Multi-Problem Surrogates: Transfer Evolutionary Multiobjective Optimization of Computationally Expensive Problems

Alan Tan Wei Min, Yew-Soon Ong, Abhishek Gupta and Chi-Keong Goh

***Abstract*—In most real-world settings, designs are often gradually adapted and improved over time. Consequently, there exists knowledge from distinct (but possibly related) design exercises, which have either been previously completed or are currently in-progress, that may be leveraged to enhance the optimization performance of a particular target optimization task of interest. Further, it is observed that modern day design cycles are typically distributed in nature, and consist of multiple teams working on associated ideas in tandem. In such environments, vast amounts of related information can become available at various stages of the search process corresponding to some ongoing target optimization exercise. Successfully exploiting this knowledge is expected to be of significant value in many practical settings where solving an optimization problem from scratch may be exorbitantly costly or time consuming. Accordingly, in this paper, we propose an *adaptive knowledge reuse* framework for surrogate-assisted multiobjective optimization of computationally expensive problems, based on the novel idea of *multi-problem surrogates*. This idea provides the capability to acquire and spontaneously transfer learned models across problems, facilitating *efficient global optimization*. The efficacy of our proposition is demonstrated on a series of synthetic benchmark functions, as well as two practical case studies.**

***Index Terms*—Multiobjective optimization, efficient global optimization, multi-problem surrogates, knowledge transfer.**

I. Introduction

In most real-world problems of interest, there generally exist more than a single criterion to be taken into account while evaluating the quality of a solution. Different from the single-objective variant, solving multiobjective optimization problems (MOPs) results in a diverse set of solutions that represent the best possible trade-offs between the different objectives. These solutions are traditionally labelled as Pareto optimal solutions [1]. It is noted that evolutionary algorithms (EAs) have generally been cited as effective methods for MOPs, primarily due to their *implicit parallelism* which enables concurrent convergence towards the entire set of Pareto optimal solutions [2]–[4]. Some popular examples of multi-objective evolutionary algorithms (MOEAs) include NSGA-II [5], PAES [6], SPEA2 [7], MOEA/D [8], HypE [9], etc.

Despite their gaining popularity, a common challenge arising in the use of MOEAs, and other population-based meta-heuristic methods, is the need to conduct a large number of function evaluations before a set of near-optimal solutions can be achieved. This is especially an impediment in many real-world applications where each function evaluation can be considerably expensive (either in terms of time and/or any other valuable resources). A commonly adopted approach for addressing this challenge, as has been previously studied in various works [10]–[16], is the use of surrogates or meta-models to approximate the expensive functions, and subsequently search on the approximation targeting more *efficient global optimization* (EGO) [17].

A well-established algorithm that is based on the EGO framework but extends to computationally expensive MOPs is ParEGO [18]. The method follows the path of *Bayesian optimization* where the probabilistic nature of Gaussian process regression models are leveraged to intelligently sample those points in the search space that are likely to be Pareto optimal with high probability. This is often achieved by assigning an *expected improvement* measure quantifying the merit of evaluation at a new design point, which is shown to provide a favourable balance between *exploration* and *exploitation* of the search space. The key advantage of probabilistic modeling of objectives is that it offers a principled manner of selecting subsequent points to be evaluated by the true (expensive) function, as opposed to the more randomized schemes of MOEAs. The resultant sample efficiency leads to considerable savings in computational effort. A related idea was also presented in [19], where Gaussian process models in conjunction with the MOEA/D algorithm were used to process several candidate solutions in parallel to converge towards the Pareto set. More recently, in [20], a Gaussian process surrogate-assisted extension of the reference vector guided EA was shown to have particular efficacy for computationally expensive *many-objective* problems. Alternatively, in [21], the S-metric selection-based EGO (SMS-EGO) was proposed to optimize the S-metric using covariance matrix adaptation evolution strategies. In [22], radial basis functions were preferred to create the surrogate response surfaces. Notably, the balance between exploration and exploitation was maintained by selecting evaluation points through a combination of various metrics. In addition to the above, a plethora of works exploiting some form of function

This work was conducted within the Rolls-Royce@NTU Corporate Lab with support from the National Research Foundation (NRF) Singapore under the Corp Lab@University Scheme. The research is supported in part by the Data Science and Artificial Intelligence Research Centre (DSAIR) at the Nanyang Technological University. (*Corresponding author: Abhishek Gupta*)

Alan Tan Wei Min is a doctoral candidate at the School of Computer Science and Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798 (e-mail: alantwm@gmail.com).

Yew-Soon Ong and Abhishek Gupta are with the Data Science and Artificial Intelligence Research Centre, School of Computer Science and Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798 (e-mail: {ASYSONG, ABHISHEKG}@ntu.edu.sg).

Chi-Keong Goh is with Rolls-Royce Seletar, 6 Seletar Aerospace Rise, Singapore 797575 (e-mail: ChiKeong.Goh@Rolls-Royce.com)

approximation in evolutionary computation exist, comprehensive surveys of which are available in [23], [24].

While substantial performance gains have been achieved by surrogate-assisted optimization techniques, these methods are limited in the sense that they only consider data originating from the current target problem of interest, often overlooking many other rich sources of information. Particularly for expensive problems, the scarcity and difficulty in acquiring enough target data to build sufficiently informative surrogate models may hinder the search progress (commonly referred to as the *cold start* problem), thereby encouraging us to look elsewhere for sources of relevant information. To this end, it is noted that real-world problems seldom exist in isolation. It is common for new designs to draw heavily from previously completed designs. In fact, the ability to effectively build upon past experiences and "lessons learned" is of critical importance in achieving success in today's competitive world. As a consequence, there generally exists a veritable well of relevant knowledge that can be exploited in order to improve the efficiency of the optimization process. Within the domain of machine learning, this concept of utilizing knowledge from related source tasks to improve the learning of a target task is the central premise behind transfer learning [25], a field of study that has seen considerable success in a wide variety of application areas. However, an equivalent notion of knowledge transfer in the domain of optimization has largely eluded researchers, with only few related works available in the literature. Accordingly, in this paper, we propose an efficient *Transfer Evolutionary Multiobjective Optimization with Multi-Problem Surrogates* (TEMO-MPS) approach that exploits knowledge embedded in surrogate models from distinct (but possibly related) design exercises to augment the optimization of a new target problem of interest.

The salient feature of our proposal is its *adaptiveness*, in the sense that it accounts for surrogate models that may originate from several sources, for example, from previously completed design exercises, or from associated design projects conducted in tandem by different teams. Thus, it is natural to expect that some of the sources may be more related (or relevant) to the target task than others. In such settings, the multi-problem surrogate (MPS) approach is capable of automatically modulating the extent to which transfer must occur, and, in principle, guarantees effectiveness of the search. Notably, the practical viability of our proposal is strongly supported by modern cloud computing platforms that enable large-scale data storage and seamless communication facilities.

It should be noted that, while existing surrogate-assisted MOEAs for computationally expensive problems have demonstrated the benefits of utilizing approximation methods such as Gaussian processes (alternatively known as Kriging) and radial basis functions, none in the literature have considered the incorporation of MPS within the context of multiobjective optimization. Further, it is worth mentioning that although the algorithmic realization of MPS has some aspects in common with ensemble learning methods [26], it is conceptually unique in the sense that traditional ensemble learning techniques typically do not accommodate potentially diverse data streams originating from differing sources.

For a detailed exposition of the ideas discussed heretofore, the remainder of this paper is organized as follows. Section II presents a brief review of related work on the application of knowledge transfer in evolutionary computation. Section III presents an overview of MOPs, and associated objective aggregation techniques for decomposition-based optimization approaches. Section IV introduces our proposed TEMO-MPS framework, and Section V details an instantiation of that framework. Section VI and VII present experimental results that showcase the efficacy of our proposition on synthetic benchmark problems, as well as two practical case studies, respectively. Section VIII concludes the paper and provides directions for future research.

II. Background Study

The notion of knowledge transfer, while fairly well-researched in machine learning (e.g. [25]), has been comparatively unexplored in the domain of optimization problem-solving. In this section, we present a brief review of some applications of knowledge transfer that appear in optimization.

In [27], [28], transfer across problems is carried out in a simple manner by using the final population from a genetic algorithm applied on a source task to intialize the population of a related target task. Mashaiov and Tal [29] applied a similar idea of *genetic transfer* as a means to overcome the bootstrapping problem of controller design in evolutionary robotics, where it was found that a traditional randomly generated initial population (with no transferred elements from prior design exercises) had a much lower chance of completing the assigned task. Termed 'family boostrapping', solutions from an evolved related source task were therefore reused as part of the initial solutions of the target task to jump-start the search process. More recently, contributions in [30]–[32] have put forward approaches that facilitate knowledge transfer in genetic programming by extracting and reusing building-blocks of knowledge (in the form of code fragments) from source tasks. The efficacy of the idea has been demonstrated on several image classification problems [30], symbolic regression tasks [31], [33], as well as boolean problems [32], yielding noteworthy performance. Distinct from the above, where the problems tackled are temporally separated, in [34], the concept of evolutionary multi-tasking was introduced to tackle multiple related tasks *concurrently* using a single population of evolving solutions. In this case, the transfer of knowledge between tasks was achieved implicitly through the sharing of genetic material contained in a unified search space, which resulted in accelerated convergence in a variety of practical optimization problems [35], [36].

While the aforementioned works rely primarily on the direct transfer of genetic building-blocks from one task to another, in contrast, the following papers utilize a higher-order model-based transfer scheme in conjunction with evolutionary search. For example, Lim et al. [37] investigated the use of knowledge transfer within a simultaneous optimization and problem learning paradigm. The authors proposed a classifier-assisted EA to extract the feasibility structure of a particular source task, which was then reused to seive out infeasible

solutions in a related target task. In another approach, Feng et al. [38], [39] proposed a memetic computation paradigm that captures transferable knowledge embedded in optimized solutions of graph-based problems in the form of a learned distance metric.

In general, the majority of existing methodologies of incorporating automatic knowledge transfer in evolutionary computation are found to be reliant on the intrinsic-nature of the evolutionary selection pressure to gradually cull out transferred information that may be irrelevant or harmful [27]. While this approach has been proven successful in many occasions, there is a chance that excessive harmful transfer across completely unrelated tasks can severly disrupt the search performance leading to decelerated convergence. As an alternative, there have been some recent attempts to incorporate knowledge transfer in Bayesian optimization by explicitly modelling the correlation between the optimization problems [40], [41] - instead of placing the onus entirely on evolution. However, most existing efforts in this regard are restrictive in scope as they only cater to single-objective optimization, and have almost exclusively been studied in the context of a specific (hyperparameter tuning) problem in machine learning.

To the best of our knowledge, the present paper is the first to propose an *adaptive* scheme combining transfer learning with evolutionary methods for multiobjective optimization of computationally expensive problems - based on the central idea of multi-problem surrogates. Motivated by [42], not only does our method leverage on past problem-solving experiences, but can also account for information streaming in live from other optimization exercises progressing in tandem. In addition to testing our proposal on a variety of synthetic benchmark problems, we demonstrate the efficacy of TEMO-MPS on practical problems in machine learning (namely, automatic hyperparameter tuning) as well as in engineering design where related problems routinely recur, thereby providing an ideal environment for transfer optimization to flourish.

III. Multiobjective Optimization

A. MOP Formulation

Without loss of generality, the basic statement of an MOP for a minimization task can be stated as follows:

$$\begin{aligned} &\text{Minimize:} \quad \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})), \\ &\text{subject to:} \quad \boldsymbol{x} = (x_1, x_2, \ldots, x_d) \in \prod_{i=1}^{d} [a_i, b_i], \end{aligned} \tag{1}$$

where m is the number of objectives, d is the dimensionality of the search (or design) space, and $-\infty < a_i < b_i < \infty$ for all $i = 1, 2, \ldots, d$ represent the box-constraints of the search space. A detailed discussion on various aspects of MOPs and associated optimization techniques can be found in [1].

B. Objective Aggregation Procedures

The methods described in [5]–[9] showcase a variety of approaches for tackling MOPs. Popular EAs such as NSGA-II evaluate solutions based on the Pareto dominance relation, along with a neighbourhood density measure in order to ensure solution diversity along the Pareto front. An alternative technique is to decompose the MOP into a series of single objective problems, via some objective aggregation technique [43], [44]. Approaching the problem in this manner has several notable advantages including computational efficiency, the option of leveraging the wide array of well-established single objective optimization techniques [45], and the relative ease of approximating a single function at a time using the surrogate model. Two of the most commonly employed methods in this regard, namely, the weighted sum and Tchebycheff approaches, are briefly discussed next.

1) Weighted Sum Approach: This method is based on minimizing convex combinations of the objectives, which can be stated as:

$$\text{Minimize:} \quad y(\boldsymbol{x}|\boldsymbol{\omega}) = \sum_{j=1}^{m} \omega_j f_j. \tag{2}$$

Here, y is the aggregation of all objectives, f_j is the j^{th} objective function, and $\boldsymbol{\omega}$ represents a vector of strictly positive weights such that:

$$\boldsymbol{\omega} = (\omega_1, \omega_2, \ldots, \omega_m) | \sum_{j=1}^{m} \omega_j = 1. \tag{3}$$

Optimal solutions found using the weighted sum aggregation approach can potentially cover the entire Pareto set (given a sufficiently diverse set of weight vectors) as long as the Pareto front is convex. However, this approach is not compatible with nonconvex Pareto fronts [8], which leads us to the next (generally more flexible) technique.

2) Tchebycheff Approach: In this case, the transformed problem after objective aggregation is stated as:

$$\text{Minimize :} \quad y(\boldsymbol{x}|\boldsymbol{\omega}) = \max_{j=1}^{m} \{\omega_j f_j - z_j^*\}, \tag{4}$$

where z_j^* is a reference point with respect to the j^{th} objective, and y, $\boldsymbol{\omega}$ are interpreted in the manner stated previously. An important characteristic of the Tchebycheff aggregation is that it is well suited for handling both convex and nonconvex Pareto fronts [19]. Although this approach may lead to non-differentiable points, the derivative-free nature of an EA provides an elegant way of overcoming the challenge. Consequently, the use of Tchebycheff aggregation in conjunction with EAs is seen as an effective means of handling general MOPs for which little is known beforehand about the nature of the Pareto front.

IV. Basic Framework of TEMO-MPS

While existing surrogate-assisted optimization techniques are indeed an improvement over predecessors in the context of computationally expensive problems, they only consider exploiting the data that originates from the target problem of interest. However, this data, by definition, is scarce and incurs significant computational cost to obtain. With this in mind, the proposed TEMO-MPS framework incorporates the capability of adapting and utilizing knowledge embedded in

surrogate models from various other sources, either from the past or even from problem-solving exercises progressing in tandem. Our basic idea, outlined in Algorithm 1, consists of four primary steps.

In the first step, an initial set of solutions of cardinality N_0 is generated, either randomly or using a design of experiments procedure. These solutions are evaluated using the exact objective functions and stored in P_{eval}. Next, the solutions in P_{eval} are aggregated using an appropriate aggregation method (e.g. Tchebycheff approach) based on a randomly sampled weight vector $\boldsymbol{\omega}^{(i)}$ to form a subproblem $y(\boldsymbol{x}|\boldsymbol{\omega}^{(i)})$. In the third step, using the aggregated solutions in P_{eval} and all available source models at iteration t, that may originate from distinct design exercises, the multi-problem surrogate model h_{MPS} is built. In the present study, this is achieved through a model-based *transfer stacking* approach, where pretrained source models encapsulating the relevant knowledge are used instead of raw source datasets [46]. Transfer stacking includes a meta-regression stage which involves *optimally* combining multiple base surrogates, namely, diverse source model(s) and a preliminary target model, to form the final MPS. In the fourth step, the EA within TEMO-MPS is executed to maximize a computationally cheap figure of merit $\alpha(\boldsymbol{x})$ (see Section V-C), derived from the learned surrogate model h_{MPS}, which quantifies the relative utility of evaluating a particular point in the search space. Subsequently, the evolved solution, $\boldsymbol{x}_{next}$, is evaluated using the exact (expensive) objective functions and appended to P_{eval}.

It should be noted that at every iteration, a weight-vector, $\boldsymbol{\omega}^{(i)}$ is randomly sampled and the corresponding single-objective subproblem is approximated and solved. Therefore, in the sections that follow, the aggregated objective $y(\boldsymbol{x}|\boldsymbol{\omega}^{(i)})$ corresponding to the sampled weight-vector will be represented by $y(\boldsymbol{x})$ or simply, y, for the sake of brevity.

Algorithm 1 General TEMO-MPS Framework

Inputs: $Source\,Models$, weight-vectors $\boldsymbol{\omega}^{(1)}, \ldots, \boldsymbol{\omega}^{(N_\omega)}$
Output: Nondominated solutions of P_{eval}.

1. Initialization: Evaluate an initial set of N_0 solutions and store as P_{eval}.

while stopping conditions are not met **do**

2. Aggregation: Aggregate the function values of P_{eval} according to a randomly sampled weight-vector $\boldsymbol{\omega}^{(i)}$ to obtain $y(\boldsymbol{x}|\boldsymbol{\omega}^{(i)})$.

3. Multi-Problem Surrogate: Build multi-problem surrogate model h_{MPS} that approximates $y(\boldsymbol{x}|\boldsymbol{\omega}^{(i)})$ using aggregated solutions in P_{eval} and available $Source\,Model$(s) at time t (e.g. Algorithm 2).

4. Evolutionary Optimization: Execute internal EA to maximize a computationally cheap figure of merit $\alpha(\boldsymbol{x})$, derived from the surrogate model h_{MPS}, which ascertains the next point to evaluate using the true (expensive) objective functions. The evolved solution $\boldsymbol{x}_{next}$ is evaluated and appended to P_{eval}.

end while

V. An Instantiation of TEMO-MPS

While there can be different variations of the framework described in Section IV, one possible instantiation is presented here. This variation utilizes a Tchebycheff aggregation approach (see Equation 4), Gaussian process surrogate models with linear meta-regression, and an *expected improvement* measure to quantify the merit of evaluating a new point using the expensive objective functions. In the subsections that follow, details regarding the various components that characterize this instantiation will be described.

A. Gaussian Process Surrogate Model Building

In the present instantiation of TEMO-MPS, the probabilistic Gaussian process is chosen as the base surrogate model for the transfer stacking approach. Gaussian processes are formulated as stochastic processes that are fully specified by a mean and covariance function [47]. They have been widely accepted as effective approximation methods that provide predictions together with uncertainty estimates, as a measure of statistical confidence. In particular, they facilitate the development of computationally cheap figures of merit, such as the expected improvement measure, that provide a balance between exploration and exploitation during the optimization search.

Given training data that consists of n_T pairs of inputs and outputs $\{(\boldsymbol{x}^{(i)}, y^{(i)}), i = 1, 2, \ldots, n_T\}$ - where $\boldsymbol{x}^{(i)}$ is the ith row of the $n_T \times d$-dimensional input matrix $\boldsymbol{X}$ and $y^{(i)}$ is the ith instance in the vector of output labels $\boldsymbol{y}$ - the predictive distribution at an unknown input, $\boldsymbol{x}^{(*)}$, is given by the following:

$$\hat{y}(\boldsymbol{x}^{(*)}) = K_*(K + \sigma_n^2 I)^{-1}\boldsymbol{y}, \tag{5}$$

$$\hat{\sigma}^2(\boldsymbol{x}^{(*)}) = K_{**} - K_*(K + \sigma_n^2 I)^{-1}K_*^T, \tag{6}$$

where $\hat{y}$ and $\hat{\sigma}^2$ are the mean and posterior variance, respectively, of the predictive distribution at $\boldsymbol{x}^{(*)}$. Further, $K_* = [k(\boldsymbol{x}^{(*)}, \boldsymbol{x}^{(1)}), \ldots, k(\boldsymbol{x}^{(*)}, \boldsymbol{x}^{(n_T)})]$ and $K_{**} = k(\boldsymbol{x}^{(*)}, \boldsymbol{x}^{(*)})$, where k represents the covariance function, K is the associated covariance matrix, and σ_n is a noise parameter. In this paper, the following automatic relevance determination (ARD) squared exponential covariance function is used throughout:

$$k(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}) = \theta_f \exp\Big(\sum_{D=1}^{d} \frac{-(x_D^{(i)} - x_D^{(j)})^2}{2\theta_D^2}\Big), \tag{7}$$

where $\boldsymbol{\theta} = \{\sigma_n, \theta_f, \theta_1, \theta_2, \ldots, \theta_d\}$ are $d+2$ hyperparameters to be learned by minimizing the negative logarithm of the marginal likelihood function:

$$l = -\frac{1}{2}log|K + \sigma_n^2 I| - \frac{1}{2}\boldsymbol{y}^T(K + \sigma_n^2 I)^{-1}\boldsymbol{y} - \frac{n_T}{2}log2\pi. \tag{8}$$

B. Transfer Stacking Gaussian Process MPS

As previously stated, in TEMO-MPS, multi-problem surrogates are built, facilitating the incorporation of knowledge

Algorithm 2 Transfer Stacking Gaussian Process MPS with Linear Meta-Regression

Input: $\boldsymbol{X}, \boldsymbol{y}$, B Source Models

Output: h_{MPS}

Step 1 - Building Target Model: Learn an initial target Gaussian process model h_T using all of $\boldsymbol{X}, \boldsymbol{y}$.

for $i = 1, 2, \ldots, n_T$ **do**

Step 2 - Generating the Out-of-Sample Target Feature Vector ($\hat{\boldsymbol{y}}_T$): Construct a Gaussian process model tmp_model_i using $\boldsymbol{\theta}$, $\boldsymbol{X}[!i]$, and $\boldsymbol{y}[!i]$. Here, $\boldsymbol{\theta}$ represents learned hyperparameters of h_T from Step 1, which are directly reused without retraining. Further, $[!i]$ signifies that the ith instance $(\boldsymbol{x}^{(i)}, y^{(i)})$ is dropped from $\boldsymbol{X}$ and $\boldsymbol{y}$ following a *leave-one-out scheme*. Thereafter, the target feature $\hat{y}_T^{(i)}$ is given by the mean of the predictive distribution of tmp_model_i at $\boldsymbol{x}^{(i)}$, conditioned on $\boldsymbol{X}[!i]$ and $\boldsymbol{y}[!i]$.

end for

Step 3 - Generating Source Features ($\hat{\boldsymbol{y}}_S$): For each $Source\,Model$, calculate the output corresponding to target points in $\boldsymbol{X}$. The observed outputs are stored as $\hat{y}_{S,1}^{(i)}, \ldots, \hat{y}_{S,B}^{(i)}$ for $i = 1, \ldots, n_T$.

Step 4 - Calculating the Meta-Regression Coefficients: Using the target and source features obtained in Steps 2 and 3, respectively, solve the optimization problem shown in Eq. 9 to obtain the mixture coefficients $(\boldsymbol{a})$.

Step 5 - Output: Save $(\boldsymbol{a}, Source\,Models, h_T)$ to form h_{MPS}.

from related design exercises as they become available. This is achieved through transfer stacking, a model-based approach that combines multiple base surrogate models using a meta-regression algorithm. In this instantiation of TEMO-MPS, the transfer stacking module, shown in Algorithm 2, utilizes a linear meta-regression for this purpose.

Given target data $\boldsymbol{X}$, $\boldsymbol{y}$ of n_T solutions evaluated so far, together with B *pretrained* source models, the mixture coefficients of meta-regression $\boldsymbol{a} = (a_{S,1}, \ldots, a_{S,B}, a_T)$ are obtained by solving the following optimization problem minimizing the squared error of *out-of-sample* predictions:

$$\begin{aligned} \text{Minimize:} \quad & SE(\boldsymbol{a}) = \sum_{i=1}^{n_T} (\sum_{j=1}^{B} a_{S,j} \hat{y}_{S,j}^{(i)} + a_T \hat{y}_T^{(i)} - y^{(i)})^2 \\ \text{Subject to:} \quad & \sum_{j=1}^{B} a_{S,j} + a_T = 1 \\ & a_{S,j} \geq 0, \text{for } j = 1, ..., B \\ & a_T \geq 0 \end{aligned} \tag{9}$$

where $\hat{y}_{S,1}^{(i)}, \ldots, \hat{y}_{S,B}^{(i)}$ are predictions by the source models $h_{S,1}, \ldots, h_{S,B}$ at $\boldsymbol{x}^{(i)}$, and $\hat{y}_T^{(i)}$ is an out-of-sample prediction of an initial target model at $\boldsymbol{x}^{(i)}$ - see Algorithm 2 for details.

By considering the out-of-sample predictions, the intention is to leverage the source models to help improve generalization performance. Once the coefficients of the meta-regression model are calculated, the final prediction of the MPS model at an unknown input $\boldsymbol{x}^{(*)}$ is given by the following:

$$\hat{y}(\boldsymbol{x}^{(*)}) = \sum_{j=1}^{B} a_{S,j} \hat{y}_{S,j}(\boldsymbol{x}^{(*)}) + a_T \hat{y}_T(\boldsymbol{x}^{(*)}) \tag{10}$$

1) Source-Target Similarity Capture: A key advantage of the proposed transfer stacking algorithm is its *adaptiveness*. This implies that, in principle, the method is able to automatically account for the similarities between the source and target optimization tasks. To elaborate, if a particular source model $h_{S,j}$ is highly correlated with the target, the magnitude of the coefficient $a_{S,j}$ will likely be large - which implies that a large weight is assigned to $h_{S,j}$. On the other hand, if a source model is irrelevant to the target, then a small coefficient (~ 0) will be learned instead - implying that the corresponding model is given small weight. As a result of the adaptive nature of transfer stacking, the practitioner is relieved of the task of accounting for the possibility of harmful (negative) transfer which is often a hindrance in transfer learning [25].

To substantiate our claims, herein we theoretically examine the relationship between the estimated source-target similarity and the learned mixture coefficient values $a_{S,j}$ (for $j = 1, \ldots, B$) and a_T for a transfer stacking model. We begin by defining a symmetric $(B+1) \times (B+1)$ correlation matrix C. The (p, q)th element of the matrix is given as $C_{pq} \equiv \sum_{i=1}^{n_T} \epsilon_p^{(i)} \epsilon_q^{(i)}$, where

$$\epsilon_q^{(i)} = \begin{cases} y^{(i)} - \hat{y}_{S,q}^{(i)} & \text{if } q \leq B \\ y^{(i)} - \hat{y}_T^{(i)} & \text{for } q = B+1 \end{cases} \tag{11}$$

is the prediction error of the qth surrogate model on the ith training input. For the mathematical derivations below, we make the following assumptions that simplify the analysis.

- The prediction error of the qth surrogate model has zero mean when averaged across the n_T target inputs, i.e., $\sum_{i=1}^{n_T} \epsilon_q^{(i)} = 0$. Note that no restriction is placed on $|\epsilon_q^{(i)}|$.
- The prediction errors of all (source + target) surrogate models are mutually uncorrelated. In other words, $C_{pq} = 0,\ \forall (p, q)\ s.t.\ p \neq q$.

Based on Equation 11, we interpret the similarity between the qth source and the target problem to be represented by the generalization error of the qth model measured on the entire (target) training set, i.e., $C_{qq} = \sum_{i=1}^{n_T} [\epsilon_q^{(i)}]^2$. If C_{qq} is large then the qth source model is intuitively expected to be less relevant to the target. On the other hand, if C_{qq} is small, then the qth source is likely closely related to the target.

Following Equation 10, and based on the constraints of Equation 9, the squared prediction error of the MPS model on the ith out-of-sample training input can be expressed in terms of the $\epsilon_q^{(i)}$'s as:

$$(y^{(i)} - \hat{y}^{(i)})^2 = (\sum_{q=1}^{B+1} a_q \epsilon_q^{(i)})^2 \tag{12}$$

where,

$$a_q = \begin{cases} a_{S,q} & \text{if } q \leq B \\ a_T & \text{for } q = B+1 \end{cases}. \quad (13)$$

Enforcing the assumption that $C_{pq} = 0,\ \forall\ p \neq q$, the squared error ($SE$) of MPS measured over the entire training set, as given in Equation 9, can be rewritten as [48]:

$$SE(\boldsymbol{a}) = \sum_{q=1}^{B+1} (a_q^2 C_{qq}). \quad (14)$$

Using the method of Lagrange multipliers to solve for the coefficients of the transfer stacking model, we have,

$$\frac{\partial}{\partial a_p}\left[\sum_{q=1}^{B+1}(a_q^2 C_{qq}) - 2\lambda(\sum_{q=1}^{B+1} a_q - 1)\right] = 0. \quad (15)$$

Simplifying this equation, we arrive at the condition:

$$a_p = \frac{\lambda}{C_{pp}}. \quad (16)$$

This result provides the insight that the coefficient a_p of the pth surrogate model is inversely proportional to the generalization error of the model calculated on the entire (target) dataset. Equivalently, it can be said that as the source-target similarity decreases, the value of the corresponding mixture coefficient a_p also decreases. As a result, its impact on the overall generalization performance of the transfer stacking model is automatically reduced.

2) Stepwise description of transfer stacking: As summarized in Algorithm 2, the procedure of generating a transfer stacking Gaussian process MPS with linear meta-regression is as follows. The steps are based on the classical stacked generalization algorithm proposed in [49] for ensemble learning (without transfer), but have been extended herein to utilize models derived from possibly diverse sources.

In the first step, an initial target Gaussian process surrogate model, h_T, is trained using the dataset $\boldsymbol{X}, \boldsymbol{y}$, as described in Section V-A. Subsequently, in the second step, a target feature vector ($\hat{\boldsymbol{y}}_T$), comprising out-of-sample predictions $\hat{y}_T^{(i)}$ at $\boldsymbol{x}^{(i)}$, for $i = 1, ..., n_T$, is generated. This is achieved through a variant of the *leave-one-out cross-validation scheme* [17]. To elaborate, the out-of-sample prediction at $\boldsymbol{x}^{(i)}$ is given by the mean of the predictive distribution of an intermediate Gaussian process model tmp_model_i which employs the same hyperparameter settings as h_T (without retraining). The only distinction between tmp_model_i and h_T is that the ith instance $(\boldsymbol{x}^{(i)}, y^{(i)})$ is left-out from the dataset of tmp_model_i. Thus, for all practical purposes, the input $\boldsymbol{x}^{(i)}$ is unseen to tmp_model_i, with its predictive distribution conditioned on the remaining $n_T - 1$ samples. The rationale behind using this leave-one-out scheme is that it provides legitimate out-of-sample predictions without the need for costly retraining of the intermediate Gaussian process models. Note that the theoretical validity of the approach stems from the observation in [17] that the log marginal likelihood (used for tuning the hyperparameters) is typically not affected significantly by the exclusion of a single (left-out) data point.

In the third step, source features ($\hat{\boldsymbol{y}}_S$), i.e. the predictions $\hat{y}_{S,1}^{(i)}, \ldots, \hat{y}_{S,B}^{(i)}$ by the source models $h_{S,1}, \ldots, h_{S,B}$ on the points in $\boldsymbol{X}$, are generated. With $\hat{\boldsymbol{y}}_S$ and $\hat{\boldsymbol{y}}_T$ in hand, the mixture coefficients ($\boldsymbol{a}$) are then calculated in step four by solving Equation 9 using any convex optimization method. Finally, in step five, the mixture coefficients, source models, and target model are saved to form h_{MPS}, which is subsequently used for the target optimization task.

3) Salient feature of transfer stacking: It is important to note that the calculation of the figure of merit, used in step four of the TEMO-MPS framework (Algorithm 1), generally requires the standard deviation of the predictive distribution in addition to the mean. In this regard, an important feature of transfer stacking with Gaussian process based surrogates and linear meta-regression emerges from the following fact:

Fact: *Let $Y_1, Y_2, \ldots, Y_N$ be independent normal random variables that are specified by means $\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_N$ and variances $\hat{\sigma}_1^2, \hat{\sigma}_2^2, \ldots, \hat{\sigma}_N^2$. Then, any linear combination of these variables, given by $\sum_{i=1}^N a_i Y_i$, is also normally distributed as $\mathcal{N}(\sum_{i=1}^N a_i \hat{y}_i, \sum_{i=1}^N a_i^2 \hat{\sigma}_i^2)$.*

Accordingly, the posterior variance of the predictive distribution at $\boldsymbol{x}^{(*)}$ obtained via transfer stacking is given by:

$$\hat{\sigma}^2(\boldsymbol{x}^{(*)}) = \sum_{j=1}^{B} a_{S,j}^2 \hat{\sigma}_{S,j}^2(\boldsymbol{x}^{(*)}) + a_T^2 \hat{\sigma}_T^2(\boldsymbol{x}^{(*)}) \quad (17)$$

where $\hat{\sigma}_{S,1}(\boldsymbol{x}^{(*)}), \ldots, \hat{\sigma}_{S,B}(\boldsymbol{x}^{(*)})$ and $\hat{\sigma}_T(\boldsymbol{x}^{(*)})$ are the standard deviations of the predictive distributions of the source and target Gaussian process models, respectively, at $\boldsymbol{x}^{(*)}$.

C. The Expected Improvement Figure of Merit

The figure of merit $\alpha(\boldsymbol{x})$, in step four of the TEMO-MPS framework (Algorithm 1), provides a simple and quick way of approximating the quality of a particular point in the search space, derived in a manner that accounts for the uncertainty of the surrogate Gaussian process prediction. While many such measures have been proposed in the literature, the *expected improvement* is perhaps most commonly used as it known to provide a favorable balance between exploration and exploitation of unknown search spaces [17]–[19]. Therefore, the expected improvement measure is also employed in TEMO-MPS, with the only change being that $\hat{y}$ and $\hat{\sigma}$ are obtained from Equation 10 and Equation 17, respectively.

The improvement in the approximated objective value $\hat{y}(\boldsymbol{x}^{(*)})$ of a so far unevaluated point $\boldsymbol{x}^{(*)}$, with respect to the current best evaluated function value y_{min}, can be written as $I(\boldsymbol{x}^{(*)}) = \max(y_{min} - \hat{y}(\boldsymbol{x}^{(*)}), 0)$. Accordingly, utilizing $\hat{y}(\boldsymbol{x}^{(*)})$ (from Equation 10) and $\hat{\sigma}(\boldsymbol{x}^{(*)})$ (from Equation 17) as the predicted mean and standard deviation of the Gaussian process regression model, the expected improvement measure at $x^{(*)}$ is given as:

$$\alpha_{EI}(\boldsymbol{x}^{(*)}) = E[I(\boldsymbol{x}^{(*)})] = (y_{min} - \hat{y}(\boldsymbol{x}^{(*)}))\Phi(\frac{y_{min} - \hat{y}(\boldsymbol{x}^{(*)})}{\hat{\sigma}(\boldsymbol{x}^{(*)})}) + \hat{\sigma}(\boldsymbol{x}^{(*)})\phi(\frac{y_{min} - \hat{y}(\boldsymbol{x}^{(*)})}{\hat{\sigma}(\boldsymbol{x}^{(*)})}) \quad (18)$$

where $\Phi(\cdot)$ and $\phi(\cdot)$ are the standard normal distribution and density functions, respectively.

VI. Synthetic Numerical Examples

In this section, we compare the performance of the proposed TEMO-MPS framework against ParEGO [6] and the popular NSGA-II [5] on a series of synthetic examples.

Parameter	Setting
Population Size	20
Number of Generations	25
Index of Polynomial Mutation	10
Index of SBX	10

TABLE I: NSGA-II Parameter Settings

Parameter	Setting
Maximum Function Evaluations	500
Number of Weight Vectors, N_ω	11 for $m = 2$; 15 for $m = 3$
Number of Internal EA iterations	500
Size of Internal EA Population	20
Mutation Probability	$1/d$
SBX and Mutation Index	10
Crossover Probability	0.9

TABLE II: TEMO-MPS and ParEGO Parameter Settings

A. Test Functions

For demonstrating the effectiveness of TEMO-MPS, two groups of synthetic *multimodal* test functions are constructed. These two groups are based on functions found in the literature [50]. The first group is derived from DTLZ1. As shown in Equation 19, the variations are as follows: (1) the value 20π within the cosine term of the original DTLZ1 was changed to 2π in order to reduce the number of local fronts (a similar strategy has been applied in [18] as the original function is excessively rugged for surrogate modeling), and (2) in order to create a suite of related optimization tasks, the function is parameterized by two variables δ_1 and δ_2. The resultant functions are denoted by DTLZ1b$-\delta_1, \delta_2$. Note that when $\delta_1 = \delta_2 = 0$, DTLZ1b$-0, 0$ is equivalent to the test function used in [18]. Further, notice that the degree of similarity between the different tasks in the group can be modified by changing δ_1, δ_2. For instance, DTLZ1b-0,0 is much more similar to DTLZ1b-10,0 than it is to DTLZ1b-30,3. In other words, larger difference in δ_1, δ_2 indicates lower similarity (this feature shall be demonstrated in Section VI-F).

$$
\begin{aligned}
&\text{Minimize } f_1 = \frac{1}{2} x_1 (1 + g) \\
&\text{Minimize } f_2 = \frac{1}{2} (1 - x_1)(1 + g) \\
&\qquad g = (100 + \delta_1) \Bigg[7 + \delta_2 + \sum_{i \in \{2,\ldots,8\}} (x_i - 0.5)^2 \\
&\qquad\quad - \cos(2\pi (x_i - 0.5)) \Bigg] \\
&\qquad x_i \in [0, 1], i \in \{1, \ldots, d\}, d = 8
\end{aligned}
\tag{19}
$$

The second group is derived from DTLZ3 [50]. As shown in Equation 20, DTLZ3b is also parameterized by δ_1 and δ_2. Resultant functions are denoted by DTLZ3b$-\delta_1, \delta_2$.

$$
\begin{aligned}
&\text{Minimize } f_1 = (1 + g) \cos(x_1 \pi/2) \cos(x_2 \pi/2) \\
&\text{Minimize } f_2 = (1 + g) \cos(x_1 \pi/2) \sin(x_2 \pi/2) \\
&\text{Minimize } f_3 = (1 + g) \sin(x_1 \pi/2) \\
&\qquad g = (100 + \delta_1) \Bigg[6 + \delta_2 + \sum_{i \in \{3,\ldots,8\}} (x_i - 0.5)^2 \\
&\qquad\quad - \cos(2\pi (x_i - 0.5)) \Bigg] \\
&\qquad x_i \in [0, 1], i \in \{1, \ldots, d\}, d = 8
\end{aligned}
\tag{20}
$$

B. Hypervolume Measure

The metric used throughout the paper to assess the quality of the Pareto front approximation is the hypervolume measure (i.e., the Labesgue integral) which quantifies the size of the region in the objective space (with respect to a user-defined reference point) that is dominated by a set of data points [51], [52]. For all experiments on synthetic test functions that are subsequently described, the reference point is provided with the tabulated results or in the figure captions.

C. Experimental Details

For fairness of comparison, both ParEGO and the instantiation of TEMO-MPS used in these experiments utilize the Tchebycheff aggregation (see Equation 4), the expected improvement measure as the figure of merit (see Equation 18), and identical evolutionary operators (for the internal EA), such that when no source models are available, the performance of TEMO-MPS mimics that of ParEGO. Specifically, the simulated binary crossover (SBX) [53], the polynomial mutation [54], and the $(\mu + \lambda)$-ES [55] selection strategy, are employed. In all our numerical experiments, we have $\mu = \lambda$. As ParEGO originally uses a steady-state population update procedure, necessary comparative studies were carried out in order to ensure that the change in selection strategy does not alter the overall performance, yielding similar outcomes.

Details of the parameter settings for each algorithm are shown in Table I (NSGA-II) and Table II (TEMO-MPS and ParEGO). Note that we use a small population size for NSGA-II as we are dealing with low computational budget. Each set of experiments was repeated 30 times and a Wilcoxon signed rank test at 95% confidence interval was applied to determine the statistical significance of the results presented in Tables III. An analysis of the initial DOE (population) size used for TEMO-MPS and ParEGO is discussed next.

D. Determining the Initial DOE (Population) Size in ParEGO and TEMO-MPS

First, the effects of the initial number of DOE data points N_0 on the performance of both ParEGO and TEMO-MPS are investigated. In this experiment, DTLZ3b$-10, 0$ is the target

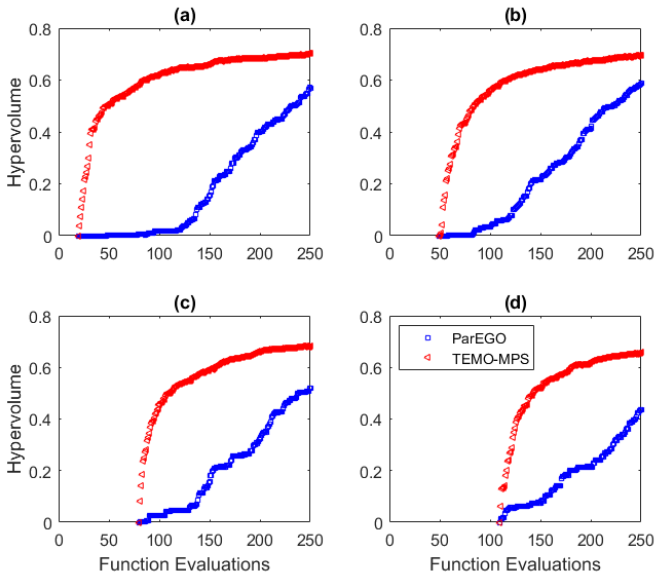

Fig. 1: Hypervolume vs Function Evaluations for DTLZ3b$-10,0$ for ParEGO (in blue) and TEMO-MPS (in red). Source models for TEMO-MPS is assumed to be drawn from DTLZ3b$-0,0$ and DTLZ1b$-0,0$. Figures (a)-(d) corresponds to $N_0 = 2(d+2), 5(d+2), 8(d+2)$, or $11(d+2)$ initial DOE data points, respectively. The results are averaged across 30 runs.

task solved using ParEGO and TEMO-MPS with $N_0 = 2(d+2), 5(d+2), 8(d+2)$, or $11(d+2)$. The source models for TEMO-MPS are assumed to be drawn from ParEGO runs on DTLZ3b$-0,0$ and DTLZ1b$-0,0$. Note that this involves a separate model for each weight vector $\boldsymbol{\omega}$.

From the results presented in Figure 1 (a)-(d), it is evident that in order to reach the same performance level of ParEGO, the TEMO-MPS algorithm takes significantly fewer function evaluations. To elaborate, the hypervolume measure achieved by ParEGO at 250 function evaluations is accomplished by TEMO-MPS within approximately 75 evaluations for $N_0 = 2(d+2)$ (in the best case) and within 132 evaluations for $N_0 = 11(d+2)$ (in the worse case). This equates to approximately 70% and 47% function evaluations saved, respectively, as opposed to ParEGO.

Note from Figure 1 that TEMO-MPS experiences a jump-start in the evolutionary search process as compared to ParEGO, demonstrating the effect of knowledge transfer from source models in improving the performance of the algorithm. Based on these results, for the remainder of the experiments, N_0 is set to $2(d+2)$ throughout as it provides the largest percentage of function evaluations saved, and better scalability for higher values of dimensionality d [22].

E. TEMO-MPS versus ParEGO and NSGA-II

In this section, to better showcase the capabilities of TEMO-MPS, the proposed algorithm is compared against ParEGO and NSGA-II on a set of 18 test functions generated by varying $\delta_1 \in \{10, 20, 30\}$ and $\delta_2 \in \{1, 2, 3\}$ for DTLZ1b and DTLZ3b. *It is important to emphasize that, while tackling a particular task, TEMO-MPS incorporates source models from a pool generated during ParEGO runs for all the 17 other test functions (i.e., with exception of the δ_1, δ_2 pair currently being solved as the target task). Note that this involves a separate model for each weight vector $\boldsymbol{\omega}$.*

Consistent with the observations of Section VI-D, the results for DTLZ1b presented in Figure 2 indicate that TEMO-MPS generally shows superior performance characteristics than ParEGO. These results are detailed in Table III at the 50, 100, 200 and 500 function evaluations stages and we observe that TEMO-MPS outperforms ParEGO in all cases.

For DTLZ3b, as shown in Figure 2 and Table III, TEMO-MPS outperforms ParEGO in all cases at 50 and 100 function evaluations, 7 out of 9 cases at 200 function evaluations and 8 out of 9 cases at 500 function evaluations. As TEMO-MPS jump-starts the evolutionary search process driven by knowledge transfer, ParEGO can only manage to narrow the gap towards the end of 500 function evaluations. On the other hand, NSGA-II, is found to be markedly inferior to both ParEGO and TEMO-MPS on all 18 test functions in terms of the convergence rate. This result is due to the fact that NSGA-II does not utilize surrogate models to assist in identifying new candidate solutions and, consequently, shows relatively poor convergence characteristics over a small computational budget. In summary, these results are encouraging as they indicate that TEMO-MPS is able to adequately leverage the knowledge embedded in surrogate models of related source optimization tasks, thereby giving confidence to the efficacy of the proposed framework.

F. Investigating the Effects of Source-Target Similarity

Building on the previous line of inquiry, we examine the effects of solving a target problem using surrogate models from increasingly dissimilar source tasks. Specifically, the target problem, DTLZ1b$-0,0$, is solved using TEMO-MPS incorporating three different sets of source models generated by varying $\delta_2 \in \{1, 2, 3\}$ and setting $\delta_1 = 10$. To elaborate, in the first experiment only DTLZ1b$-10,1$ acts as the source, in the second experiment DTLZ1b$-10,2$ is the source, and in the third experiment only DTLZ1b$-10,3$ is the source.

We begin by calculating the Pearson correlation coefficient which measures linear correlation between the *aggregated* objective function landscape of the target problem and each of the source tasks. Note that a decreasing Pearson correlation value points towards decreasing source-target similarity. As shown in Figure 3(a), such a decreasing trend in the linear correlation between the various sources and the target is observed, with DTLZ1b$-10,1$ being the most correlated, and DTLZ1b$-10,3$ being the least. To visualize the reason behind the drop in correlation, we present a scatter plot in Figure 3(b) that shows aggregated objective function values of all (source + target) optimization tasks corresponding to the same inputs. As is revealed, DTLZ1b$-10,3$ has a much wider spread and is located farther away from DTLZ1b$-0,0$ as compared to DTLZ1b$-10,1$, which is spread over a narrow band located much closer to the DTLZ1b$-0,0$ curve.

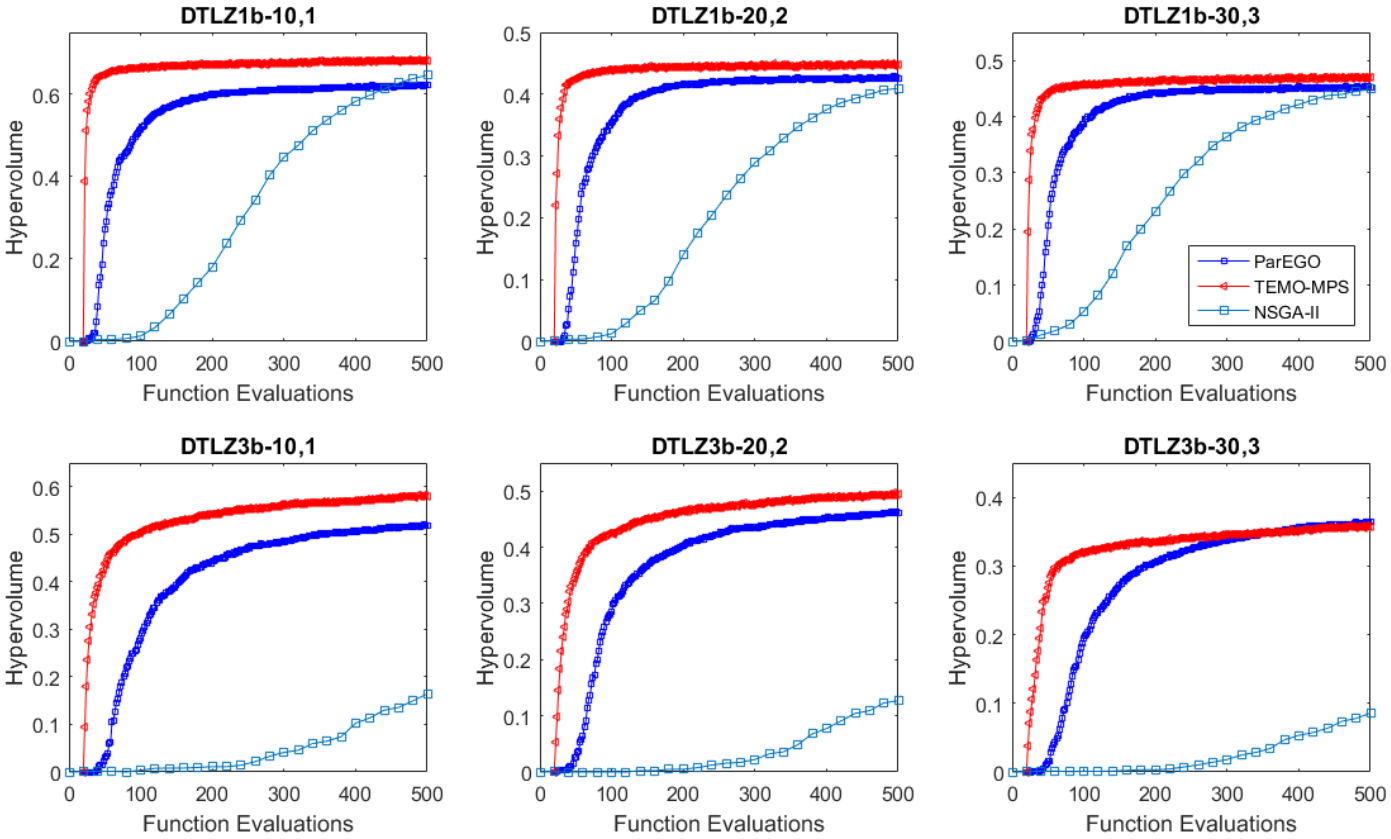

Fig. 2: Hypervolume vs. Function Evaluations comparison between ParEGO, TEMO-MPS, and NSGA-II for DTLZ1b (**TOP-ROW**), DTLZ3b (**BOTTOM-ROW**), and various values of δ_1, δ_2. The results in the figure are averaged across 30 independent runs.

TABLE III: Comparing the mean performance and standard deviation of ParEGO, NSGA-II and TEMO-MPS (see Section VI-E). Statistically significant superior results evaluated using a Wilcoxon signed rank test with a 95% confidence interval are presented in bold.

Experiments	Bound	50 Generations				100 Generations				200 Generations				500 Generations					
		ParEGO		TEMO-MPS		ParEGO		TEMO-MPS		ParEGO		TEMO-MPS		ParEGO		TEMO-MPS		NSGA-II	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
DTLZ1b - 10,1	75,75	0.2612	0.1109	**0.6508**	**0.0149**	0.5139	0.0305	**0.6620**	**0.0118**	0.5992	0.0154	**0.6709**	**0.0099**	0.6214	0.0114	**0.6822**	**0.0115**	0.6444	0.0261
DTLZ1b - 20,1	75,75	0.1835	0.1091	**0.5970**	**0.0099**	0.4738	0.0262	**0.6120**	**0.0065**	0.5504	0.0118	**0.6165**	**0.0082**	0.5720	0.0128	**0.6263**	**0.0090**	0.5993	0.0274
DTLZ1b - 30,1	75,75	0.1596	0.1124	**0.5338**	**0.0104**	0.4204	0.0349	**0.5463**	**0.0065**	0.5030	0.0158	**0.5549**	**0.0072**	0.5201	0.0106	**0.5628**	**0.0090**	0.5063	0.0438
DTLZ1b - 10,2	120, 120	0.1876	0.1092	**0.5050**	**0.0092**	0.4192	0.0231	**0.5149**	**0.0054**	0.4851	0.0103	**0.5231**	**0.0065**	0.4996	0.0075	**0.5291**	**0.0069**	0.5010	0.0249
DTLZ1b - 20,2	120, 120	0.1455	0.0826	**0.4226**	**0.0081**	0.3520	0.0203	**0.4379**	**0.0056**	0.4153	0.0113	**0.4430**	**0.0061**	0.4263	0.0074	**0.4475**	**0.0061**	0.4100	0.0250
DTLZ1b - 30,2	120, 120	0.1141	0.0660	**0.3512**	**0.0070**	0.2945	0.0179	**0.3620**	**0.0058**	0.3442	0.0107	**0.3665**	**0.0066**	0.3528	0.0070	**0.3705**	**0.0064**	0.3264	0.0315
DTLZ1b - 10,3	200, 200	0.2796	0.1261	**0.5795**	**0.0139**	0.5043	0.0213	**0.5968**	**0.0058**	0.5612	0.0103	**0.6019**	**0.0080**	0.5830	0.0107	**0.6070**	**0.0067**	0.6010	0.0169
DTLZ1b - 20,3	200, 200	0.2167	0.0917	**0.5170**	**0.0096**	0.4452	0.0190	**0.5297**	**0.0063**	0.5009	0.0091	**0.5352**	**0.0056**	0.5206	0.0075	**0.5397**	**0.0067**	0.5249	0.0160
DTLZ1b - 30,3	200, 200	0.1897	0.1056	**0.4426**	**0.0119**	0.3888	0.0196	**0.4564**	**0.0056**	0.4405	0.0073	**0.4630**	**0.0054**	0.4509	0.0071	**0.4695**	**0.0070**	0.4496	0.0189
DTLZ3b - 10,1	150,150,150	0.0283	0.0488	**0.4327**	**0.0540**	0.2784	0.0964	**0.5027**	**0.0452**	0.4420	0.0429	**0.5414**	**0.0384**	0.5187	0.0320	**0.5825**	**0.0307**	0.1642	0.0833
DTLZ3b - 20,1	150,150,150	0.0155	0.0356	**0.3416**	**0.0424**	0.1918	0.0743	**0.4178**	**0.0278**	0.3426	0.0440	**0.4557**	**0.0251**	0.4174	0.0350	**0.4868**	**0.0304**	0.1045	0.0606
DTLZ3b - 30,1	150,150,150	0.0064	0.0156	**0.2321**	**0.0569**	0.1225	0.0520	**0.3012**	**0.0236**	0.2459	0.0394	**0.3319**	**0.0238**	0.3204	0.0201	**0.3748**	**0.0249**	0.0492	0.0401
DTLZ3b - 10,2	300,300,300	0.0699	0.0830	**0.4343**	**0.0529**	0.4009	0.0628	**0.5329**	**0.0394**	0.5143	0.0394	**0.5690**	**0.0294**	0.5562	0.0362	**0.6016**	**0.0274**	0.2198	0.0510
DTLZ3b - 20,2	300,300,300	0.0266	0.0417	**0.3455**	**0.0647**	0.2832	0.0572	**0.4245**	**0.0376**	0.4025	0.0342	**0.4635**	**0.0268**	0.4613	0.0241	**0.4949**	**0.0269**	0.1275	0.0511
DTLZ3b - 30,2	300,300,300	0.0214	0.0384	**0.2584**	**0.0367**	0.1871	0.0461	**0.3084**	**0.0195**	0.2887	0.0262	**0.3324**	**0.0170**	0.3420	0.0249	**0.3590**	**0.0196**	0.0703	0.0430
DTLZ3b - 10,3	450,450,450	0.0854	0.0904	**0.4145**	**0.0785**	0.4275	0.0668	**0.4980**	**0.0355**	0.5350	0.0349	0.5295	0.0359	0.5871	0.0325	**0.5667**	**0.0353**	0.2179	0.0558
DTLZ3b - 20,3	450,450,450	0.0310	0.0585	**0.3209**	**0.0555**	0.2929	0.0890	**0.3892**	**0.0274**	0.4187	0.0382	0.4197	0.0258	0.4694	0.0290	**0.4524**	**0.0249**	0.1503	0.0346
DTLZ3b - 30,3	450,450,450	0.0150	0.0325	**0.2670**	**0.0375**	0.1914	0.0535	**0.3172**	**0.0161**	0.3052	0.0253	**0.3347**	**0.0179**	0.3636	0.0261	0.3596	0.0203	0.0860	0.0391

Examining the differences in the hypervolume measure at 50, 75 and 100 function evaluations of TEMO-MPS, as shown in Figure 3(c), the impact of increasing Pearson correlation coefficient is revealed. As expected, as increasingly relevant source models are used in TEMO-MPS, the overall optimization performance is monotonically improved.

In order to explain the observations above, we demonstrate here that performance improvements in TEMO-MPS are indeed an outcome of the improved accuracy of the multi-problem surrogate, and not an artefact of the blessings of uncertainty as may sometimes be the case [56]. In particular, we compare the generalization performance on the target optimization task of transfer stacking Gaussian process MPS against traditional Gaussian process. The performance is measured via the root mean squared error (RMSE). The results, shown in Figure 3(d), indicate that as the correlation between

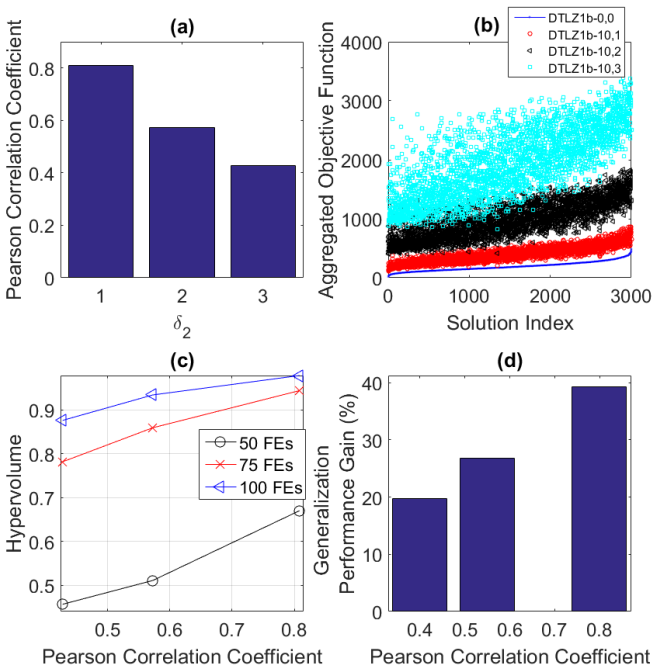

Fig. 3: **(a)** Pearson correlation coefficient of DTLZ1b$-0,0$ and DTLZ1b$-10,\delta_2$. **(b)** aggregated objective values of DTLZ1b$-0,0$, DTLZ1b$-10,1$, DTLZ1b$-10,2$ and DTLZ1b$-10,3$ given the same inputs. **(c)** Hypervolume vs. Pearson correlation coefficient for TEMO-MPS at the specified function evaluation intervals (FEs). **(d)** Generalization performance gain (in %) of MPS vs. Pearson correlation coefficient.

source and target increases, the *Generalization Performance Gain* $= (RMSE_{NT} - RMSE_{MPS})/RMSE_{NT}$ also improves; where NT refers to the no transfer case.

VII. Practical Examples

A. A Case Study in Engineering Process Design

One of the key beneficiaries of our proposed TEMO-MPS framework is expected to be the domain of complex engineering design where a variety of products, often with only incremental changes, are produced over time. This clearly gives rise to the potential for knowledge transfer across design exercises which are bound to share much in common, thereby cutting down on the often exorbitantly expensive design time. To demonstrate, in a naive evolutionary-based design setting, it is common for several thousand numerical simulations to be required before an acceptable solution can be found. Further, real-world high fidelity numerical simulations can take anywhere from several minutes to several hours for a single run, thereby further compounding the challenges.

To this end, in this section, we demonstrate the efficacy of TEMO-MPS against other popular methods from the literature on a real-world finite element simulation-based optimization case study from the composites manufacturing industry. As is well known, such simulations are often computationally expensive and can form a major bottleneck in the design cycle. Thus, being able to speed-up the simulation-based optimization stage can be of significant value in engineering design settings.

In particular, herein we conisder the multiobjective optimization of the *resin transfer moulding* (RTM) process corresponding to two different composite parts that share geometric similarities. Before we proceed to the details of our experimental settings, we present a brief overview of the steps involved in a typical RTM cycle, which, as shown in Figure 4, is a popular method for the manufacture of high-quality composite parts in the aero and auto industries. For full details on RTM and other composites manufacturing processes, the reader is referred to [57], [58]. The equipment used for an RTM cycle includes a metallic mold machined according to the geometry of the fibre-reinforced polymer part to be manufactured. The fibrous reinforcement is first placed inside the mold cavity. The mold is then closed, thereby compressing the fibre mat to the final desired thickness of the composite part. Finally, a liquid thermosetting resin is injected at high pressure into the closed mold until all the remaining voids within it are fully saturated.

Based on the aforementioned steps, the RTM cycle gives rise to four key design variables, namely, velocity of mold closure $V_{closure}$, resin injection pressure P_{inj}, temperature of the mold T_{mold}, and temperature of the resin T_{resin}. The aim of the design stage of an RTM cycle for a particular composite part is thus to tune the design variables so as to achieve minimum processing time while simultaneously minimizing the capital layout and running costs. Notably, the setup and running costs are treated indirectly by minimizing the peak internal forces (originating from resin pressure and fibre stress) that may be generated within the part during the manufacturing process, as this dictates the size and cost of peripheral equipments (such as a hydraulic press) that may be required. The final MOP formulation of interest can therefore be written as [59]:

$$\begin{aligned} \text{Minimize:} \quad & (\text{Mold Filling Time, Peak Internal Force}) \\ \text{s.t.} \quad & Variables \in Feasible\, Design\, Space \end{aligned} \tag{21}$$

In this demonstration, we consider the manufacturing of two glass-fibre reinforced epoxy composite discs which differ in some aspects. To elaborate, one of the discs has a diameter of 1 m and a fibre volume fraction of 35%. The other disc has a smaller diameter of 0.8 m but a larger fibre volume fraction of 50%. Details of material properties, and the partial differential equations governing the finite element simulations are not reproduced herein for the sake of brevity, but can be found in [58]. While there are clear distinctions, there is also expected to exist some underlying synergy between the two MOPs due to the similarity of the involved part geometries.

For the numerical experiments, the upper and lower bounds of the design variables are provided in Table IV. The parameter settings used for TEMO-MPS, ParEGO, and NSGA-II are identical to those considered for the synthetic benchmark functions. Note that in the results in Figure 5, the focus is on the manufacturing of the composite disc with diameter 1.0 m and fibre volume fraction of 35%. Thus, for TEMO-MPS,

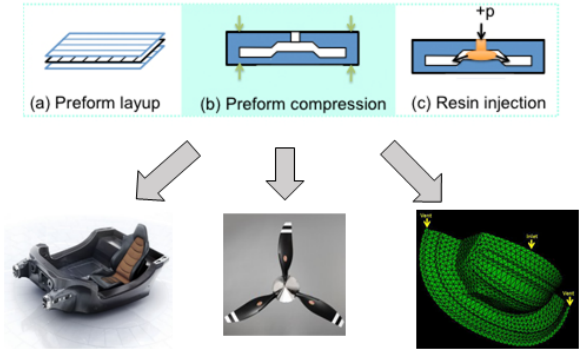

Fig. 4: An illustration of a typical RTM process workflow for fiber-reinforced polymer composite parts [36].

TABLE IV: Upper and lower bounds of the design variables in the RTM manufacturing cycle

Design Variable	Lower Bound	Upper Bound
$V_{closure}$	1 mm/min	10mm/min
P_{inj}	1 MPa	10 MPa
T_{mold}	293 K	348 K
T_{resin}	293 K	348 K

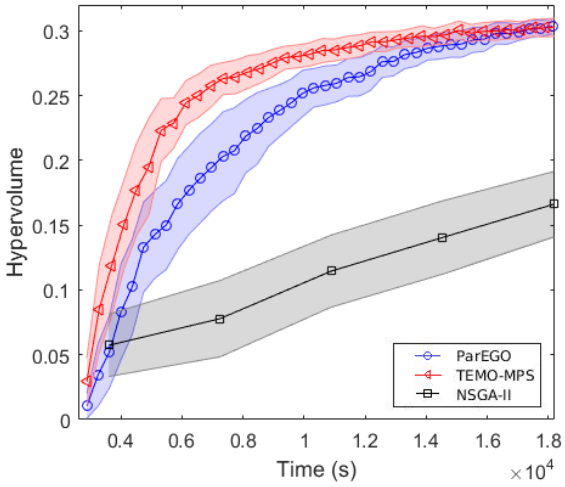

Fig. 5: Convergence trend comparison between TEMO-MPS, ParEGO, and NSGA-II for the simulation-based optimization of an RTM cycle applied to a composite disc of 1 m diameter and fibre volume fraction of 35% . TEMO-MPS makes use of source models from a surrogate model building exercise on a composite disc of 0.8 m diameter and fibre volume fraction of 50%. Results presented herein are averaged across 30 runs and based on the hypervolume reference point of $(40\text{s}, 40\text{N})$. The shaded area spans $1/2$ standard deviation on either side of the mean.

the source models are obtained from the design optimization of the composite disc with diameter 0.8 m. Further, we simulate the effect of a constant stream of incoming source data from a distinct project assumed to be progressing in tandem. In other words, the source data grows even as TEMO-MPS runs for the target task of interest. In particular, at the start of the TEMO-MPS run, the source dataset contains 100 instances. By the 10th iteration, the source dataset also grows to 110 instances, and so on. Note that in the real-world, the viability of data continuously streaming in from various sources is made possible by modern-day cloud computing platforms.

As is revealed in Figure 5, TEMO-MPS experiences an initial impetus to the optimization search process, which leads to improved convergence characteristics in comparison to ParEGO over the provided computational budget of 100 exact function evaluations. Specifically, to achieve a hypervolume measure of 0.25 using high fidelity simulations, TEMO-MPS requires $\sim$ 6700 seconds relative to the $\sim$ 9950 seconds of ParEGO, which represents a savings of $\sim$ 3250 seconds or 32%. Based on our prior investigations for the synthetic functions, this observation may be explained by the exploitation of source surrogate models in TEMO-MPS which leads to a better approximation of the target optimization function landscape from the get-go. As is also consistent with earlier numerical experiments, NSGA-II is unable to match the performance of TEMO-MPS or even ParEGO within limited computational budgets since it does not make use of any surrogate-assistance.

B. A Case Study in Automatic Hyperparameter Tuning of Machine Learning Models

A principled approach towards determining an optimal set of hyperparameters for machine learning models can be a time consuming affair as it typically requires a rigorous cross-validation procedure. For neural networks in particular, the hyperparameters include the number of hidden units, regularization coefficient, learning rate, training epochs, activation functions, etc. Appropriately selecting them can often be the difference between a useless model and state-of-the-art performance. The task of tuning hyperparameters becomes especially daunting when dealing with big data where the number of data points and/or features are massive. Notably, in such cases, each evaluation of a given set of hyperparameters is computationally expensive as it entails complete training and validation of the underlying model.

An key component of this study is to demonstrate the possibility of utilizing a small version of a dataset to efficiently explore hyperparameter settings for the full (significantly larger) dataset [40]. We demonstrate the use of TEMO-MPS for this purpose, in particular, on the optimization of the hyperparameters of a multilayer feedforward neural network applied to a multi-output classification problem. As a simple proof-of-concept, we consider a yeast gene functional classification problem with 103 features, 14 non-exclusive output classes and 2417 data instances [60]. In our experiments, the hyperparameters tuned are: (1) the number of hidden layers (n_{hidden}), (2) the number of nodes for each layer (n_{nodes}), (3) regularization coefficient (γ), and (4) the number of training epochs (max_epochs). The lower and upper bounds of these parameters are shown in Table V.

In multi-output classification, each instance is classified into a number of *non-exclusive* classes. As a result, it is possible

TABLE V: Upper and lower bounds of the input hyperparameters to the neural network.

Hyperparameters (Input)	Lower Bound	Upper Bound
n_{hidden}	1	10
n_{nodes}	1	100
γ	0	1
max_epochs	1	500

to perform learning as a multiobjective optimization problem, where the accuracy on each output constitutes a different objective function to be optimized. The rationale behind our approach lies in the possibility that each output may have a different set of optimal hyperparameters. In other words, a set of hyperparameters that is ideal for a particular output may in fact be poor for another. Therefore, a trade-off in the performance with respect to the multiple outputs may often be pursued - which can be hard to capture through a single scalar objective function.

For simplicity, in this study, we restrict the yeast gene functional classification problem to three outputs (thereby forming a three-objective optimization problem). Note that for a given set of hyperparameters, the underlying neural network model was trained using a scaled conjugate gradient backpropagation algorithm with a mean squared error performance measure.

Next, we show that by leveraging on a small dataset, one can quickly optimize for larger datasets. To elaborate, TEMO-MPS incorporates source models generated from a ParEGO run of 100 function evaluations on a small subset of the original dataset. Specifically, the neural network in the source task is trained using only 500 data points and is validated on the remaining 1917 points. This simulates a fast exploratory run on an inexpensive smaller subset of the training data in order to perform a preliminary performance analysis before committing to the much more expensive larger dataset. The target task utilizes all available data and a 30% hold-out cross validation scheme. Both ParEGO and TEMO-MPS are given a budget of 500 function evaluations on the target task, with results averaged over 30 repeat runs.

The results presented in Figure 6 indicate that TEMO-MPS shows superior convergence characteristics relative to ParEGO. Specifically, the hypervolume achieved by ParEGO after 500 function evaluations is accomplished by TEMO-MPS after only 347 function evaluations. This equates to approximately 153 function evaluations saved (or 30.6% savings). The promising outcome serves to demonstrate the potential application of TEMO-MPS for accelerating hyperparameter tuning, which is an important aspect of machine learning.

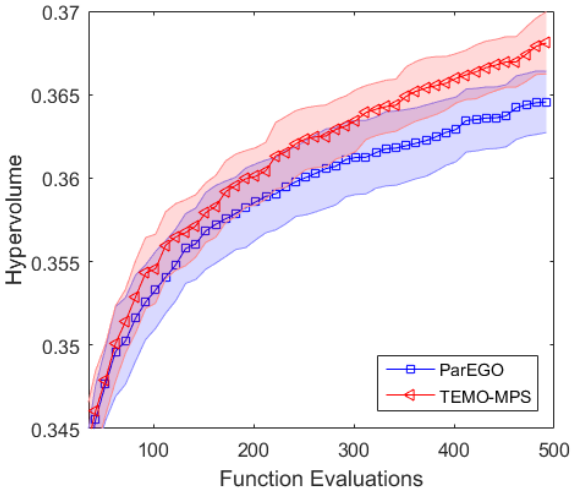

Fig. 6: Evolution of Hypervolume vs Function Evaluations for ParEGO and TEMO-MPS on a multiobjective formulation of the hyperparameter tuning problem of a feedforward neural network applied to a yeast gene functional multi-output classification problem. All results are averaged across 30 independent runs and based on a reference point of [1,1,1]. The shaded area spans $1/2$ standard deviation on either side of the mean.

VIII. Conclusion

In this paper, a novel framework for adaptive knowledge transfer across expensive multiobjective optimization problems has been introduced. The concept of *multi-problem surrogates* (MPS) is proposed for the first time, providing the capability of reusing information gained from distinct (but possibly related) problem-solving experiences in order to augment the optimization process of a target task of interest. Our proposed algorithmic contribution herein is labelled as TEMO-MPS.

Experimental results on synthetic test functions, a complex composite materials manufacturing case study, and a case study in automatic hyperparameter tuning of machine learning models show that TEMO-MPS can significantly outperform ParEGO, saving as much as 70% of function evaluations through relevant knowledge exploitation. A deeper analysis of the results reveals that the similarity between the source and target tasks affects the achievable performance of TEMO-MPS. In particular, the adaptive nature of the MPS module in TEMO-MPS is theoretically shown to play a key role in regulating the contributions of the source models. When the sources are highly correlated to the target, the MPS exploits the relationships to greatly speed-up convergence characteristics for the target optimization task. On the other hand, when the sources are unrelated to the target, their possible negative effects are minimized automatically.

In addition to providing encouraging results that demonstrate the utility of knowledge transfer in real-world problems, the proposed framework also opens doors to a plethora of research opportunities to further explore the scope of multi-problem surrogates in optimization. A particular point of interest is the treatment of knowledge transfer across tasks with heterogeneous search spaces, e.g., when the search space dimensionality is different for different optimization tasks. While the present paper is restricted to the handling of tasks with homogeneous search spaces, our future works will focus on extensions to heterogeneity.

References

[1] K. Deb, *Multi-objective optimization using evolutionary algorithms*, vol. 16. John Wiley & Sons, 2001.

[2] M. N. Le, Y. S. Ong, S. Menzel, C.-W. Seah, and B. Sendhoff, "Multi co-objective evolutionary optimization: Cross surrogate augmentation for computationally expensive problems," in *2012 IEEE Congress on Evolutionary Computation*, IEEE, Brisbane, Australia, June 10–15 2012. doi: 10.1109/CEC.2012.6252915.

[3] H. Ishibuchi and T. Murata, "A multi-objective genetic local search algorithm and its application to flowshop scheduling," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 28, no. 3, pp. 392–403, 1998.

[4] C. A. C. Coello, D. A. Van Veldhuizen, and G. B. Lamont, *Evolutionary algorithms for solving multi-objective problems*, vol. 242. Springer, 2002.

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

[6] J. Knowles and D. Corne, "The Pareto Archived Evolution Strategy: A new baseline algorithm for pareto multiobjective optimisation," in *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, vol. 1, pp. 98–105, IEEE, 1999.

[7] E. Zitzler, M. Laumanns, L. Thiele, *et al.*, "SPEA2: Improving the strength pareto evolutionary algorithm," in *Eurogen*, vol. 3242, pp. 95–100, 2001.

[8] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on evolutionary computation*, vol. 11, no. 6, pp. 712–731, 2007.

[9] J. Bader and E. Zitzler, "HypE: An algorithm for fast hypervolume-based many-objective optimization," *Evolutionary computation*, vol. 19, no. 1, pp. 45–76, 2011.

[10] W. Song and A. J. Keane, "Surrogate-based aerodynamic shape optimization of a civil aircraft engine nacelle," *AIAA Journal*, vol. 45, no. 10, pp. 2565–2574, 2007.

[11] Y. S. Ong, P. B. Nair, and A. J. Keane, "Evolutionary optimization of computationally expensive problems via surrogate modeling," *AIAA Journal*, vol. 41, pp. 687–696, April 2003.

[12] Y. S. Ong, P. B. Nair, and K. Y. Lum, "Max-min surrogate-assisted evolutionary algorithm for robust design," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 4, pp. 392–404, 2006.

[13] Y. Jin, M. Olhofer, and B. Sendhoff, "A framework for evolutionary optimization with approximate fitness functions," *IEEE Transactions on evolutionary computation*, vol. 6, no. 5, pp. 481–494, 2002.

[14] C. Sun, J. Ding, J. Zeng, and Y. Jin, "A fitness approximation assisted competitive swarm optimizer for large scale expensive optimization problems," *Memetic Computing*, pp. 1–12, 2016.

[15] Z. Zhou, Y. S. Ong, P. B. Nair, A. J. Keane, and K. Y. Lum, "Combining global and local surrogate models to accelerate evolutionary optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 1, pp. 66–76, 2007.

[16] D. Lim, Y. Jin, Y. S. Ong, and B. Sendhoff, "Generalizing surrogate-assisted evolutionary computation," *IEEE Trans. Evol. Comput.*, vol. 14, pp. 329–355, December 2010.

[17] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[18] J. Knowles, "ParEGO: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 50–66, 2006.

[19] Q. Zhang, W. Liu, E. Tsang, and B. Virginas, "Expensive multiobjective optimization by MOEA/D with gaussian process model," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 3, pp. 456–474, 2010.

[20] T. Chugh, Y. Jin, K. Miettinen, J. Hakanen, and K. Sindhya, "A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization," *IEEE Transactions on Evolutionary Computation*, 2016. 10.1109/TEVC.2016.2622301.

[21] W. Ponweiser, T. Wagner, D. Biermann, and M. Vincze, "Multiobjective optimization on a limited budget of evaluations using model-assisted $\mathcal{S}$-metric selection," in *International Conference on Parallel Problem Solving from Nature*, pp. 784–794, Springer, 2008.

[22] T. Akhtar and C. A. Shoemaker, "Multi objective optimization of computationally expensive multi-modal functions with rbf surrogates and multi-rule selection," *Journal of Global Optimization*, vol. 64, no. 1, pp. 17–32, 2016.

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

[24] Y. Jin, "A comprehensive survey of fitness approximation in evolutionary computation," *Soft computing*, vol. 9, no. 1, pp. 3–12, 2005.

[25] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, pp. 1345–1359, October 2010.

[26] T. G. Dietterich, "Ensemble methods in machine learning," in *Multiple classifier systems*, pp. 1–15, Springer, 2000.

[27] S. J. Louis and J. McDonnell, "Learning with case-injected genetic algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 4, pp. 316–328, 2004.

[28] B. Koçer and A. Arslan, "Genetic transfer learning," *Expert Systems with Applications*, vol. 37, no. 10, pp. 6997–7002, 2010.

[29] A. Moshaiov and A. Tal, "Family bootstrapping: A genetic transfer learning approach for onsetting the evolution for a set of related robotic tasks," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pp. 2801–2808, IEEE, 2014.

[30] M. Iqbal, B. Xue, H. Al-Sahaf, and M. Zhang, "Cross-domain reuse of extracted knowledge in genetic programming for image classification," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 4, pp. 569–587, 2017.

[31] E. Haslam, B. Xue, and M. Zhang, "Further investigation on genetic programming with transfer learning for symbolic regression," in *Evolutionary Computation (CEC), 2016 IEEE Congress on*, pp. 3598–3605, IEEE, 2016.

[32] M. Iqbal, W. N. Browne, and M. Zhang, "Reusing building blocks of extracted knowledge to solve complex, large-scale boolean problems," *IEEE Transactions on Evolutionary Computation*, vol. 18, pp. 465–480, September 2014.

[33] D. O'Neill, H. Al-Sahaf, B. Xue, and M. Zhang, "Common subtrees in related problems: A novel transfer learning approach for genetic programming," in *Evolutionary Computation (CEC), 2017 IEEE Congress on*, pp. 1287–1294, IEEE, 2017.

[34] A. Gupta, Y. S. Ong, and L. Feng, "Multifactorial evolution: towards evolutionary multitasking," *IEEE Trans. Evol. Comput.*, vol. 20, pp. 343–357, July 2016.

[35] A. Gupta, Y. S. Ong, L. Feng, and K. C. Tan, "Multiobjective multifactorial optimization in evolutionary multitasking," *IEEE Trans. on Cybern.*, vol. 47, pp. 1652–1665, May 2016.

[36] Y. S. Ong and A. Gupta, "Evolutionary multitasking: A computer science view of cognitive multitasking," *Cognitive Computation*, vol. 8, pp. 125–142, March 2016.

[37] D. Lim, Y. S. Ong, A. Gupta, C. K. Goh, and P. S. Dutta, "Towards a new praxis in optinformatics targeting knowledge re-use in evolutionary computation: simultaneous problem learning and optimization," *Evolutionary Intelligence*, vol. 9, pp. 203–220, December 2016.

[38] L. Feng, Y. S. Ong, M.-H. Lim, and I. W. Tsang, "Memetic search with interdomain learning: A realization between CVRP and CARP," *IEEE Trans. Evol. Comput.*, vol. 19, pp. 644–658, October 2015.

[39] L. Feng, Y.-S. Ong, A.-H. Tan, and I. W. Tsang, "Memes as building blocks: a case study on evolutionary optimization+ transfer learning for routing problems," *Memetic Computing*, vol. 7, no. 3, pp. 159–180, 2015.

[40] K. Swersky, J. Snoek, and R. P. Adams, "Multi-task bayesian optimization," in *Advances in Neural Information Processing Systems 26*, pp. 2004–2012, Curran Associates, Inc., Lake Tahoe, NV, December 5–10 2013.

[41] M. Feurer, J. T. Springenberg, and F. Hutter, "Initializing bayesian hyperparameter optimization via meta-learning.," in *AAAI*, pp. 1128–1135, 2015.

[42] A. Gupta, Y.-S. Ong, and L. Feng, "Insights on Transfer Optimization: Because Experience is the Best Teacher," *IEEE Trans. on Emerging Topics in Computational Intelligence*, pp. 1–14, November 2017. doi: 10.1109/TETCI.2017.2769104.

[43] M. Ehrgott, "A discussion of scalarization techniques for multiple objective integer programming," *Annals of Operations Research*, vol. 147, no. 1, pp. 343–360, 2006.

[44] H. Ishibuchi, Y. Sakane, N. Tsukamoto, and Y. Nojima, "Adaptation of scalarizing functions in MOEA/D: An adaptive scalarizing function-based multiobjective evolutionary algorithm," in *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 438–452, Springer, 2009.

[45] J. Siwei, C. Zhihua, Z. Jie, and O. Yew Soon, "Multiobjective optimization by decomposition with pareto-adaptive weight vectors," in *Natural Computation (ICNC), 2011 Seventh International Conference on*, vol. 3, pp. 1260–1264, IEEE, 2011.

[46] D. Pardoe and P. Stone, "Boosting for regression transfer," in *Proceedings of the 27th international conference on Machine learning (ICML-10)*, pp. 863–870, 2010.

[47] C. E. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press, 2006.
[48] M. P. Perrone and L. N. Cooper, "When networks disagree: Ensemble methods for hybrid neural networks," tech. rep., BROWN UNIV PROVIDENCE RI INST FOR BRAIN AND NEURAL SYSTEMS, 1992.
[49] D. H. Wolpert, "Stacked generalization," *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.
[50] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, *Scalable test problems for evolutionary multiobjective optimization*. Springer, 2005.
[51] A. Auger, J. Bader, D. Brockhoff, and E. Zitzler, "Hypervolume-based multiobjective optimization: Theoretical foundations and practical implications," *Theoretical Computer Science*, vol. 425, pp. 75–103, 2012.
[52] S. Jiang, Y.-S. Ong, J. Zhang, and L. Feng, "Consistencies and contradictions of performance metrics in multiobjective optimization," *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2391–2404, 2014.
[53] H.-G. Beyer and K. Deb, "Self-adaptive genetic algorithms with simulated binary crossover," tech. rep., Universität Dortmund, 2001.
[54] K. Deb and M. Goyal, "A combined genetic adaptive search (geneas) for engineering design," *Computer Science and Informatics*, vol. 26, pp. 30–45, 1996.
[55] H.-G. Beyer and H.-P. Schwefel, "Evolution strategies–a comprehensive introduction," *Natural computing*, vol. 1, no. 1, pp. 3–52, 2002.
[56] Y. S. Ong, Z. Zhou, and D. Lim, "Curse and blessing of uncertainty in evolutionary algorithm using approximation," in *2006 IEEE International Conference on Evolutionary Computation*, pp. 2928–2935, IEEE, 2006.
[57] W. A. Walbran, *Experimental validation of local and global force simulations for rigid tool liquid composite moulding processes*. PhD thesis, ResearchSpace at Auckland, 2011.
[58] A. Gupta, *Numerical modelling and optimization of non-isothermal, rigid tool liquid composite moulding processes*. PhD thesis, ResearchSpace at Auckland, 2013.
[59] S. Hsu, M. Ehrgott, and P. Kelly, "Optimisation of mould filling parameters of the compression resin transfer moulding process," in *45th annual conference of the operations research society of New Zealand (ORSNZ)*, 2010.
[60] A. Elisseeff, J. Weston, *et al.*, "A kernel method for multi-labelled classification," in *NIPS*, vol. 14, pp. 681–687, 2001.