C5		C6		C7		C8	
Methods	Transfer	Random	Transfer	Random	Transfer	Random	Transfer	Random
FDA4	7.23e-02(1.18e-04)	**5.99e-02(4.24e-09)**	6.86e-02(1.20e-04)	**6.01e-02(5.67e-09)**	**7.01e-02(1.55e-04)**	7.32e-02(5.29e-09)	7.35e-02(3.09e-04)	**7.33e-02(5.46e-09)**
FDA5	1.35e+00(4.76e-01)	**1.43e-01(4.38e-07)**	1.86e+00(2.89e-01)	**1.44e-01(1.24e-06)**	1.24e+00(5.75e-01)	**1.70e-01(5.22e-07)**	6.70e-01(9.62e-03)	**1.74e-01(7.32e-06)**
FDA5$_{iso}$	5.33e-01(1.37e-02)	**1.78e-01(1.47e-05)**	5.44e-01(6.02e-02)	**1.85e-01(6.78e-06)**	6.49e-01(2.01e-02)	**2.51e-01(9.83e-06)**	5.51e-01(4.31e-03)	**2.49e-01(9.90e-06)**
FDA5$_{dec}$	2.28e+00(2.43e-01)	**1.59e-01(9.46e-06)**	2.33e+00(2.77e-01)	**1.59e-01(6.53e-06)**	2.14e+00(3.27e-01)	**1.87e-01(1.27e-06)**	1.21e+00(2.80e-01)	**1.90e-01(7.73e-06)**
DIMP2	1.09e+01(9.25e-01)	**3.84e+00(1.07e-04)**	9.81e+00(9.88e-01)	**3.83e+00(1.06e-04)**	1.30e+01(1.59e+00)	**3.78e+00(3.37e-04)**	1.27e+01(1.63e+00)	**3.78e+00(6.19e-04)**
DMOP2	1.10e+02(7.79e+01)	**1.81e+01(4.03e-04)**	1.13e+02(1.15e+02)	**1.80e+01(8.19e-05)**	2.70e+00(2.18e+00)	**5.15e-03(4.60e-06)**	3.64e+00(1.41e+00)	**5.26e-03(4.97e-06)**
DMOP2$_{iso}$	4.36e-01(5.21e-08)	**9.08e-02(4.66e-10)**	4.36e-01(9.49e-08)	**1.12e-01(8.16e-11)**	**2.67e-03(2.45e-07)**	4.44e-03(1.10e-10)	**2.89e-03(5.71e-07)**	4.45e-03(9.63e-11)
DMOP2$_{dec}$	5.38e+00(3.82e+01)	**1.74e-01(7.42e-10)**	3.94e+00(5.68e+00)	**1.94e-01(1.38e-10)**	2.04e+00(8.07e-01)	**2.33e-02(7.25e-05)**	2.69e+00(2.12e+00)	**1.71e-02(1.12e-05)**
DMOP3	1.12e+02(4.89e+01)	**1.80e+01(2.02e-08)**	1.13e+02(2.98e+02)	**1.80e+01(4.12e-09)**	1.44e+00(8.44e-01)	**3.33e-03(2.49e-11)**	2.22e+00(9.56e-01)	**3.30e-03(6.02e-11)**
HE2	1.25e-01(7.15e-04)	**5.64e-02(1.19e-08)**	1.21e-01(5.20e-04)	**5.49e-02(4.11e-10)**	6.06e-01(5.69e-03)	**5.84e-02(3.57e-08)**	5.02e-01(2.14e-03)	**5.66e-02(6.04e-10)**
HE7	1.72e-01(1.14e-03)	**3.42e-02(1.17e-08)**	1.82e-01(1.94e-04)	**3.42e-02(2.40e-08)**	2.53e-01(4.60e-04)	**3.70e-02(4.02e-09)**	2.73e-01(5.03e-04)	**3.70e-02(1.33e-08)**
HE9	2.76e-01(1.01e-04)	**2.09e-01(2.78e-07)**	2.58e-01(6.38e-04)	**2.08e-01(1.83e-07)**	3.19e-01(7.59e-05)	**2.34e-01(2.96e-08)**	2.79e-01(2.94e-04)	**2.31e-01(1.58e-07)**

b) Solution Quality Comparison after Optimisation: Here, we compare the quality of optimised solutions with transfer learning and those solutions which originate from random generation. The mean and standard deviation of MIGD values are presented in Table 9, in which ‘Transfer’ and ‘Random’ refer to the MIGD values of these two solution sets. The better values that the method gets are highlighted in bold face.

Table 9: Mean and standard deviations of MIGD values of optimised solutions with transfer learning and random solutions that first iterate for some generation which consumes the same computation time as transfer learning, and then are optimised the same generation as those with transfer

Prob.	C1		C2		C3		C4	
Methods	Transfer	Random	Transfer	Random	Transfer	Random	Transfer	Random
FDA4	**5.85e-02(6.23e-05)**	7.30e-02(2.52e-08)	**5.73e-02(4.51e-06)**	7.26e-02(3.46e-08)	**6.38e-02(6.46e-06)**	7.28e-02(5.53e-09)	**6.56e-02(1.88e-06)**	7.32e-02(4.77e-09)
FDA5	**1.17e-01(3.18e-04)**	1.82e-01(2.80e-06)	**7.23e-02(2.49e-05)**	1.74e-01(1.04e-06)	**7.33e-02(2.75e-06)**	1.82e-01(3.32e-07)	**7.77e-02(3.69e-06)**	1.76e-01(2.84e-06)
FDA5$_{iso}$	**6.12e-02(8.74e-06)**	2.61e-01(2.33e-06)	**7.45e-02(9.93e-06)**	2.57e-01(4.63e-06)	**8.48e-02(4.85e-06)**	2.77e-01(1.60e-05)	**8.76e-02(3.31e-06)**	2.62e-01(2.35e-06)
FDA5$_{dec}$	4.86e-01(9.24e-03)	**2.05e-01(4.34e-06)**	4.00e-01(8.08e-03)	**1.94e-01(3.20e-06)**	3.05e-01(5.72e-03)	**2.00e-01(1.61e-06)**	**1.91e-01(1.96e-03)**	1.93e-01(1.43e-06)
DIMP2	7.02e+00(8.61e-01)	**3.57e+00(9.26e-04)**	5.43e+00(4.27e-01)	**3.53e+00(6.75e-05)**	3.64e+00(1.26e-01)	**3.54e+00(5.96e-04)**	**3.18e+00(6.77e-03)**	3.56e+00(6.10e-04)
DMOP2	2.77e-01(2.33e-02)	**1.46e-02(1.57e-04)**	1.17e-01(1.35e-03)	**1.80e-02(2.68e-04)**	8.78e-03(4.43e-05)	**5.12e-03(4.50e-06)**	**4.17e-03(5.86e-06)**	1.15e-02(8.40e-05)
DMOP2$_{iso}$	**2.58e-03(5.02e-08)**	4.45e-03(1.12e-10)	**3.12e-03(1.45e-07)**	4.45e-03(1.06e-10)	**3.73e-03(6.53e-08)**	4.47e-03(1.32e-10)	**4.06e-03(2.81e-08)**	4.49e-03(1.42e-10)
DMOP2$_{dec}$	4.77e-01(4.35e-02)	**2.41e-02(1.87e-05)**	2.87e-01(2.41e-02)	**2.27e-02(2.69e-05)**	6.02e-02(4.78e-04)	**2.29e-02(2.37e-05)**	**1.27e-02(1.87e-04)**	1.48e-02(4.15e-06)
DMOP3	2.19e-01(4.21e-02)	**3.29e-03(3.28e-11)**	4.90e-02(3.69e-04)	**3.30e-03(3.16e-11)**	4.21e-03(5.04e-07)	**3.29e-03(3.09e-11)**	3.51e-03(5.93e-09)	**3.30e-03(1.43e-11)**
HE2	4.22e-01(1.83e-02)	**5.97e-02(7.59e-08)**	3.93e-01(8.25e-03)	**5.88e-02(2.45e-08)**	2.19e-01(2.25e-03)	**5.77e-02(4.34e-08)**	1.04e-01(6.91e-04)	**5.66e-02(4.21e-10)**
HE7	1.62e-01(9.35e-05)	**3.71e-02(1.93e-08)**	1.23e-01(8.81e-05)	**3.72e-02(4.49e-09)**	7.07e-02(5.22e-05)	**3.70e-02(3.81e-09)**	5.01e-02(9.82e-06)	**3.70e-02(9.77e-09)**
HE9	3.02e-01(1.43e-04)	**2.34e-01(7.38e-08)**	2.72e-01(1.82e-04)	**2.34e-01(1.30e-07)**	2.43e-01(4.81e-05)	**2.33e-01(1.45e-07)**	2.37e-01(2.59e-05)	**2.33e-01(1.64e-07)**

Prob.	C5		C6		C7		C8	
Methods	Transfer	Random	Transfer	Random	Transfer	Random	Transfer	Random
FDA4	**5.06e-02(2.99e-07)**	6.01e-02(1.04e-08)	**5.63e-02(2.88e-06)**	5.99e-02(1.14e-08)	**5.08e-02(1.33e-06)**	7.31e-02(7.13e-09)	**5.62e-02(1.89e-06)**	7.28e-02(2.19e-08)
FDA5	7.45e-01(5.23e-02)	**1.43e-01(3.21e-06)**	3.35e-01(3.64e-03)	**1.39e-01(2.60e-06)**	8.96e-01(2.55e-01)	**1.71e-01(1.07e-06)**	2.84e-01(2.00e-03)	**1.72e-01(6.59e-06)**
FDA5$_{iso}$	3.88e-01(8.47e-03)	**1.80e-01(1.55e-05)**	2.24e-01(1.08e-03)	**1.91e-01(4.60e-06)**	4.34e-01(6.13e-03)	**2.51e-01(1.01e-05)**	**2.48e-01(1.82e-03)**	2.61e-01(8.09e-06)
FDA5$_{dec}$	1.57e+00(4.95e-01)	**1.61e-01(2.65e-06)**	4.35e-01(9.41e-03)	**1.57e-01(3.00e-06)**	1.39e+00(3.41e-01)	**1.89e-01(5.32e-06)**	4.09e-01(9.33e-03)	**1.89e-01(1.93e-06)**
DIMP2	7.49e+00(7.21e-01)	**3.84e+00(1.03e-04)**	4.02e+00(3.28e-01)	**3.83e+00(1.06e-04)**	9.18e+00(1.04e+00)	**3.78e+00(3.36e-04)**	5.89e+00(1.17e-01)	**3.78e+00(6.20e-04)**
DMOP2	9.16e+01(4.48e+01)	**1.81e+01(4.05e-04)**	7.42e+01(1.45e+01)	**1.80e+01(8.33e-05)**	7.34e-01(1.39e-01)	**5.13e-03(4.52e-06)**	4.46e-02(1.11e-02)	**5.23e-03(4.74e-06)**
DMOP2$_{iso}$	4.36e-01(1.11e-08)	**9.08e-02(1.79e-10)**	4.36e-01(1.96e-09)	**1.12e-01(7.45e-11)**	**3.39e-03(1.99e-07)**	4.46e-03(4.45e-10)	**4.35e-03(7.25e-08)**	4.50e-03(2.86e-10)
DMOP2$_{dec}$	7.66e-01(1.60e+00)	**1.74e-01(4.99e-10)**	4.32e-01(8.19e-08)	**1.94e-01(1.74e-10)**	5.40e-01(3.17e-02)	**2.30e-02(6.80e-05)**	**8.76e-03(3.02e-05)**	1.68e-02(9.73e-06)
DMOP3	9.05e+01(1.17e+01)	**1.80e+01(1.38e-08)**	7.33e+01(1.01e+00)	**1.80e+01(1.28e-09)**	3.78e-01(7.07e-02)	**3.33e-03(1.51e-11)**	5.44e-03(3.04e-05)	**3.31e-03(2.85e-11)**
HE2	6.95e-02(1.68e-04)	**5.64e-02(1.11e-08)**	**5.36e-02(2.14e-06)**	5.49e-02(4.23e-09)	**5.38e-01(5.53e-03)**	5.84e-02(3.39e-08)	1.57e-01(1.19e-03)	**5.66e-02(5.72e-10)**
HE7	8.26e-02(1.95e-04)	**3.43e-02(7.46e-09)**	3.73e-02(1.17e-06)	**3.42e-02(2.07e-08)**	1.43e-01(2.91e-04)	**3.69e-02(3.33e-09)**	5.21e-02(8.34e-06)	**3.70e-02(2.96e-09)**
HE9	2.21e-01(1.54e-04)	**2.09e-01(2.57e-07)**	**1.80e-01(4.12e-05)**	2.08e-01(1.71e-07)	2.86e-01(2.78e-05)	**2.34e-01(3.44e-08)**	2.54e-01(9.71e-05)	**2.31e-01(1.70e-07)**

In order to indicate the significance between optimised solutions with transfer learning and those from random generation, the Wilcoxon rank sum test with the significance level 0.05 is carried out. MIGD value of 'Transfer' and 'Random' for each problem with one parameter is regarded as one observation data for the test. The result with h = 1 and p = 0.0085 shows that solutions which originates from random generation are significantly better than optimised solutions with transfer learning in the last generation of optimisation. It can be concluded that under the same computation time budget, random solution after optimisation will obtain better results than those from transfer learning, no matter whether the optimisation process is conducted on transferred solutions or not. In the original paper, Tr-DMOEA is better than the algorithm with random solutions. The difference is that in the original paper, the same evaluation times and generations are given to these two algorithms, while the same computation time budget is given in this paper. This shows that it is not worth to consume such long time on transfer learning, while achieving worse results than consuming these computation time on optimising. In other words, it vanishes the original purpose of using transfer learning in DMO, reducing the efforts of optimisation. On the other side, when comparing Tables 8 and 9, it is clear that the improvement of transferred solutions are more than that of optimised solution from randomization after τ_t generations' optimisation. In addition, as τ_t increases, the quality of transferred solutions becomes better while those from randomization remain unchanged. These two observations show that transferred solutions are far from the optimum of problems, while solutions from randomization have already been nearly Pareto optimal.

3.3 Computational Analysis of Three Tr-DMOEA Variants

It has been experimentally shown that the existing Tr-DMOEA is extremely time-consuming and randomly generated solutions after being optimised using transfer learning time are greatly better than transferred ones. It is unclear whether the efficiency could be improved and whether the solution quality could be affected by the improved efficiency if the efficiency could be improved.

3.3.1 Computation Time Comparison of Three Tr-DMOEA Variants

Given that the computation time of solving the inner optimisation problem in Tr-DMOEA is very large, it is unclear whether the efficiency of solving the inner problem (equation (3)) could be enhanced. To explore this, other two popular optimisation methods are used here, which are the active set [24] and sequential quadratic programming (sqp) [24] methods. We have conducted a set of experiments to validate the efficiency and effectiveness of these three Tr-DMOEA variants.

Here, we give a brief description of active set and sqp method:
1). The active set method [24] considers an optimisation problem with n constraints $g_1(x) \geq 0, \ldots, g_n(x) \geq 0$. For a point x in the feasible region, a constraint is called active at x if $g_i(x) = 0$ and inactive if $g_i(x) > 0$. The procedures for the active set are as follows: solve the equality problem defined by the active set (approximately) with a solution x_*; compute the Lagrange multipliers of the active set; remove a subset of the constraints with negative Lagrange

multipliers; search for the new boundary based on x_* along the feasible region boundary formed by the active set and add the constraint related to the new boundary to the approximated active set. Iterate these procedures until the optima is found.
2). Sqp [24] is an iterative method for constrained nonlinear optimisation. At each iteration, a basic sequential quadratic programming algorithm defines an appropriate search direction as a solution to the quadratic programming subproblem. Note that all three optimisation methods (interior point, sqp and active set) used in this paper are from the optimisation toolbox of MATLAB 2018b. The specific implementation of these three methods and the difference between the sqp and the active-set algorithms can be found in MATLAB's online documents.

The computation time of three different optimisation methods in these three Tr-DMOEA variants is only recorded when solving the inner problem (equation (3)), for all test problems with all parameter setting. The specific computation time comparisons of these three optimisation methods in DMO, the comparison results of computation time of three Tr-DMOEA variants are shown in Table 10. Due to space limitations, only results of C5 and C6 are presented. Results of other parameter settings are similar to those of C5 and C6. It is clear from Table 10 that for all test problems with those parameter settings, the interior point method consumes the most time among the three compared methods to solve the inner optimisation problem, while the active set method is the most efficient optimisation algorithm. In addition, the time interior points method consumes is several times over that another two methods consume. This shows that another two inner optimisation methods can greatly improve the efficiency of transfer learning in DMO.

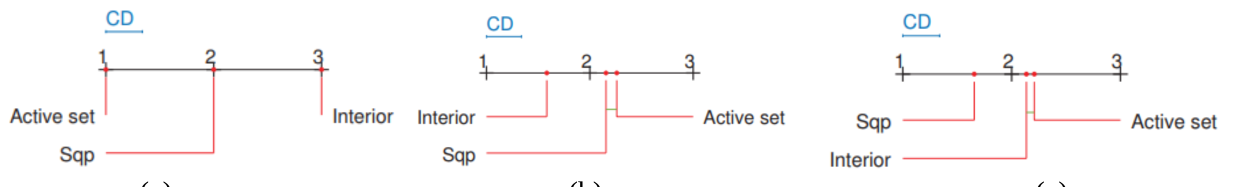
Table 10: Mean and standard deviations of computation time (in seconds) of three optimisation methods for minimizing the distance of source and target problems in the latent space with 20 environment changes for all problems with 2 parameter settings under 20 independent runs.

Prob.	C5			C6		
Methods	Interior	sqp	active	Interior	sqp	active
FDA4	6.38e+03(8.93e+04)	8.77e+02(1.28e+03)	**4.59e+02(7.89e+02)**	6.42e+03(1.34e+05)	8.31e+02(7.18e+02)	**4.25e+02(7.10e+02)**
FDA5	3.48e+03(1.46e+05)	3.64e+02(5.91e+02)	**2.00e+02(6.25e+02)**	5.67e+03(2.38e+05)	7.62e+02(8.12e+02)	**3.70e+02(1.58e+02)**
FDA5_{iso}	1.36e+03(8.46e+03)	4.18e+02(4.03e+02)	**2.69e+02(1.64e+02)**	1.31e+03(1.35e+04)	4.18e+02(2.77e+02)	**2.69e+02(7.39e+01)**
FDA5_{dec}	3.03e+03(1.56e+04)	3.46e+02(8.70e+02)	**1.44e+02(2.95e+02)**	4.74e+03(5.48e+04)	7.01e+02(4.91e+03)	**3.11e+02(2.48e+02)**
DIMP2	9.62e+02(2.11e+04)	2.80e+01(2.25e+01)	**2.11e+01(1.29e+01)**	6.74e+03(1.78e+06)	1.49e+02(4.52e+02)	**7.74e+01(2.76e+02)**
DMOP2	7.26e+02(8.73e+03)	3.15e+02(1.35e+03)	**1.03e+02(1.43e+02)**	5.35e+03(2.68e+05)	3.69e+02(1.11e+03)	**2.05e+02(2.84e+01)**
DMOP2_{iso}	1.34e+03(5.58e+03)	5.97e+02(2.33e+04)	**4.44e+02(1.25e+03)**	1.44e+03(5.92e+03)	5.78e+02(2.20e+04)	**4.33e+02(1.47e+03)**
DMOP2_{dec}	5.81e+02(3.71e+03)	5.07e+01(5.06e+01)	**1.86e+01(3.25e+00)**	3.84e+03(6.01e+04)	3.47e+02(4.55e+02)	**1.97e+02(1.84e+01)**
DMOP3	8.64e+02(2.20e+04)	2.98e+02(1.46e+03)	**1.13e+02(2.51e+02)**	5.45e+03(4.75e+05)	3.88e+02(4.98e+02)	**1.99e+02(1.07e+02)**
HE2	2.11e+03(5.02e+04)	4.35e+02(5.15e+03)	**3.28e+02(3.65e+03)**	1.35e+04(1.52e+06)	2.29e+03(7.51e+04)	**1.55e+03(5.45e+04)**
HE7	5.84e+03(3.36e+05)	1.28e+03(5.17e+03)	**1.15e+03(6.44e+03)**	1.07e+04(1.94e+06)	2.81e+03(1.29e+04)	**2.47e+03(4.82e+04)**
HE9	2.91e+03(1.76e+05)	8.51e+02(5.40e+02)	**6.68e+02(1.71e+03)**	6.47e+03(7.73e+05)	1.79e+03(6.05e+02)	**1.50e+03(1.06e+04)**

Friedman and Nemenyi statistical tests [14] were carried out across benchmark problems. The computation time that all algorithms get on one problem with one parameter setting is regarded as an observation of the test. Therefore, there are 96 (12 problems and 8 parameters) observed data. Friedman detects significant differences in median accuracy with a p value of 2.0311e-42. The Nemenyi post-tests are shown in Figure 2(a). This shows that the sqp and active set method are significantly more efficient than the interior point method.

3.3.2 Solutions Quality Comparison of Three Tr-DMOEA Variants

In order to evaluate the influence of the improved efficiency on the solution quality, in this section, transferred solutions of these Tr-DMOEAs in the first generation after change are first compared. The optimised solution of these Tr-DMOEAs in the last generation after change are then compared.



(a) (b). (c).

Figure 2: Friedman ranking among **(a).** computation time of three optimisation methods (active set, sqp and interior point) from left to right with Friedman test's p-value 2.0311e-42; **(b)**. MIGD values of three transferred solution sets obtained by Tr-DMOEAs with different inner optimisation methods (interior point, sqp and active set) in the first generation after change from left to right with Friedman test's p-value 1.7723e-6; **(c)**. MIGD values of three optimised solution sets obtained by Tr-DMOEAs with different inner optimisation methods (sqp, interior point and active set) in the last generation after optimisation from left to right with Friedman test's p-value 1.1162e-4. Any pair of approaches whose distance between them is larger than CD are considered to be significantly different based on the Nemeyi post-hoc test.

A. Solutions Quality Comparison in the First Generation

Inverted Generational Distance (IGD) [12] is used to compare the quality of solution sets obtained by these three Tr-DMOEAs. IGD can measure the diversity and convergence of a solution set found by an algorithm, so it can give us a comprehensive understanding about the performance of the compared algorithms. MIGD [13] is a modified version of IGD, which is the average IGD values in all changes. The smaller the MIGD the better the algorithm. For each Tr-DMOEA on one problem with one parameter setting, in the first generation after each change, a solution set obtained by transfer learning in the experiment of section 3.3.1 is used to calculate the IGD value. MIGD is the average of these 20 IGD values under 20 changes. Therefore, these three Tr-DMOEA variants will obtain one MIGD value on each problem for each parameter. Each algorithm is independently run 20 times on each problem instance with each parameter setting. The mean and standard deviation of MIGD values of transferred solutions obtained by the three Tr-DMOEAs in the first generation after changes are shown in Table 11. In order to show the significant superiority of one method to others, Friedman and Nemenyi statistical tests were carried out across benchmark problems following Demsar's recommendation [14]. The MIGD values that all algorithms get on one problem with one parameter setting are regarded as an observation of the test. Therefore, there are 96 (12 problems and 8 parameters) observed data. Friedman detects significant differences in average accuracy with a p-value of 1.7723e-6. The Nemenyi post-tests are shown in Figure 2(b). Only results of C5 and C6 are presented due to space limitation. It has been observed that in most cases the TrDMOEA with the interior point method achieves the best results, compared with the two other Tr-DMOEA variants. The Friedman and Nemenyi statistical tests show that Tr-DMOEA with interior point significantly outperforms the other two variants.

Table 11: Mean and standard deviations of MIGD values of transferred **in the first generation after changes** optimised by three Tr-RMMEDA variants with different inner optimisation algorithms

Prob.	C5			C6		
Methods	Interior	sqp	active	Interior	sqp	active
FDA4	**9.85e-02(3.99e-06)**	1.00e-01(6.48e-06)	1.03e-01(1.32e-05)	**1.06e-01(7.04e-06)**	1.12e-01(1.35e-05)	1.18e-01(2.15e-05)
FDA5	1.28e+00(7.92e-03)	6.23e-01(1.65e-03)	**5.72e-01(1.46e-03)**	1.22e+00(8.41e-03)	5.15e-01(7.53e-04)	**5.07e-01(3.91e-04)**
$FDA5_{iso}$	**5.39e-01(2.67e-03)**	5.42e-01(6.59e-03)	6.79e-01(4.93e-03)	**5.07e-01(1.15e-03)**	5.31e-01(2.75e-03)	6.42e-01(5.70e-03)
$FDA5_{dec}$	2.11e+00(4.11e-02)	1.13e+00(1.44e-02)	**9.91e-01(1.92e-02)**	2.69e+00(4.38e-02)	6.93e-01(7.07e-04)	**6.36e-01(2.28e-03)**
DIMP2	**1.53e+01(2.25e-01)**	1.79e+01(4.41e-01)	1.69e+01(2.74e-01)	**1.21e+01(1.53e-01)**	1.44e+01(9.16e-02)	1.44e+01(2.01e-01)
DMOP2	3.92e+01(3.28e-01)	**1.82e+01(2.64e-04)**	1.82e+01(4.64e-04)	3.52e+01(1.01e+00)	1.81e+01(2.52e-04)	**1.81e+01(1.85e-04)**
$DMOP2_{iso}$	**8.50e-02(1.29e-05)**	8.85e-02(8.31e-06)	8.68e-02(6.61e-06)	**1.07e-01(2.36e-05)**	1.10e-01(7.90e-06)	1.10e-01(1.18e-05)
$DMOP2_{dec}$	**8.86e+00(2.38e+00)**	1.57e+05(4.70e+11)	8.08e+05(1.95e+12)	6.72e+00(4.26e-01)	**2.76e+00(4.07e+00)**	2.85e+00(4.32e+00)
DMOP3	3.70e+01(3.75e-01)	**1.82e+01(4.23e-04)**	1.82e+01(5.34e-04)	3.39e+01(8.30e-01)	1.82e+01(5.21e-04)	**1.82e+01(5.19e-04)**
HE2	7.20e-01(4.59e-02)	5.89e-01(1.69e-02)	**4.72e-01(2.68e-02)**	2.36e-01(9.37e-05)	2.81e-01(2.57e-04)	**2.25e-01(3.64e-04)**
HE7	**3.05e-01(3.48e-05)**	3.46e-01(6.97e-05)	3.48e-01(6.82e-05)	**2.85e-01(2.29e-05)**	3.18e-01(2.12e-05)	3.17e-01(3.26e-05)
HE9	3.12e-01(5.27e-05)	**3.07e-01(1.73e-05)**	3.19e-01(8.23e-05)	**2.69e-01(3.58e-05)**	2.84e-01(1.08e-05)	2.85e-01(1.55e-05)

B. Solutions Quality Comparison after Optimisation

Similarly, in the experiment of section 3.3.1, each TrDMOEA will get a solution set on one problem with one parameter setting, after optimisation for τ_t generations. Each algorithm is independently run 20 times on each problem instance with each parameter setting. The obtained solution set is used to calculate the IGD value. Also, the IGD values of 20 changes are averaged to get the MIGD for each problem with one parameter. The mean and standard deviation of MIGD values of transferred solutions obtained by three Tr-DMOEAs in the last generation after change are shown in Table 12. Friedman and Nemenyi statistical tests [14] were carried out across benchmark problems. The MIGD values that all algorithms get on one problem with one parameter setting are regarded as an observation of the test. Therefore, there are 96 (12 problems and 8 parameters) observed data. Friedman detects significant differences in average accuracy with a p value of 1.1162e-4. The Nemenyi post-tests are shown in Figure 2(c).

Table 12: Mean and standard deviations of MIGD values of transferred in the last generation after optimisation optimised by three Tr-RMMEDA variants with different inner optimisation algorithms

Prob.	C5			C6		
Methods	Interior	sqp	active	Interior	sqp	active
FDA4	5.06e-02(8.71e-09)	**4.87e-02(1.80e-08)**	4.97e-02(2.80e-08)	5.65e-02(8.32e-09)	**5.53e-02(1.22e-08)**	5.60e-02(9.64e-09)
FDA5	5.33e-01(1.27e-04)	3.84e-01(8.14e-06)	**3.79e-01(2.42e-05)**	**1.91e-01(3.94e-06)**	2.08e-01(1.19e-05)	2.31e-01(8.89e-06)
$FDA5_{iso}$	2.22e-01(4.64e-06)	**1.83e-01(1.02e-05)**	2.05e-01(2.02e-05)	1.61e-01(1.42e-06)	**1.51e-01(5.54e-06)**	1.55e-01(3.01e-06)
$FDA5_{dec}$	8.39e-01(1.48e-03)	**6.48e-01(5.58e-05)**	6.53e-01(3.82e-04)	**2.79e-01(7.93e-06)**	3.51e-01(2.10e-05)	3.53e-01(6.43e-06)
DIMP2	**7.28e+00(7.10e-04)**	7.93e+00(4.81e-03)	8.34e+00(4.94e-03)	4.21e+00(2.69e-04)	**4.11e+00(3.85e-03)**	4.29e+00(1.44e-04)
DMOP2	2.36e+01(1.07e-02)	**1.80e+01(1.24e-07)**	1.81e+01(2.31e-06)	1.85e+01(1.15e-02)	1.80e+01(7.61e-09)	**1.80e+01(1.48e-10)**
$DMOP2_{iso}$	**8.98e-02(1.90e-10)**	9.04e-02(1.94e-10)	9.05e-02(9.52e-10)	**1.12e-01(8.90e-11)**	1.12e-01(3.06e-10)	1.12e-01(1.22e-10)
$DMOP2_{dec}$	1.20e+00(4.20e-03)	**3.50e-01(8.52e-05)**	4.16e-01(1.15e-03)	2.37e-01(4.19e-05)	**1.11e-01(3.23e-08)**	1.18e-01(3.16e-05)
DMOP3	2.32e+01(1.65e-02)	**1.80e+01(2.98e-07)**	1.80e+01(1.93e-06)	1.86e+01(3.18e-03)	**1.80e+01(1.53e-11)**	1.80e+01(1.36e-08)
HE2	3.27e-01(1.26e-04)	2.28e-01(2.91e-04)	**1.88e-01(2.47e-04)**	7.80e-02(3.22e-07)	7.12e-02(2.55e-06)	**6.78e-02(2.48e-06)**
HE7	1.12e-01(4.66e-07)	1.11e-01(4.26e-07)	**1.09e-01(2.36e-07)**	4.82e-02(4.88e-08)	**4.65e-02(3.88e-08)**	4.66e-02(2.40e-08)
HE9	2.48e-01(3.91e-07)	**2.39e-01(2.06e-07)**	2.44e-01(3.44e-06)	2.21e-01(5.59e-08)	**2.17e-01(6.19e-08)**	2.18e-01(1.94e-07)

Only results of C5 and C6 are presented in Table 12. It has been observed that in most cases the Tr-DMOEA with sqp method achieves the best results, compared with two other Tr-DMOEA variants. The Friedman and Nemenyi statistical tests show that Tr-DMOEA with sqp significantly outperforms the other two variants. After getting the computation time and solution quality comparison results of three Tr-DMOEA variants, it can be concluded that the interior point method in the original Tr-DMOEA is ineffective and extremely inefficient. Although interior point method can achieve the best transferred solutions among three compared methods, it consumes several or even more than ten times of what two other methods consume. In addition, transferred solutions found by interior point method become significantly worse than those found by sqp method after optimisation, which shows that it is not beneficial for the interior point method to improve the efficiency of optimisation, compared with sqp method.

3.4 Summary

This section studies how efficient transfer learning is in DMO through time complexity and computation time analysis, showing that the interior point method in transfer learning [8] is extremely time-consuming when solving the inner problem. Another two inner optimisation methods are computationally studied, to figure out whether the efficiency of transfer learning can be improved and whether the improved efficiency will affect the effectiveness. Experimental results shows that the sqp method achieve a better balance between the transfer learning efficiency and effectiveness of transferred solutions, which makes the changes more proactively and quickly adapted. In addition, the sqp method is more robust to changes as it can obtain good solutions in the first generation after the changes. Lastly, another experiment is conducted to verify whether the purpose of using transfer learning in DMO vanishes, which leverages the computation time of transfer learning to optimise randomly generated solutions. The results show that randomly generated solutions after being optimised using transfer learning time are greatly better than transferred ones, which therefore encourages the proposal of more efficient transfer learning algorithms for DMO.

4. Summary and Outlook

This report studies how to improve the effectiveness of transfer learning in dynamic multi-objective optimisation from the aspect of trying to answer when and how to transfer knowledge in dynamic multi-objective optimisation. It has been found that transfer learning fails on problems with fixed Pareto optimal sets and when changes are small. In addition, the existing used Gaussian kernel function is not ideal. Through avoiding transfer on problems for which transfer learning fails and replacing the Gaussian kernel with a linear one, environmental changes in DMOPs are more proactively and quickly adapted soon after they happen, rather than spending a lot of time adapting to changes. Experimental results show that the proposed approach is able to obtain good solutions in the first generation after the changes, which means that the proposed approach is more robust to changes. Then, this report investigates how to improve the efficiency of transfer learning in dynamic multi-objective optimisation. It has been found that the ‘inner’ optimisation method is the most time-consuming in transfer learning after an experimental study. Other two alternatives of the ‘inner’ optimisation method are adopted to investigate whether the efficiency of transfer learning could be improved and whether the improved efficiency affects the performance of transfer learning. Experimental results show that the greatly enhanced efficiency does not result in large deterioration of the quality of transfer learning. In addition, the good results obtained with the solutions by the ‘sqp’ method in the last generation after the changes mean that Tr-DMOEA with ‘sqp’ is more robust in a long run.

In the future, a potential work is to find another kernel function that can achieve non-linear transfers and does not have the weakness of the Gaussian kernel. Also, another optimisation algorithm can be found to solve the inner problem efficiently and effectively. In addition, other transfer learning methods in the field of machine learning can also be studied to solve DMOPs, the efficiency of which should be also considered. At last, it is important to explore real-world applications of transfer learning based DMO, e.g., in smart manufacturing and smart logistics.

Bibliography

[1] WJ. Kramer, B. Jenkins, RS. Katz. The role of the information and communications technology sector in expanding economic opportunity[J]. Cambridge, MA: Kennedy School of Government, Harvard University, 2007, 22: 1-45.

[2] E. Williams. Environmental effects of information and communications technologies[J]. Nature, 2011, 479(7373): 354-358.

[3] K. Deb, S. Karthik, et al., Dynamic multi-objective optimisation and decision-making using modified NSGA-II: a case study on hydro-thermal power scheduling, in *International Conference on Evolutionary Multi-criterion Optimisation (EMO)*, Springer, 2007, pp. 803–817.

[4] M. Farina, K. Deb, and P. Amato, “Dynamic multi-objective optimisation problems: test cases, approximations, and applications,” *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 5, pp. 425–442, 2004.

[5] S. Yang and X. Yao, Evolutionary Computation for Dynamic Optimisation Problems. Springer, 2013, vol. 490.

[6] J. Ou, J. Zheng, G. Ruan, Y. Hu, J. Zou, M. Li, S. Yang, and X. Tan, “A pareto-based evolutionary algorithm using decomposition and truncation for dynamic multi-objective optimisation,” *Applied Soft Computing*, vol. 85, p. 105673, 2019.

[7] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[8] M. Jiang, Z. Huang, L. Qiu, W. Huang, and G. G. Yen, “Transfer learning-based dynamic multiobjective optimisation algorithms,” *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 4, pp. 501–514, 2017.

[9] G. Ruan, G. Yu, J. Zheng, J. Zou, and S. Yang, “The effect of diversity maintenance on prediction in dynamic multi-objective optimisation,” *Applied Soft Computing*, vol. 58, pp. 631–647, 2017.

[10] A. Zhou, Y. Jin, and Q. Zhang, “A population prediction strategy for evolutionary dynamic multiobjective optimisation,” *IEEE Transactions on Cybernetics*, vol. 44, no. 1, pp. 40–53, 2013.

[11] M. Helbig and A. Engelbrecht, “Benchmark functions for cec 2015 special session and competition on dynamic multi-objective optimisation,” *Dept. Comput. Sci., Univ. Pretoria, Pretoria, South Africa, Rep*, 2015.

[12] M. R. Sierra and C. A. C. Coello, “Improving pso-based multi-objective optimisation using crowding, mutation and-dominance,” in *International conference on evolutionary multi-criterion optimisation (EMO)*. Springer, 2005, pp. 505–519.

[13] Q. Li, J. Zou, S. Yang, J. Zheng, and G. Ruan, “A predictive strategy based on special points for evolutionary dynamic multi-objective optimisation,” *Soft Computing*, vol. 23, no. 11, pp. 3723–3739, 2019.

[14] J. Demsar, “Statistical comparisons of classifiers over multiple data sets,” Journal of Machine Learning Research, vol. 7, no. Jan, pp. 1–30, 2006.

[15] Q. Zhang, A. Zhou, and Y. Jin, “RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm,” IEEE Transactions on Evolutionary Computation, vol. 12, no. 1, pp. 41–63, 2008.

[16] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[17] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, “A theory of learning from different domains,” *Machine Learning*, vol. 79, no. 1-2, pp. 151–175, 2010.

[18] A. Smola, A. Gretton, L. Song, and B. Scholkopf, “A hilbert space ¨ embedding for distributions,” in *International Conference on Algorithmic Learning Theory*. Springer, 2007, pp. 13–31.

[19] J. Shawe-Taylor, N. Cristianini et al., Kernel methods for pattern analysis. *Cambridge University Press*, 2004.

[20] J. Nam, S. J. Pan, and S. Kim, “Transfer defect learning,” in *2013 35th international conference on software engineering (ICSE)*. IEEE, 2013, pp. 382–391.

[21] B. Chen, W. Lam, I. Tsang, and T.-L. Wong, “Extracting discriminative concepts for domain adaptation in text mining,” in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data minin*g. ACM, 2009, pp. 179–188.

[22] S. J. Pan, X. Ni, J.-T. Sun, Q. Yang, and Z. Chen, “Cross-domain sentiment classification via spectral feature alignment,” in *Proceedings of the 19th International Conference on World Wide Web (WWW)*. ACM, 2010, pp. 751–760.

[23] C. C. Coello and M. S. Lechuga, “Mopso: A proposal for multiple objective particle swarm optimisation,” in *Proceedings of the 2002 Congress on Evolutionary Computation (CEC) (Cat. No. 02TH8600)*, vol. 2. IEEE, 2002, pp. 1051–1056.

[24] S. Wright and J. Nocedal, “Numerical optimisation,” *Springer Science*, vol. 35, no. 67-68, p. 7, 1999.

[25] Y. Jin, “Surrogate-assisted evolutionary computation: Recent advances and future challenges,” *Swarm and Evolutionary Computation*, vol. 1, no. 2, pp. 61–70, 2011.

Appendix

Table of acronym

ICT	Information and Communications Technology
DMO	Dynamic multi-objective optimisation
DMOEA	Dynamic multi-objective evolutionary algorithm
DMOPs	dynamic multi-objective optimisation problems
POS	Pareto optimal set
POF	Pareto optimal front
Tr-DMOEAs	Transfer learning-based dynamic multi-objective optimisation algorithms
IGD	Inverted generational distance
RMMEDA	Regularity model-based multi-objective estimation of distribution algorithm
DAL	Domain Adaption Learning
MMD	Maximum Mean Discrepancy
RKHS	Reproducing Kernel Hilbert Space
NSGA-II	Elitist non-dominated sorting genetic algorithm II
MOPSO	multi-objective particle swarm optimisation algorithm (
TCA	Transfer component analysis
Sqp	Sequential quadratic programming