Spatially distributed statistical significance approach for real parameter tuning with restricted budgets

Adolph J. Vogel[a], Daniel N. Wilke[a,*]

[a] *University of Pretoria, c/o Lynnwood Road and Roper Street, Hatfield, 0002*

Abstract

Parameter tuning aims to find suitable parameter values for heuristic optimisation algorithms that allows for the practical application of such algorithms. Conventional tuning approaches view the tuning problem as two distinct problems, namely, a stochastic problem to quantify the performance of a parameter vector and a deterministic problem for finding improved parameter vectors in the meta-design space. A direct consequence of this viewpoint is that parameter vectors are sampled multiple times to resolve their respective performance uncertainties. In this study we share an alternative viewpoint, which is to consider the tuning problem as a single stochastic problem for which both the spatial location and performance of the optimal parameter vector are uncertain. A direct implication, of this alternative stance, is that every parameter vector is sampled only once. In our proposed approach, the spatial and performance uncertainties of the optimal parameter vector are resolved by the spatial clustering of candidate parameter vectors in the meta-design space. In a series of numerical experiments, considering 16 test problems, we show that our approach, Efficient Sequential Parameter Optimisation (ESPO), outperforms both F/Race and Sequential Parameter Optimisation (SPO), especially for tuning under restricted budgets.

Keywords: Heuristic Algorithms, Response Surfaces, Radial Basis Functions, Sequential Parameter Optimisation, Particle Swarm Optimisation

1. Introduction

Heuristic optimisation algorithms are influenced by parameter values that affect their performance. Finding a set of good parameter values is a challenging task in particular when only restricted budgets of function evaluations can be afforded to tune an optimisation algorithm. The main objective of this paper is to propose a spatially distributed statistical significance approach for real parameter tuning of heuristic optimisation algorithms under a restricted budget.

*Corresponding author
Email address: nico.wilke@up.ac.za (Daniel N. Wilke)

This study is restricted to parameter tuning that aims to find a suitable set of parameter values that remain fixed over the solution of a problem, which is distinct from parameter control that evolves the parameter values as a problem is solved [16]. In particular, we restrict ourselves to real parameter tuning for quantitative parameters of heuristic optimisation algorithms. Although there has been much research in discrete quantitative and qualitative parameter tuning, it is beyond the scope of this paper. Consequently, this study aims to find a vector of real parameter values that results in good performance of a heuristic optimisation algorithm when a limited number of function evaluations can be computed. In particular, tuning under restricted budgets is a recurring problem when solving numerous practical engineering optimisation problems that are multimodal [7, 9, 23, 26, 39, 45]. One such sector is engineering consulting industries that regularly solve new problems using commercial simulation packages. Here, every function evaluation has a direct monetary cost associated with it, in the form of time to solution, software licensing costs to conduct simulations and paying for cloud computing to perform the simulations.

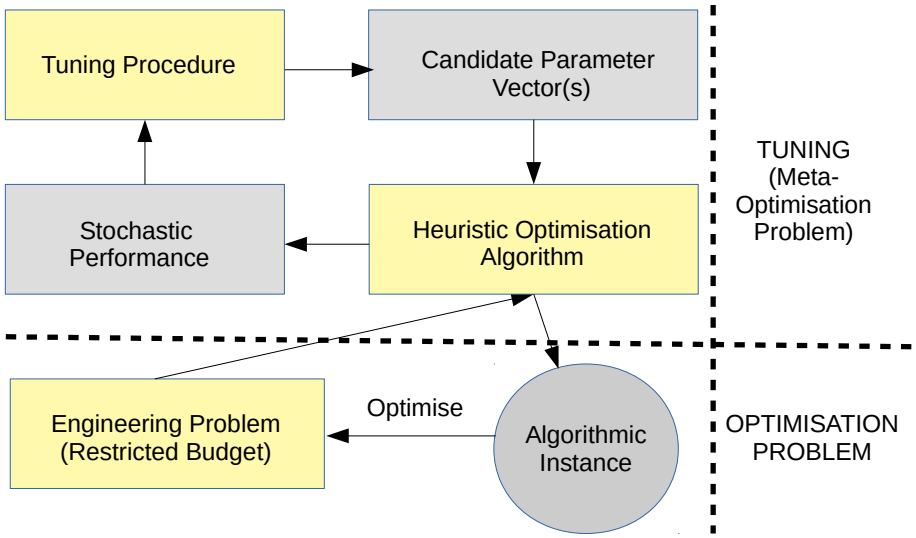


Figure 1: Engineering optimisation problems that first require a tuning problem to be solved.

Consider the application of a heuristic optimisation algorithm to solve a new engineering optimisation problem as depicted in Figure 1. Outlined are two optimisation problems that need to be solved, namely the tuning problem and solving the actual engineering optimisation problem. Firstly, finding an appropriate parameter vector for the heuristic optimisation algorithm that performs effectively and efficiently. This is followed by solving the actual optimisation problem of interest, often a number of times, using the parameter vector obtained from the parameter tuning optimisation problem. The associated com-

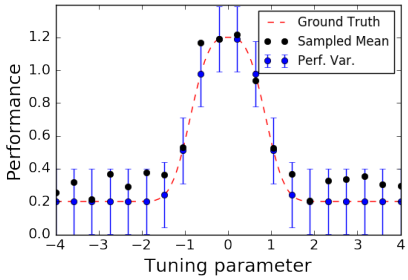
putational cost of solving the tuning problem is well known [17, 33]. Following the terminology and taxonomy outlined for classification of tuning algorithms by Eiben and Smit [17], we restrict this study to iterative tuning methods for which the total search effort can be expressed by

$$\text{Effort} = N_{pv} \times N_{tests} \times N_f, \tag{1}$$

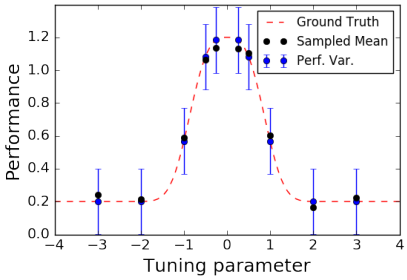
where N_{pv} indicates the number of parameter vectors to be tested, N_{tests} the number of tests per parameter vector to be performed and N_f the number of function evaluations required per run of the heuristic optimisation algorithm. Philosophically, two dominant strategies have evolved that aim to save tuning effort as depicted in Figures 2(a)-(b) [17]. Figure 2(a) represents tuning strategies that were predominantly designed to resolve the performance variance using the least amount of test parameters, N_{tests}. Hence the sampled mean is representative of the actual mean (ground truth) for only the most promising parameter vectors from a large selection of parameter vectors. In turn, Figure 2(b) exemplify tuning strategies that aim to reduce the number of parameter vectors N_{pv} by estimating parameter domains with potentially good candidates and accurately resolving the sampled mean to be representative of the actual mean (ground truth) for each parameter vector.

Generally, tuning strategies that aim to reduce N_{tests} usually start with a large number of candidate parameter vectors each sampled only a few times $N_{tests} \approx 5$ and intelligently add new samples to only promising candidate parameter vectors that can be justified statistically using statistical screening, ranking and selection. The initialisation of these strategies are usually done by design of experiments [21, 31], that aims to uniformly sample the parameter vector space as opposed to uniform random sampling that often leads to clustering of parameter vectors in the meta-parameter space. These include Latin-Hypercube Sampling [21, 31, 25] and Taguchi Orthogonal Arrays [46]. Design of experiment strategies are usually limited to less than ten parameters due to the curse of dimensionality [8]. Tuning strategies following this approach include ANOVA pioneered by Ronald Fisher [20] and statistical racing methods [30] with F/Race [10] often being considered the *de facto* standard in tuning algorithms [2, 11]. As a consequence we select F/Race as one of our benchmarking algorithms.

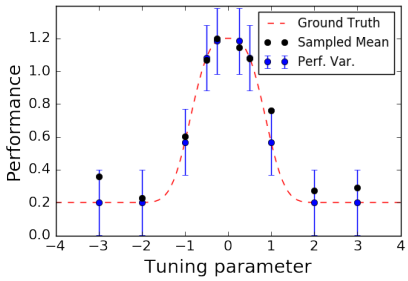
In turn, tuning strategies that aim to reduce N_{pv} start only with a few candidate parameter vectors with the statistical performance properly resolved. Informed by the available information new candidate parameter vectors are added. They include the classical meta-GA [32], meta-EA [48], REVAC [34] and ParamILS [24]. Since, the performance of a parameter vector is stochastic these methods quantify the stochastic nature of each candidate parameter vector by performing a fixed number of tests $N_{tests} >> 5$ that result in statistically significant estimates of performance for each candidate parameter vector. This ensures that only a limited number of parameter vectors are resolved statistically. For restricted tuning budgets, computational improvements to these strategies can be made by reducing both N_{pv} and N_{tests} as depicted in Figure 2(c). Instead of accurately resolving the sampled mean for each parameter



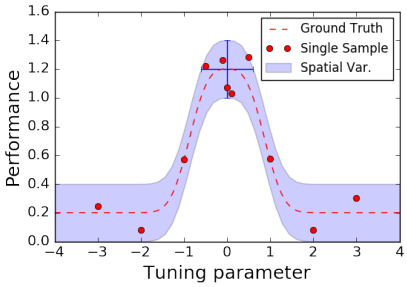
(a) Performance variance significance, strategies to reduce N_{tests}.



(b) Parameter vector significance, strategies to reduce N_{pv}.



(c) Parameter vector and performance vector significance, strategies to reduce both N_{tests} and N_{pv}.



(d) Spatially distributed significance, strategies to reduce both N_{tests} and N_{pv}.

Figure 2: Comparison of parameter tuning paradigms (a)-(d), to resolve computational effort in finding the optimal parameter vector.

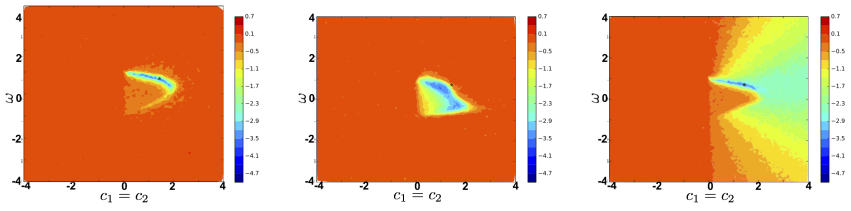
vector, statistical screening, ranking and selection or statistical racing methods are applied to approaches that reduce N_{pv} with the result that both N_{pv} and N_{tests} are reduced.

The number of algorithms that aim to reduce both N_{pv} and N_{tests} are limited with this area of research not receiving significant attention albeit at the reported success of the only four strategies developed in this domain. Two strategies are based on meta-EA with racing [48], while the other two strategies are Sequential Parameter Optimisation (SPO) [4, 3] and REVAC++ [40]. REVAC is a population-based stochastic search method with the ++ indicating that racing and sharpening has been added as strategies to control the number of tests N_{tests} required per parameter vector. SPO in turn starts with an initial population of candidate parameter vectors from which an approximation to the expected performance of the stochastic utility landscape is constructed. This approximation is then used to determine which parameter vectors should be sampled additionally as well as to propose new candidate parameter vectors in regions with promising expected performance. The latter is done by evaluating the approximation for parameter vectors sampled from a design of experiment

in the regions with promising expected performance and retaining the best performing parameter vectors. Strategies that reduce both N_{pv} and N_{tests} are promising tuning strategies when restricted tuning budgets are considered.

Essentially, the approaches discussed so far consider the tuning problem as two distinct problems. A stochastic problem to quantify the performance of a parameter vector and a deterministic problem of finding an improved parameter vector. It is evident that each problem is solved by its own strategy or heuristic, i.e. the stochastic problem is resolved by sampling a candidate parameter vector multiple times, i.e. $N_{tests} >> 1$, following the classical statistical analysis paradigm in parameter tuning, whereas which parameter vectors to consider next are resolved using spatial sampling techniques that may or may not require the minimising of some regression surface using evolutionary algorithms. The associated computational cost of the stochastic problem is usually reduced by utilising statistical screening, ranking and selection or statistical racing methods. Albeit, the associated computational cost of the stochastic problem remains prohibitively expensive when practical engineering simulation based problems are considered for which practitioners need to tune algorithms with a restricted tuning budget, which is typically below 100 meta-function evaluations in total.

We propose to consider the tuning problem as a single stochastic problem, in which, the spatial location of the optimal parameter vector and its performance are uncertain following a statistical mathematical optimisation perspective [18]. Importantly, both uncertainties are simultaneously resolved using the same strategy, namely, the minimisation of a regression surface such that every parameter vector is sampled only once, i.e. $N_{tests} = 1$. The spatial and performance uncertainty of the optimal parameter vector are resolved by the spatial clustering of candidate parameter vectors in meta-design space as depicted in Figure 2(d). Note the two statistical distributions namely the variance in performance of a candidate parameter vector and the variance of candidate parameter vectors around the optimal parameter vector, where each depicted sample is a single sample and not the sampled mean as before. Having multiple samples cluster in the meta-design space allows for both the refinement of the optimal parameter vector and its respective performance utilising candidate parameter vectors that are always only tested once, i.e. $N_{tests} = 1$.



(a) (b) (c)

Figure 3: Comparison of parameter tuning landscapes for three different PSO formulations namely (a) standard PSO, (b) PSO with craziness [22], and (c) PSO with maximum velocity constraint [15].

2. Heuristic Algorithms and Tuning

For engineering applications, it is important that the tuning technique can operate on a restricted budget, as a given set of parameters is only valid for a specific

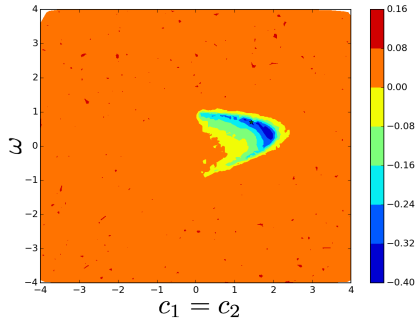
1. choice of heuristics in a heuristic optimisation algorithm,
2. problem or set of problems,
3. budget to solve the problem or set of problems, and
4. choice of qualitative heuristic optimisation parameters.

In this study the objective function that defines the real parameter tuning problem will be referred to as the meta-optimisation fitness (MOF), which is also known as the utility landscape. In turn the real parameter tuning problem will be referred to as the meta-optimisation problem [33], to distinguish it from the engineering optimisation problem that the heuristic optimisation algorithm needs to ultimately solve. The meta-optimisation fitness function is stochastic, hence sampling this function multiple times and averaging the performance is referred to as the expected meta-optimisation fitness (EMOF).

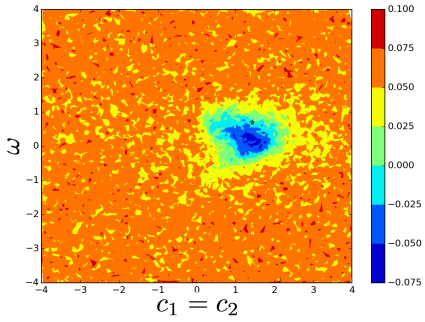
To elaborate, consider the standard Particle Swarm Optimisation (PSO) algorithm [27], which has inertia ω, particle best c_1, global best c_2 attractors, swarm size p and problem budget usually expressed as number of iterations i.e. p function evaluations per iteration. To allow us to visualise the meta-design space, we choose two parameters, namely ω and $c_1 = c_2$ to be varied while a standard test problem set [15] is optimised. The log of the expected meta-optimisation fitness (specifically the Mean Best Fitness [17]) is depicted in Figure 3(a), where the expected meta-optimisation fitness is obtained by averaging 100 meta-optimisation fitness values for each combination of ω and $c_1 = c_2$. In addition, the published parameter vector by Clerc [15] is plotted as a black dot. The non-linear relationship between suitable parameters is evident as well as the localised domain of suitable parameters, with large parts of the parameter domain not delivering suitable parameters. Figures 3(b) and (c) respectively depicts the expected meta-optimisation fitness landscape when the standard PSO is modified to include either craziness [22] or by restricting the maximum velocity [15]. Again, the black dot signifies the published parameter vector of Clerc [15]. It is evident that by including craziness Clerc's optimal parameter set for the standard PSO has been reduced to one of the worst performing parameter vectors in the parameter domain, while it remains near optimal for maximum velocity. Significant changes in domains of suitable parameter vectors in the parameter domain is also evident. Similarly, Figure 4(a)-(d) shows changes in the expected meta-optimisation fitness landscape when the fitness evaluation budget, swarm size, or test problem is changed.

2.1. Formal real parameter tuning problem

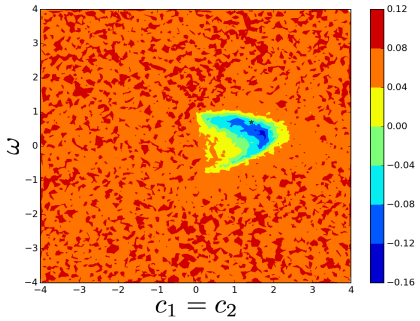
Formally, the stochastic meta-optimisation problem for real parameter tuning requires some scalar MOF function computed from the performance on a set of objective functions $f_i(\vec{x}), \quad 1, \ldots, k$, with $\vec{x}$ feasible in the hypercube defined



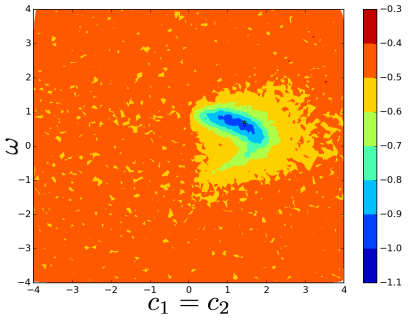
(a) Rastrigin, 10 000 evaluations, 50 Particles



(b) Rastrigin, 500 evaluations, 50 Particles



(c) Rastrigin, 10 000 evaluations, 15 Particles



(d) Griewank, 10 000 evaluations, 50 Particles

Figure 4: Comparison of fitness landscapes under different conditions.

by the lower $\vec{x}_l$ and upper bounds $\vec{x}_u$, for a single meta-optimisation run of the heuristic optimisation *Algorithm*:

$$\phi\left(\vec{p}\right) \leftarrow \left[\text{Algorithm}(\vec{p}) : \min_{\vec{x}} f_i\left(\vec{x}\right) \text{ s.t. } \vec{x} \in [\vec{x}_{l,i}, \vec{x}_{u,i}], \;\; i = 1, \ldots, k\right], \quad (2)$$

that depends on parameters $\vec{p}$. As pointed out, the stochastic nature of the meta-optimisation problem is usually resolved using one strategy, while a different strategy is employed to find suitable parameter vectors. The stochastic nature of the meta-optimisation problem is usually resolved by computing an expected meta-optimisation fitness (EMOF)

$$\min_{\vec{p}} \Phi\left(\vec{p}\right) = \sum_{k=1}^{l} \phi_k\left(\vec{p}\right), \quad (3)$$

which is generally the averaged meta-optimisation fitness computed l times, with $l >> 1$, for a single parameter vector. To find an improved or the optimal parameter vector, $\vec{p}^*$, the EMOF is minimised using a search based strategy that may be based on meta-GA [32], Revac [34] or ParamILS [24]. Numerous proposals have been made to consolidate multiple MOF runs for a given parameter vector into an EMOF. The three primary scalarisations listed by Eiben and Smit [17] are

- *Mean Best Fitness (MBF)* for a fixed number of evaluations, we determine the average best fitness value over a number of optimisation runs;
- *Average Evaluations to Solution (AES)* where we record the number of function evaluations required to reach a previously known solution; and
- *Success Rate (SR)* the percentage of runs to reach a previously known solution within a given evaluation budget.

As highlighted before to compute the EMOF a fixed set of parameter vectors is evaluated a number of times. The problem with this approach is that while the statistical significance is properly accounted for a parameter vector, the spatial resolution of parameter vectors are usually crudely approximated on some predetermined grid as with F/Race and statistical exploratory analysis (SEA). Sequential parameter optimisation (SPO) [5] aims to address both uncertainties in some way, by utilising EMOF on an adaptively sampled grid. First, EMOF is crudely approximated at a few design of experiment parameter vectors. A regression function is then constructed that approximates the EMOF at unknown parameters vectors using radial basis functions [19]. The approximated EMOF is then computed for a denser design of experiment sampling of the meta-design space. Promising parameter vectors are retained and the EMOF improved by conducting additional sampling of the selected parameter vectors. The regression allows for estimating a unique value at a parameter vector when multiple tests for that parameter vector has been computed. SPO requires multiple tests

to be computed for promising candidate parameter vectors, but this can be prohibitively expensive when restricted tuning budgets are considered.

This study addresses this issue by consolidating the uncertainties in both i) the performance of a parameter vector, as well as, ii) the expected optimal parameter vector. This is done by extending SPO to use an alternative to EMOF, namely *Statistical Meta-Optimisation Fitness (SMOF)*. SMOF accounts for the statistical variability of a heuristic *Algorithm* by never computing the MOF more than once for a parameter vector. It rather resolves the uncertainty in performance as well as the spatial uncertainty of the optimal parameter vector in the meta-design domain by having parameter vectors for which MOFs have been computed cluster in the meta-design domain. The more points clustered in a part of the meta-design domain the better both uncertainties are resolved.

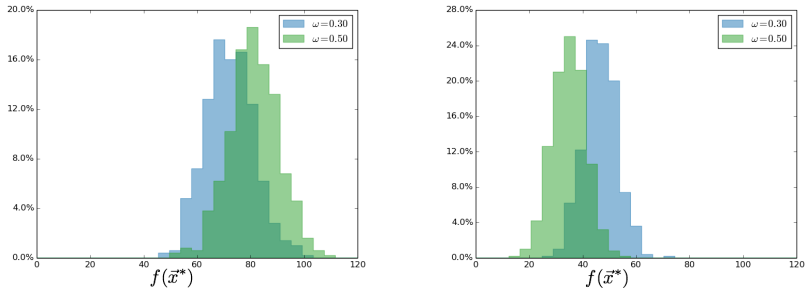
Formally, it follows from the *Central Limit Theorem* that increasing the sample size from a distribution with a *finite* level of variance, the mean of the samples will be approximately equal to the mean of the distribution. Since, both the variance in performance and the spatial variance around the optimal parameter vector is finite, it follows that the mean of the samples performance will be approximately equal to the mean of the actual performance, while the mean of the samples for the spatial location of the optimal parameter vector will be approximately equal to the mean of the actual optimal parameter vector. This proves statistically that the strategy is convergent in the limit of large sample sizes should appropriate numerical strategies be employed to realise this approach.

2.2. Set of Test Problems and Meta-Optimisation Fitness

The set of test problems used to tune an optimisation algorithm significantly influences the performance of a heuristic algorithm. This is due to a parameter vector that performs well on one set of problems will not necessarily perform well on another set of problems. This is illustrated in Figures 5(a) and (b), where a standard PSO solved two global test problems, Alpine and Griewank [15], using $c_1 = c_2 = 1.43$, while ω is chosen as 0.5 and 0.3. Figures 5(a) and (b) shows the histogram of the computed objective function value after repeating the optimisation process 100 times. It is clear that $\omega = 0.5$ performs better on the Griewank problem, while $\omega = 0.3$ performs better on the Alpine problem.

This is analogous to the well known *No Free Lunch* statement in optimisation [47], that no parameter vector outperforms all other parameter vectors on all test problems. As demonstrated it is possible to find a parameter vector that performs exceptionally well on a specific problem or class of problems, which is often referred to as specialist heuristic algorithm settings [41], as they are efficient for the right problem but not robust should the problem or problem class change. On the other hand, generalist heuristic algorithm settings [41], are parameter vectors that work reasonably well on a number of problems or classes of problems.

In this study we consider two problem sets. Our initial investigation is conducted on a set of six problems [15] that include both unimodal and multimodal



(a) Alpine (b) Griewank

Figure 5: Comparison of PSO performance using two parameter vectors, 1) $w = 0.3$ and $c_1 = c_2 = 1.43$ and 2) $w = 0.5$ and $c_1 = c_2 = 1.43$, on (a) the Alpine and (b) Griewank test problems.

Table 1: Set of six test problems ($f_1 - f_6$) [15], used to conduct our initial investigation in this study.

Problem	**Formulation**	**Bounds** $[\vec{x}_{l,i}, \vec{x}_{u,i}]$
Parabola	$f_1(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$	$[-100, 100]$
Rosenbrock	$f_2(\mathbf{x}) = \sum \left((1 - x_i)^2 + 100(x_i^2 - x_{i+1})^2\right)$	$[-10, 10]$
Ackley	$f_3(\mathbf{x}) = -20e^{\left(-0.2\sqrt{\frac{\sum_{i=1}^{n} x_i^2}{n}}\right)} - e^{\left(\frac{\sum_{i=1}^{n} cos(2\pi x_i)}{n}\right)} + 20 + e$	$[-30, 30]$
Alpine	$f_4(\mathbf{x}) = \sum_{i=1}^{n} \lvert x_i \sin x_i + 0.1x_i \rvert$	$[-10, 10]$
Griewank	$f_5(\mathbf{x}) = \sum_{i=1}^{n} \frac{(x_i - 100)^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i - 100}{\sqrt{n}}\right)$	$[-300, 300]$
Rastrigin	$f_6(\mathbf{x}) = \sum_{i=1}^{n} x_i^2 - 10 \cos 2\pi x_i + 10$	$[-5.12, 5.12]$

Table 2: Set of ten test problems ($f_7 - f_{16}$) to verify the reported results in this study on an independent set of problems to conclude this study.

Problem	**Formulation**	**Bounds** $[\vec{x}_{l,i}, \vec{x}_{u,i}]$
Levy	$f_7(\mathbf{x}) = \sin^2(\pi w_1) + \sum_{i=1}^{n-1}(w_i - 1)^2$ $\left[1 + 10\sin^2(\pi w_i + 1)\right](w_d - 1)^2\left[1 + \sin^2(2\pi w_n)\right]$, $w_i = 1 + \frac{x_i - 1}{4},\ i = 1, \ldots, n$	$[-10, 10]$
Power Sum	$f_8(\mathbf{x}) = \sum_{i=1}^{d}\left(\left(\sum_{j=1}^{d} x_j^i\right) - b_i\right)^2$	$[0, d]$
Zakharov	$f_9(\mathbf{x}) = \sum_{i=1}^{d} x_i^2 + \left(\sum_{i=1}^{d} 0.5 i x_i\right)^2$ $+ \left(\sum_{i=1}^{d} 0.5 i x_i\right)^4$	$[-5, 10]$
Michalewicz	$f_{10}(\mathbf{x}) = -\sum_{i=1}^{d} \sin(x_i)\sin^{20}\left(\frac{i x_i^2}{\pi}\right)$	$[0, \pi]$
Sum Squares	$f_{11}(\mathbf{x}) = \sum_{i=1}^{d} i x_i^2$	$[-10, 10]$
Sum Different Powers	$f_{12}(\mathbf{x}) = \sum_{i=1}^{d} \lvert x_i\rvert^{i+1}$	$[-1, 1]$
Rotated Hyper-Ellipsoid	$f_{13}(\mathbf{x}) = \sum_{i=1}^{d}\sum_{j=1}^{i} x_j^2$	$[-65.536, 65.536]$
Perm	$f_{14}(\mathbf{x}) = \sum_{i=1}^{d}\left(\sum_{j=1}^{d}(j + 0.5)(x_j^i - \frac{1}{j^i})\right)^2$	$[-d, d]$
Styblinski-Tang	$f_{15}(\mathbf{x}) = \frac{1}{2}\sum_{i=1}^{d}\left(x_i^4 - 16x_i^2 + 5x_i\right)$	$[-5, 5]$
Quartic Noise	$f_{16}(\mathbf{x}) = \sum_{i=1}^{d}\left(x_i^4\right) + U([0, 1])$	$[-1.28, 1.28]$

test problems as tabulated in Table 1. This initial set of problems is indicated by the vector valued function,

$$\vec{F}(\vec{x}) = \{f_1(\vec{x}), f_2(\vec{x}), \ldots, f_6(\vec{x})\}. \tag{4}$$

This multi-objective optimisation problem can be solved by finding the Pareto front of non-dominated solutions. This approach is however limited to a low number of functions as the curse of dimensionality requires a large number of evaluations before the actual Pareto front becomes evident. For higher dimensional multi-objective problems most of the solutions computed would be non-dominated due to the sparse sampling of the Pareto front in higher dimensions. In this study we scalarise the multi-objective optimisation problem. Instead of following the weighted-sum approach

$$F(x) = \frac{1}{N}\sum_{i=1}^{N} w_i f_i(\vec{x}), \tag{5}$$

with unit weights $w_i = 1$ as motivated by Pedersen [35], we normalise the different test problems such to allow us to better understand and interpret the results. We therefore scale the problems to be indicative of the improvement from where the heuristic optimisation algorithm started. To ensure we have resolution during both the initial and final phases of the optimisers we consider the orders of improvement from where the heuristic optimisation algorithm started as stated by

$$\phi\left(\vec{p}\right) = \frac{1}{k}\sum_{i=1}^{k} \log_{10} \frac{f_i^*}{f_i^{\{0\},}}, \tag{6}$$

with f_i^* obtained by

$$\min_{\vec{x}} f_i\left(\vec{x}\right) \text{ s.t. } \vec{x} \in \left[\vec{x}_{l,i}, \vec{x}_{u,i}\right], \quad i = 1, \ldots, k, \tag{7}$$

using Algorithm($\vec{p}$), while $f_i^{\{0\}}$ is the best value after the swarm is initialised at the beginning of the optimisation run. (6) therefore represents the average *orders of improvement* over the initial value of the test problem. This implies that an algorithm that reduces the result from 10^3 to 10^2 carries the same weight as an improvement from 10^1 to 10^0. To highlight the initial improvement the $\log_{10}$ can be omitted but to not distract from the focus of our study we choose (6) as our MOF.

3. Proposed Restricted Budget Tuning Technique

Our proposed approach is an extension of Sequential Parameter Optimisation (SPO) [3], we therefore first give a brief outline of SPO in Section 3.1. Thereafter we present our Efficient Sequential Parameter Optimisation (ESPO) strategy to resolve the tuning problem for restricted tuning budgets in Section 3.2. Both SPO and ESPO are model based techniques that construct

surrogate models of EMOF (i.e. multiple evaluations of MOF, averaged) and MOF (i.e. single evaluation of MOF), respectively. This is done to improve the efficiency of the tuning process as the models can be evaluated instead of the actual EMOF or MOF.

SPO specifically relies on approximating EMOF and the meta-design space by first defining new sampled locations of parameter vectors and then evaluating the approximation of the EMOF from which parameter vectors are chosen for which the EMOF is resolved. SPO constructs regression based surrogate models for which the number of parameter vectors at which EMOF have been resolved is usually low, but the number of tests conducted per parameter vector is high. By averaging the multiple tests per parameter vector to estimate the EMOF, an interpolation based surface can be constructed that directly estimates the EMOF.

On the other hand, ESPO constructs a response surface by using only single MOF evaluations per parameter vector. The surrogate model is then optimised to obtain potentially new locations for improved parameter vectors for which only single MOF evaluations are conducted. As the sample locations cluster in sample space the accuracy of both the optimal parameter vector values and the expected performance for these parameter vectors are resolved without having to have sampled the same parameter vector more than once. ESPO constructs regression based surrogate models since the number of parameter vectors for which the MOF has been computed is usually high in comparison to the number of regression coefficients that needs to be estimated.

3.1. Sequential Parameter Optimisation (SPO)

SPO proposed by Bartz-Beielstein et al. [3] efficiently finds the optimal parameter vector for an heuristic algorithm by intelligently increasing both the number of candidate parameter vectors as well as the number of fitness evaluations of MOF to resolve the EMOF per parameter vector per iteration. The points added during each iteration is determined by constructing a model through the existing points and using these points to predict new candidate parameter vectors. SPO essentially follows the following three steps:

1. sample the EMOF,
2. interpolation or regression based approximation to EMOF, and
3. add new candidate parameter vectors.

These three steps are repeated until the utility budget is depleted. Let's consider each step in more detail as implemented in this study.

3.1.1. SPO - Step 1: Sample the EMOF

Initially, the meta-design space is sampled at n different candidate parameter vectors using latin-hypercube sampling (LHS) [31], with each candidate parameter vector sampled $m_0 = 2$ times to estimate EMOF. During subsequent iterations only the best candidate parameter vector is retained and sampled together with the new candidate parameter vectors. The number of samples per candidate parameter vector is doubled during each iteration, that is, $m_{k+1} \leftarrow 2 \times m_k$.

As the iterations progress the sampled EMOF converges to the true EMOF. The number of samples for the best candidate parameter vector from the previous iteration is also doubled. This ensures a fair comparison between the sampled points.

3.1.2. SPO - Step 2: Regression Based Approximation to EMOF

All the candidate parameter vectors at the current iteration is used to approximate the EMOF using a regression surrogate model. Multiple tests have been conducted for each candidate parameter vector, which now requires a regression surrogate surface to estimate the EMOF.

Although, numerous regression approximation approaches are available we choose radial basis functions (RBF). This allows for a consistent comparison between SPO and ESPO.

3.1.3. SPO - Step 3: Add New Candidate Parameter Vectors

The regression surrogate model is then evaluated on a denser LHS grid of parameter vectors to identify promising candidate parameter vectors. The number of samples on the regression surrogate model is typically significantly more than the number of current parameter vectors. The parameter vectors deemed the most promising are then retained together with the best parameter vector found so far.

3.2. Efficient Sequential Parameter Optimisation (ESPO)

SPO has two limitations when it comes to parameter tuning for restricted evaluation budgets, both of which we address with our proposed ESPO.

Firstly, SPO relies on the repeated re-evaluation of individual parameter vectors to obtain a statistically significant EMOF. The EMOF is then approximated using a regression based surrogate model to resolve the multiple tests per parameter vector. Sampling EMOF until it is statistically significant for numerous parameter vectors could potentially waste large portions of the tuning budget on suboptimal parameter vectors as already pointed out. Our proposed ESPO instead constructs a regression based surrogate model [28] through spatially distinct parameter vectors that are always sampled only once by evaluating the MOF once-off per parameter vector. This allows us to simultaneously resolve the uncertainty in the response as well as the spatial uncertainty of the optimal parameter vector in the domain through denser spatial sampling as opposed to higher sampling of individual candidates. We demonstrate in Section 3.2.2 that regression surfaces compensates for the stochastic nature of MOF to estimate the EMOF from individually sampled parameter vectors for which only MOF is computed. Addressing this first limitation, of sampling the same parameter vector multiple times, forms the primary contribution of this study.

Secondly, SPO samples the constructed utility model using a denser LHS. This still leaves finding the best predicted candidate parameter vector to chance, which primarily depends on the number sample parameter vectors in the denser LHS. The number of sample points could be increased, but Bartz-Beielstein

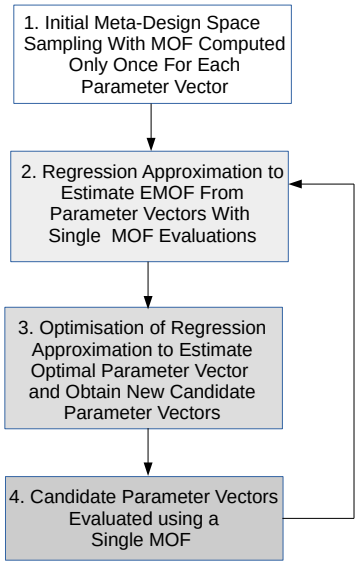


Figure 6: Outline of efficient sequential parameter optimisation (ESPO).

et al. [3] admits that this wastes the tuning budget since we may add candidate solution points closely clustered together. If these clusters are far from the actual optimum, which will happen during the initial searches, these clusters help little in finding the actual optimum solution. In fact, Bartz-Beielstein and Preuss [5] used an order of magnitude more iterations during their optimisation process to find parameter settings than we use in our approach. In ESPO we merely optimise the regression surrogate surface to obtain candidate parameter vectors. These candidate parameter vectors are then evaluated by evaluating the MOF only once per parameter vector.

The outline of ESPO is depicted in Figure 6 as four steps with sufficient detail to distinguish it from the similar SPO steps. Let's consider each step in more detail as implemented in this study.

3.2.1. ESPO - Step 1: Sample the Meta-Design Space

Initially, the meta-design space is sampled at m different candidate parameter vectors using LHS [31]. This is achieved by first sampling the design domain using, m, points between the lower $\vec{p}_l$ and upper bounds $\vec{p}_u$,

$$LHS(m, \vec{p}_l, \vec{p}_u) \rightarrow \vec{p}_i,\ i, \ldots, m. \tag{8}$$

Each j^{th} candidate parameter vector, $\vec{p}_j$, is then only sampled once by computing its corresponding MOF, $\phi_j = \phi(\vec{p}_j)$, by evaluating MOF only once for each parameter vector

$$\vec{\Phi} = \{\phi_1, \phi_2, \ldots, \phi_m\}. \tag{9}$$

3.2.2. ESPO - Step 2: Construct Statistical Meta-Optimisation Fitness (SMOF)

Radial basis functions (RBFs) can be used to construct interpolation or regression based approximations to scalar functions by computing a linear combination of n non-linear radial basis functions to satisfy m equations. Each radial basis function has the same functional form but is centred around different parameter vectors $\vec{c}_j,\ j = 1,\ldots,n$ in the meta-design space, which is mathematically expressed by

$$\tilde{\phi}\left(\vec{p}\right) = \sum_{j=1}^{n} \alpha_j \lambda\left(||\vec{p} - \vec{c}_j||\right). \tag{10}$$

In this study a Gaussian basis function [38], $\lambda\left(\vec{p}\right) = e^{-s||\vec{p}-\vec{c}_j||^2}$ is chosen due to its smooth characteristics, although it is susceptible to ill-conditioning when s is poorly chosen and as a consequence requires a strategy to resolve s. Here α_j is the associated scalar value or weight in the linear combination to approximate some function. Hence, α_j first needs to be solved from a system of equations that aims to approximate the desired function. Given $m = n$ unique equations we can uniquely solve for n weights, $\alpha_j,\quad j = 1,\ldots,n$. However, should we have more unique equations to satisfy $m > n$ than the number of weights n available, we can only approximate the desired function. We can evaluate the design domain at m locations and enforce that the RBF recover the computed responses at the m locations with the least error using $n < m$ basis functions. In this study, we randomly select the n centre points $\vec{c}_j$ of the RBF basis functions using

$$LHS(n, \vec{p}_l, \vec{p}_u) \rightarrow \vec{c}_j,\ j = 1,\ldots,n \tag{11}$$

which avoids clustering of basis functions in the meta-design domain. As a result the following system of equations

$$\phi_1 = \tilde{\phi}\left(\vec{p}_1\right) = \sum_{j=1}^{n} \alpha_j \lambda\left(||\vec{p}_1 - \vec{c}_j||\right) \tag{12}$$

$$\phi_2 = \tilde{\phi}\left(\vec{p}_2\right) = \sum_{j=1}^{n} \alpha_j \lambda\left(||\vec{p}_2 - \vec{c}_j||\right) \tag{13}$$

$$\vdots \quad \vdots \quad \vdots \tag{14}$$

$$\phi_m = \tilde{\phi}\left(\vec{p}_m\right) = \sum_{j=1}^{n} \alpha_j \lambda\left(||\vec{p}_m - \vec{c}_j||\right) \tag{15}$$

can be rewritten ((12) - (15)) into matrix form, using the convention $\lambda_{ij} = \lambda\left(||\vec{p}_i - \vec{c}_j||\right)$, to obtain

$$\begin{bmatrix} \lambda_{11} & \cdots & \lambda_{1n} \\ \vdots & \ddots & \vdots \\ \lambda_{m1} & \cdots & \lambda_{mn} \end{bmatrix} \begin{Bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{Bmatrix} = \begin{Bmatrix} \phi_1 \\ \vdots \\ \phi_m \end{Bmatrix} \text{ or } \Lambda\vec{\alpha} = \vec{\phi}. \tag{16}$$

Since $n < m$, the residual $R(\vec{\alpha}) = \Lambda\vec{\alpha} - \vec{\phi}$ expresses the difference between the approximation $\Lambda\vec{\alpha}$ and the $\vec{\phi}$ values to be recovered. Finding $\vec{\alpha}$ to reduce the difference can be solved as an optimisation problem that minimises the sum of the errors squared given by

$$\min_{\vec{\alpha}} E(\vec{\alpha}) = \min_{\vec{\alpha}} \left(\Lambda\vec{\alpha} - \vec{\Phi}\right)^T \left(\Lambda\vec{\alpha} - \vec{\Phi}\right). \tag{17}$$

The optimality criterion for (17) obtained by direct differentiation of $E(\vec{\alpha})$ w.r.t. $\vec{\alpha}$ gives

$$\nabla E(\vec{\alpha}) = \vec{0} = 2\Lambda^{\mathrm{T}}\Lambda\vec{\alpha} - 2\Lambda^{\mathrm{T}}\vec{\Phi}, \tag{18}$$

which leads to the least squares solution for overdetermined linear systems [44],

$$\Lambda^T\Lambda\vec{\alpha} = \Lambda^T\vec{\phi}. \tag{19}$$

Consequently the difference between the approximation $\Lambda\vec{\alpha}$ and the computed MOF responses $\vec{\phi}$, is minimised. This is depicted in Figure 7 which depicts both interpolation and regression based RBF approximations in conjunction with the actual EMOF (averaged over 100 runs). The interpolation based RBF (green) and regression based RBF (red) are constructed using only single MOF samples at 21 locations. The mean of the function (blue) and 95% variance (blue halo) are also depicted. To compute the MOF the test problems listed in Table 1 are optimised using the PSO heuristic optimisation algorithm with $\omega = 0.5$ and the parameters $c_1 = c_2$ varying between 0 and 2.5. For the regression RBF only 10 equally spaced centres across the domain were chosen. In both cases the range parameter s was taken as the average distance between the data points.

From Figure 7, it is clear that regression that minimizes the sum of the error squared, acts as a filter against stochastic variation when compared to interpolation for the same number of points. We can see that the interpolation line deviates from the mean line in order to go through the sample points. On the other hand the regression line tends to follow the general trend of the mean line. We quantify the error of the interpolation RBF versus regression RBF by computing the root mean square error (RMSE) from the actual mean for both approximations. The interpolation RBF has a 45% larger error than the regression RBF using exactly the same data. To improve the accuracy of the interpolation RBF we would have needed to evaluate the same points multiple times to compute the expected (mean) value, which is essentially what SPO does to resolve the expected response but on average this would require significantly more data points than regression RBF to obtain the same accuracy of the actual mean estimate.

In addition, the above observation is supplemented by investigating to what extent the regression approximation is representative of the actual EMOF. We start our investigation by assuming that a linear regression model [37] is able to describe the EMOF of the actual stochastic spatial distribution of the MOF under consideration i.e. an instance of MOF over the entire meta-design domain of interest is assumed to be expressed by

$$\vec{\phi} = \Lambda\vec{\alpha}^* + \vec{u}, \;\; \vec{u} \sim IID(\vec{0}, \sigma^2\mathbf{I}), \tag{20}$$

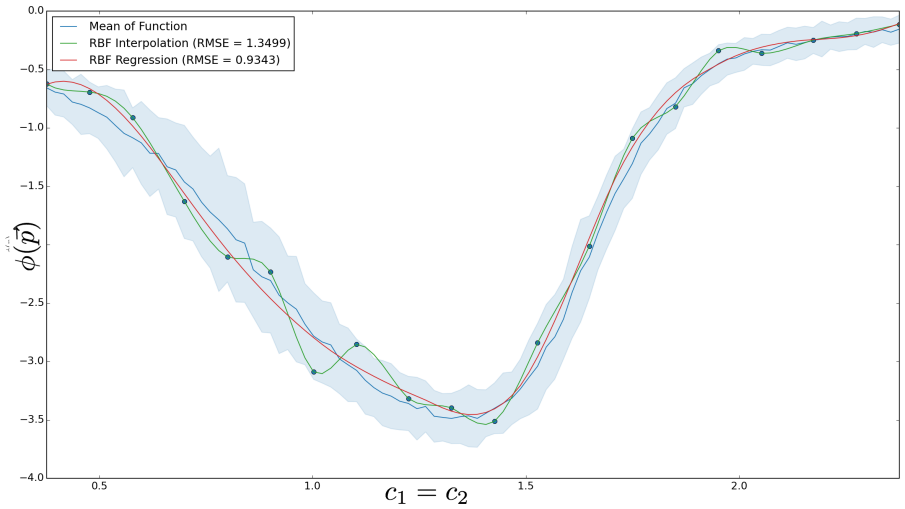


Figure 7: Interpolation RBF (green) and regression RBF (red) fitted through twenty one spatially distinct evaluations of a random function with mean (blue) and 95% variance (blue halo). The Root Mean Square Error (RMSE) for both RBFs from the actual mean is indicated in the legend by sampling 10 points in the domain.

where $IID(\vec{0}, \sigma^2\mathbf{I})$ signifies that each instance of $\vec{u}$ is independently and identically distributed given some variance σ^2. Here, $\vec{\alpha}^*$ signifies the actual linear estimator that is assumed to be able to exactly reproduce the EMOF. The least squares estimator is given by $\vec{\alpha} = (\Lambda^T\Lambda)^{-1}\Lambda^T\vec{\phi}$, which can be evaluated using our assumed model for the actual responses $\Lambda\vec{\alpha}^* + \vec{u}$ to obtain

$$\vec{\alpha} = (\Lambda^T\Lambda)^{-1}\Lambda^T\left(\Lambda\vec{\alpha}^* + \vec{u}\right) = \vec{\alpha}^* + (\Lambda^T\Lambda)^{-1}\Lambda^T\vec{u}. \tag{21}$$

The expected value $E(\cdot)$,

$$E(\vec{\alpha}) = E(\vec{\alpha}^*) + E\left((\Lambda^T\Lambda)^{-1}\Lambda^T\vec{u}\right), \tag{22}$$

reduces to

$$\vec{\alpha} = \vec{\alpha}^*, \tag{23}$$

under the condition that $E\left((\Lambda^T\Lambda)^{-1}\Lambda^T\vec{u}\right) = (\Lambda^T\Lambda)^{-1}\Lambda^T E\left(\vec{u}\right) \to E\left(\vec{u}\right) = \vec{0}$. This condition holds for the uncertainty regarding the response for a parameter vector per definition of EMOF. Secondly, by computing instances of MOF clustered in the domain per definition implies that EMOF is resolved in the domain of the clustered points. The larger the number of points in the cluster the better EMOF is resolved. The only assumption that does not strictly hold is that each instance of $\vec{u}$ is identically distributed. This is evident from Figure 7, but again the spatial variation in σ^2 is usually within the same order in the domains of interest which approximately satisfies this assumption. We refer to this approach as *Statistical Meta-Optimisation Fitness (SMOF)*.

The time complexity to compute SMOF, once the MOF for each parameter vector has been evaluated, is dictated by the solution of $\vec{\alpha} = (\Lambda^T\Lambda)^{-1}\Lambda^T\vec{\phi}$. Following the Bachmann-Landau notation [1], the time complexity to solve $\vec{\alpha}$ for a system of m equations and n regression coefficients are as follows:

1. $\mathcal{O}(n^2m)$ to compute $\Lambda^T\Lambda$,
2. $\mathcal{O}(nm)$ to compute $\Lambda^T\vec{\phi}$,
3. $\mathcal{O}(n^3)$ to solve the linear system $(\Lambda^T\Lambda)^{-1}\vec{\alpha} = \Lambda^T\vec{\phi}$ using Cholesky factorisation.

Asymptotically $\mathcal{O}(n^2m)$ dominates $\mathcal{O}(n^3)$ since we are dealing with a regression problem, i.e. $n < m$. Consequently the total asymptotic time complexity to compute SMOF is $\mathcal{O}(n^2m)$. In addition, the error rate for non-parametric regression surfaces constructed with a sample size m is proportional to $m^{\frac{-2}{2+k}}$, where k is the number of parameters i.e. $\vec{p} \in \mathcal{R}^k$ [43]. This leads to exponential time complexity of m as k increases when aiming to construct a regression surface with similar accuracy. Fortunately, the number of parameters k for a heuristic optimisation algorithm is usually limited or can be reduced using dimensional reduction or feature extraction techniques [13], that aims to capture the data without redundancy e.g. maximum relevance and minimum common redundancy (N-MRMCR-MI) [14] or the tolerance rank-variance formulation (TRVF) [12]. In addition, evaluating m parameter vectors is an embarrassingly parallel computational problem.

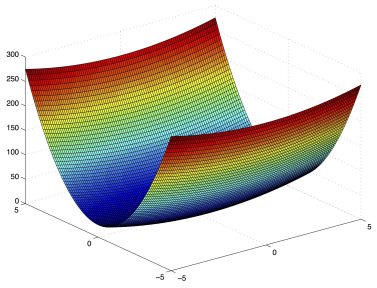
3.2.3. Basis Functions and Shape Parameters

A secondary benefit of regression RBF is that numerical ill-conditioning is delayed in particular when the susceptible Gaussian radial basis function is selected, requiring s to be resolved with due care.

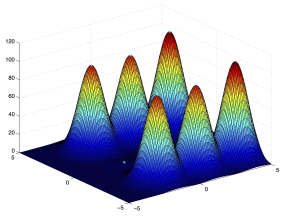
For example, consider the Gaussian basis function evaluated at a unit distance $||\vec{p}-\vec{c}_j|| = 1$, which gives e^{-s}. For large s a large negative value exponent is evaluated which is close to zero. In turn, for a small s the exponent of a number close to 0 is evaluated, which is close to 1. However, making s too small results in ill-conditioning as the linear system represented by (16) becomes singular. This is elegantly illustrated by the following example.

Consider the quadratic surface, $f(\mathbf{x}) = x_1^2 + 10x_2^2$, depicted in Figure 8(a). This is approximated by sampling this function at nine points on a 3×3 grid. An interpolation based RBF surface using Gaussian basis functions is constructed from the nine sampled points using three selected shape parameters, namely $s = 1, 0.1$ and 5×10^{-5} and depicted in Figures 8(b)-(d). The ill-conditioning resulting from choosing s too small is evident.

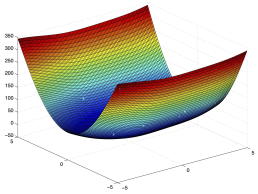
Usually an appropriate shape parameter is computed by minimising the well-known *Leave One Out Cross Validation Error (LOOCVE)* as a function of s [38]. This is illustrated in Figure 9 for our example problem, which shows the error function is insensitive in the region of the actual optimum. Instead of minimising LOOCVE we confirmed in all our tests that the actual optimum was close to the average distance between the different points making up the RBF



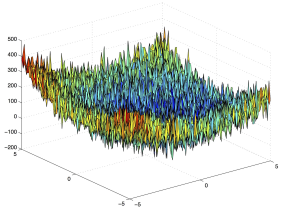
(a) Actual quadratic surface



(b) $s = 1 \times 10^{0}$



(c) $s = 1 \times 10^{-1}$



(d) $s = 5 \times 10^{-5}$

Figure 8: Effect of the shape parameter s when approximating the (a) actual quadratic surface using interpolation based RBF with Gaussian basis functions.

response surface. We exploit this observation in this study by always resolving s as the average distance between the centre points. However, we suggest that the LOOCVE be formally minimised using the average distance between centres as a starting guess for the optimisation problem to resolve s.

3.3. ESPO - Step 3: Add SMOF Optimum as Candidate Parameter Vector

The global best parameter vector is estimated by finding the optimal SMOF, which is then included in the new construction of a regression response surface using a new LHS set of RBF centres as a fraction of the total number of points, including the newly added point.

In this study, the SMOF model is optimised using the limited memory version of the BFGS Quasi-Newton method with bound constraints (L-BFGS-B) [29, 49]. The multi-start strategy allows for multi-modality in the SMOF model to be handled. The model is optimised 25 times from uniform randomly chosen starting points in the meta-design domain in addition to including the previous global best as a starting point. The best solution over all the runs is recorded. Fortunately, analytical gradients for the regression RBF surface can be easily computed making a multi-start gradient based strategy available as an efficient optimisation approach.

This elementary strategy can easily be improved using a more robust multi-start strategy. One such an example is the Bayesian Stopping criteria proposed by Snyman and Fatti [42]. They proposed that the optimisation runs be repeated from different starting bounds until the global minimum is found within a given

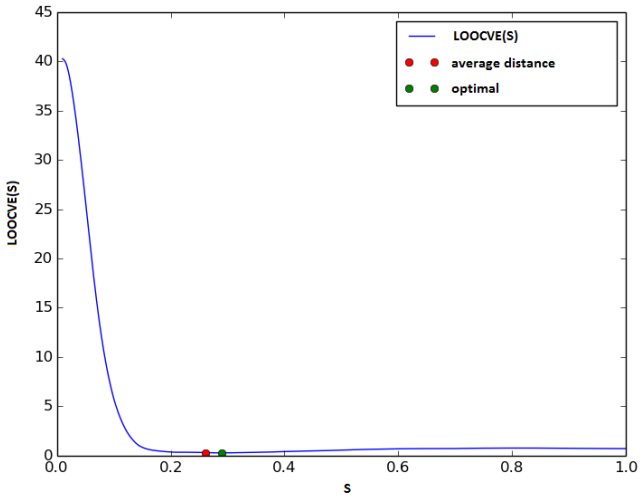


Figure 9: Optimal (green) and averaged distance estimated (red) shape parameter s.

level of confidence. However, the freedom of choosing an optimisation strategy ultimately lies with the user as long as the estimated global optimum parameter vector is robustly resolved.

3.4. Pseudocode for ESPO

The pseudocode for ESPO is listed under Algorithm 1. The function ESPO requires the initial number of LHS designs m, parameter vector upper and lower bounds ($\vec{p}_l$ and $\vec{p}_u$), fraction of centers of the total number of parameter vectors ($nfraction$), number of optimisation restarts ($kmax$) and the total function evaluation budget ($budget$) to be specified. This is in addition to the availability of a function to compute latin-hypercube samples LHS and a selected MOF $\phi(\vec{p})$.

4. Numerical Results

The numerical results in this study are presented in three sections. Firstly, a sensitivity study of the parameters of ESPO is conducted to find robust settings for these parameters. This is followed by a detailed investigation on the performance of F/Race, SPO and ESPO on the set of six test problems, where after the study is concluded by confirming the results on an independent set of ten test problems.

For this study, we include the SPO strategy as a benchmark strategy for two reasons. Firstly, SPO has proved to be an efficient and competitive tuning strategy [6]. Secondly, our proposed approach extends on the concepts on which SPO is based. As outlined, our proposed strategy also constructs approximations of the utility function but with the difference that the approximation

Algorithm 1 Efficient Sequential Parameter Optimisation (ESPO)

function ESPO($m, \vec{p}_l, \vec{p}_u, nfraction, kmax, budget$)
Initialise:
  $\vec{p}_1, \ldots, \vec{p}_m \leftarrow LHS(m, \vec{p}_l, \vec{p}_u)$ sampled parameter vectors between $\vec{p}_l$ and $\vec{p}_u$
  $\vec{\Phi} = \{\phi_1, \phi_2, \ldots, \phi_m\} \leftarrow \phi_j = \phi(\vec{p}_j),\ j = 1, \ldots, m$
Construct SMOF:
  while $m < budget$ **do**
    $n \leftarrow int(nfraction \times m)$.
    $\vec{c}_j,\ j = 1, \ldots, n \leftarrow LHS(n, \vec{p}_l, \vec{p}_u)$ sampled centres for RBF
    $\Lambda_{ij} \leftarrow \lambda(\|\vec{p}_i - \vec{c}_j\|),\ i = 1, \ldots, m;\ j = 1, \ldots, n$
    $\vec{\phi} = \{\phi_1, \phi_2, \ldots, \phi_m\} \leftarrow \phi_j = \phi(\vec{p}_j),\ j = 1, \ldots, m$
    $\vec{\alpha} \leftarrow (\Lambda^{\mathrm{T}}\Lambda)^{-1}\Lambda^{\mathrm{T}}\vec{\phi}$
    $\tilde{\phi}(\vec{x}) \leftarrow \sum_{j=1}^{n} \alpha_j \lambda(\|\vec{x} - \vec{c}_j\|)$
Optimise SMOF:
    $k \leftarrow 0$
    $\phi_{\mathrm{best}} \leftarrow \min(\vec{\Phi})$
    $\vec{x}_{\mathrm{best}} \leftarrow phi(\vec{x}_{\mathrm{best}}) = \min(\vec{\Phi})$
    while $k < kmax$ **do**:
      if $k = 1$ **then** $\vec{x}_0 = x_{\mathrm{best}}$
      else $\vec{x}_0 = rand(\vec{p}_l, \vec{p}_u)$
      end if
      Find $\vec{x}^* = \min_{\vec{x}} \tilde{\phi}(\vec{x})$ using $\vec{x}_0$ as starting point
      if $\tilde{\phi}(\vec{x}^*) < \phi_{\mathrm{best}}$ **then**
        $\phi_{\mathrm{best}} = \tilde{\phi}(\vec{x})$
        $\vec{x}_{\mathrm{best}} = \vec{x}^*$
      end if
      $k \leftarrow k + 1$.
    end while
    $\vec{p}_{m+1} \leftarrow \vec{x}_{\mathrm{best}}$
    $m \leftarrow m + 1$
  end while
return $\vec{p}_m$

estimates both the performance and spatial location of the candidate optimal parameter vector using parameter vectors that were tested only once.

4.1. ESPO Parameter Sensitivity Study

Before we proceed, it is clear that ESPO has its own set of parameters that needs to be selected. They are the following three parameters:

1. R_s - the fraction of the total meta-evaluation budget to evaluate for the initial regression RBF surface,
2. R_c - the fraction of centres of the total number of points,
3. R_r - the ratio of the shape parameter to the average distance between points.

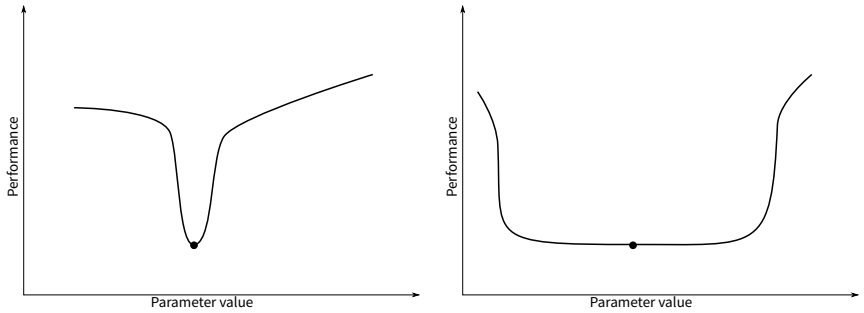
The number of centres $n^{\{k\}}$ at iteration k for the regression RBF is some chosen as some fraction R_c of the total number of MOF evaluations $m^{\{k\}}$, which can be expressed as $n^{\{k\}} = R_c \times m^{\{k\}}$ with $R_c \in (0, 1] \bigcap \mathbf{R}$. The centres, $\vec{c}_j$, are determined by computing a Latin Hyper Cube sample over the defined meta-design domain which avoids spatial clustering of the centres that may otherwise lead to ill-conditioning.

This necessarily raises the question "Are we better off, aren't we just replacing one set of parameters for another?". However, there are two scenarios that would leave us better tuning ESPO than trying to directly tune the heuristic optimisation algorithm.

Firstly, we are better off when ESPO has less parameters than what needs to be tuned for a typical heuristic optimisation algorithm. Secondly, we are better off if the ESPO parameters are less sensitive than the parameters of a typical heuristic optimisation algorithm. To illustrate the later observation regarding the difference between a sensitive and robust parameter, consider Figures 10(a) and (b). Therefore to ascertain the robustness of the three parameters of ESPO we conduct a sensitivity analysis, which demonstrates that the ESPO parameters are found to be robust. We are therefore able to propose appropriate values for each of the three parameters.

This is done by performing a coarse 3^3 full factorial experiment, i.e. three values per parameter for the three ESPO parameters as tabulated in Table 3. It is evident that the parameters values in the full factorial experiment differ significantly allowing for a proper sensitivity versus robustness study to be performed. The performance of each point on the full factorial grid of ESPO parameters was sampled a 100 times to get an accurate estimate of the EMOF and MOF variance of the estimated optimal PSO parameters. For the PSO we estimated w and $c_1 = c_2$, while the problem dimension for the problems listed in Table 1 was chosen to be 30 and the particle swarm size 50.

Figure 11 shows the results of the coarse parameter search for the ESPO tuning method with the ESPO parameter vector given in the form $\{R_c, R_s, R_r\}$. From it we see that all the ESPO parameters yielded PSO parameters that lead to a mean MOF between -3 and -3.5. This shows that in all cases the ESPO found parameters that lead the PSO algorithm to make on average three orders



(a) (b)

Figure 10: Illustration of the difference between (a) a sensitive and (b) robust parameter value.

Table 3: Parameter values tested for *ESPO*.

Parameter	**Value 1**	**Value 2**	**Value 3**
R_s	0.10	0.20	0.40
R_c	0.25	0.50	0.75
R_r	0.50	1.00	2.00

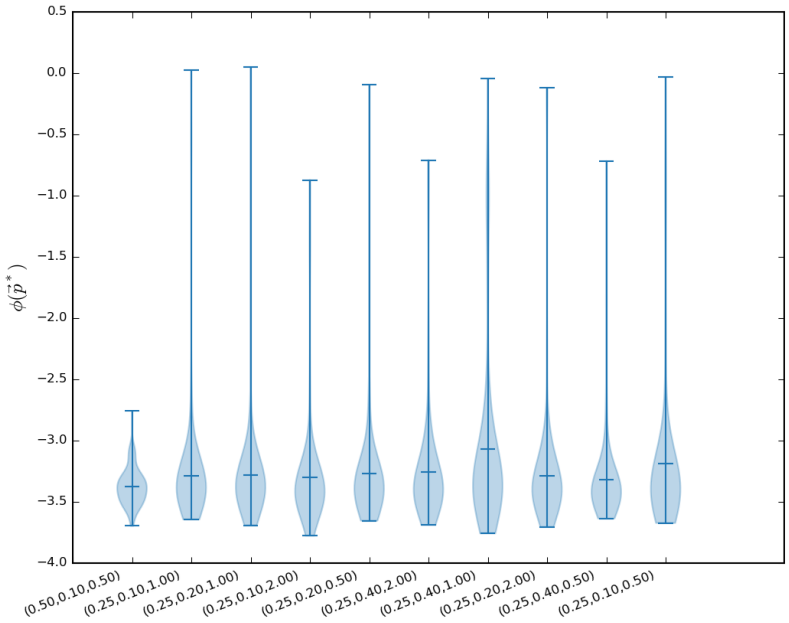


Figure 11: Violin-plots of the mean MOF, averaged over 100 ESPO runs, for the 10 parameter vectors with the most variance of 27 computed ESPO parameter vectors. The ESPO parameter vectors are given in the form $\{R_c, R_s, R_r\}$.

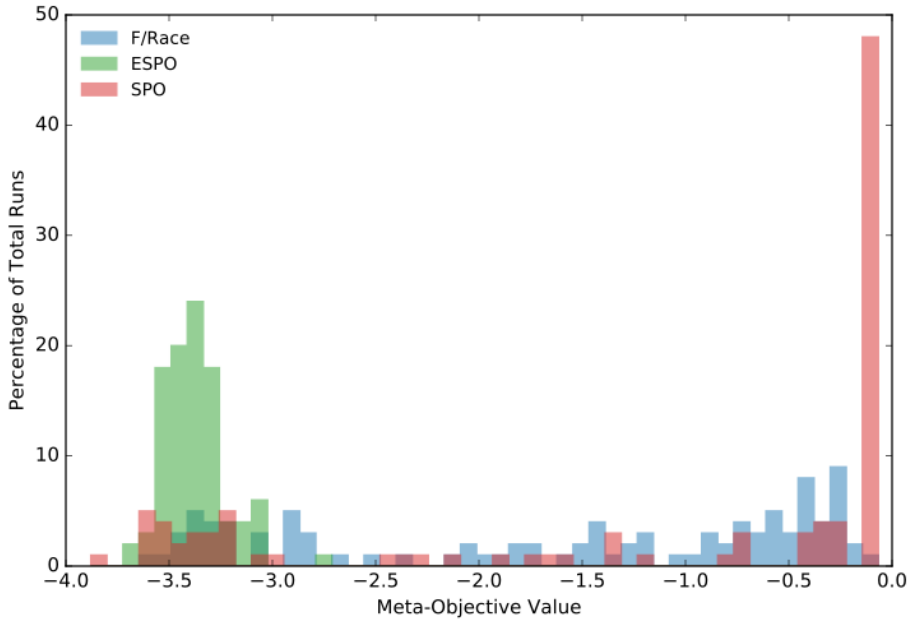


Figure 12: Comparison of the MOF (log scale) estimated 100 times for F/Race, SPO and ESPO on the six test problems listed in Table 1.

of magnitude improvement on the individual test problems. The fact that all the violin-plots in Figure 11 are more or less on the same level shows that the ESPO algorithm is fairly insensitive to its own parameters. The violin plot with parameters $R_c = 0.5$, $R_s = 0.1$, and $R_r = 0.5$ performed marginally better and had the lowest variance and thus we will use it for the remainder of our study.

4.2. Performance Comparison on Six Test Problems

In addition to SPO and ESPO we also consider arguably the *de facto* standard tuning algorithm namely F/Race [2], as a standard benchmark. To compare the performance of F/Race, SPO and ESPO we tune the Particle Swarm Optimisation algorithm on the six test problems listed in Table 1. We now deliberately choose the problem dimension and number of particles in the PSO distinct from the values chosen in the sensitivity study in Section 4.1. The problem dimension is now selected as 15 and following Pedersen and Chipperfield [36] we select the swarm size 30.

For all three tuning techniques we restrict the meta-evaluation budget to 100 MOF evaluations, while the total fitness budget for the PSO algorithm is 5000. Consequently, to tune the algorithm using one approach therefore requires a total of 500 000 function evaluations to evaluate the six test problems.

Figure 12 shows the results obtained from the multiple tuning runs on a logarithmic scale. We see that F/Race and SPO produced parameters that had anything from 1 to 3 orders improvement on average on the set of six test problems. On the other hand, ESPO consistently produced parameters that reached at least three orders improvement on the test problem set.

The PSO parameters obtained by SPO had the greatest variability in the as depicted in Figures 13(a) and (b), while ESPO had the least variability. This is most likely because SPO evaluates the intermediate models using a LHS instead of optimising the RBF response surface. By optimising the RBF regression surface we see that ESPO is able to resolve PSO parameters more consistently. In ESPO we optimise the model using deterministic gradient optimisation algorithms, so we are confident that at least we find a local minima of the model. Through the multi-start strategy employed the probability of finding the actual global minimum of the model is also improved. It is evident that the PSO is by far more sensitive to ω than the c_{12} parameter. All three tuning methods found ω parameter vectors clustered around 0.7 with SPO and F/Race more dispersed than ESPO. Interestingly, these ESPO values are clustered around the published values by Clerc [15].

To interrogate the quality of the estimated optimal PSO parameters by ESPO specifically, we performed an exhaustive line search at one of the obtained solutions along both parameter axes in Figure 14. Each line search consisted of 100 different points, where each point sampled the MOF a 100 times. We then plotted the EMOF value along the line and also showed the 95% confidence interval for the values. The optimal parameter vectors found by ESPO is shown on the Figure as a blue dot. This clearly illustrates the estimated optimal parameter vectors are at the basin of at least a local minimum as both parameters are at the minimum of a local basin.

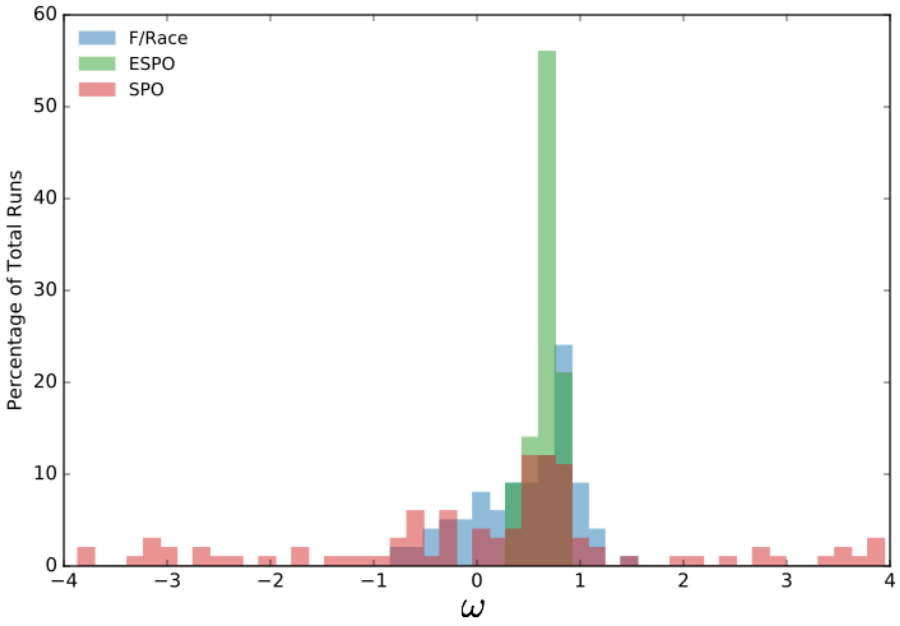
Finally, to investigate the dependency between the two parameters in Figures 17(a), 17(b) and Figure 14 we plotted all the points the tuning techniques found on a contour plot of EMOF in Figure 15. This shows a banana shaped valley, similar to the two-dimensional Rosenbrock function, in which a domain of better performing parameter vectors lie. All three tuning methods found parameter vectors close to and along this valley. However, the ESPO points are more localised within the valley. Looking at the EMOF in Figure 15, we have confidence that this is in fact globally optimal within a statistical margin.

4.3. Performance Comparison on Ten Independent Test Problems

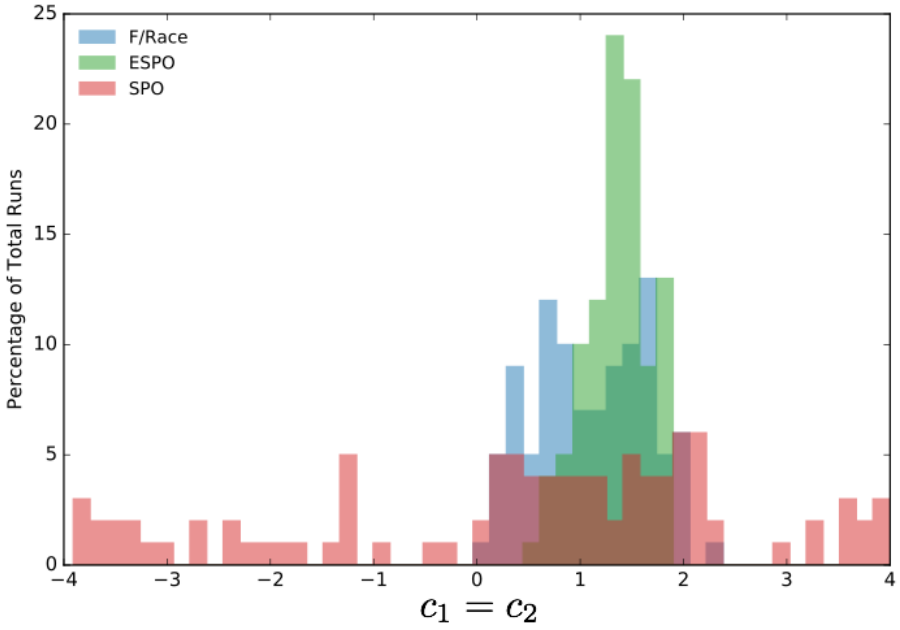
To ensure that the observed response can be generalised we consider all three tuning methods on ten different test problems listed in Table 2. The problem dimension and particle size is still chosen as 15 and 30 respectively, while the meta-optimisation budget and function evaluation budget is chosen as before.

Figure 16 shows the results obtained from the multiple tuning runs on a logarithmic scale. We see that the bulk of ESPO performance is between 7 and 8 orders better than F/Race and SPO.

The PSO parameters obtained by ESPO had the least variance as depicted in Figures 17(a) and (b), while F/Race and SPO had higher variance. Although



(a)



(b)

Figure 13: Distribution of the (a) ω and (b) $c_1 = c_2$ PSO parameters obtained by 100 runs of the different tuning techniques on the six test problems listed in Table 1.

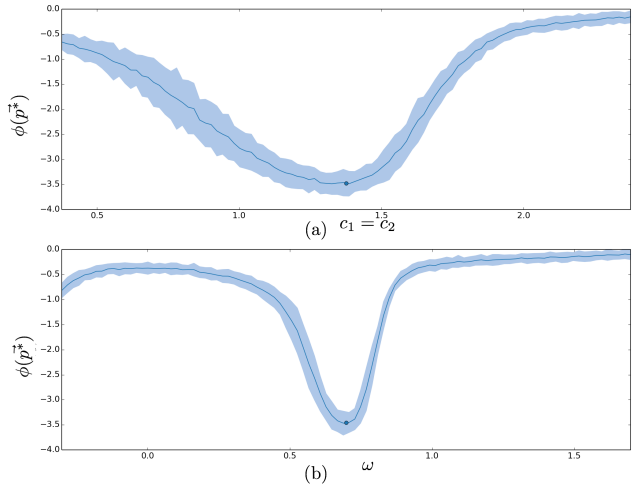


Figure 14: Line-searches around the optimal point parameter vector during meta-optimisation along a random search direction.

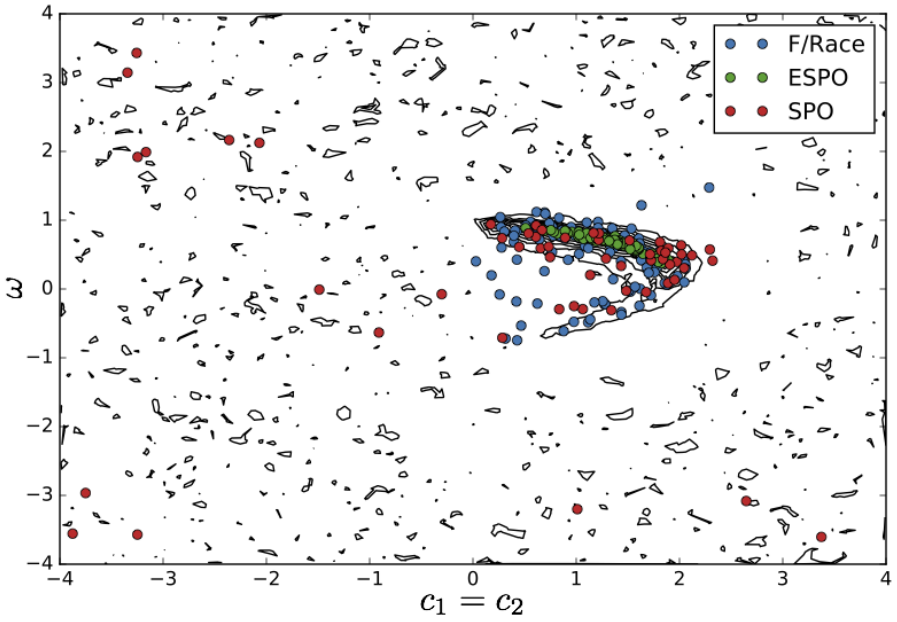


Figure 15: Spatial distribution of the optimal PSO parameters found by the different tuning methods.

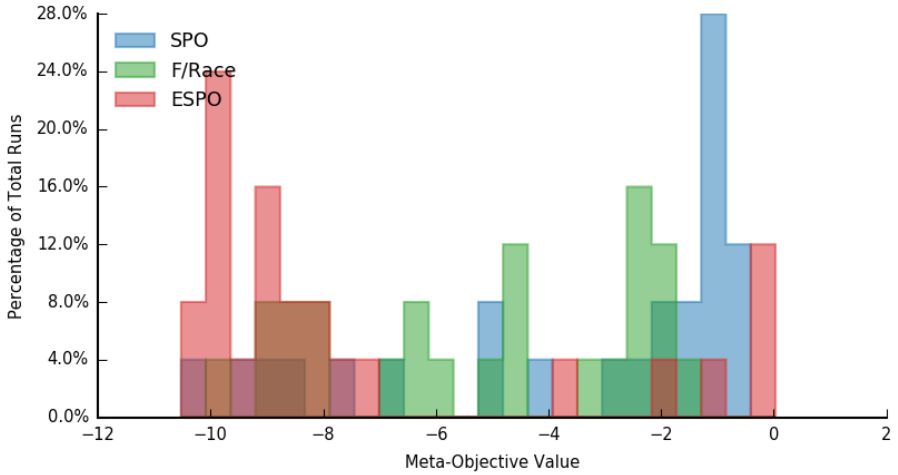


Figure 16: Comparison of the MOF (log scale) estimated 100 times for F/Race, SPO and ESPO on the ten test problems listed in Table 2.

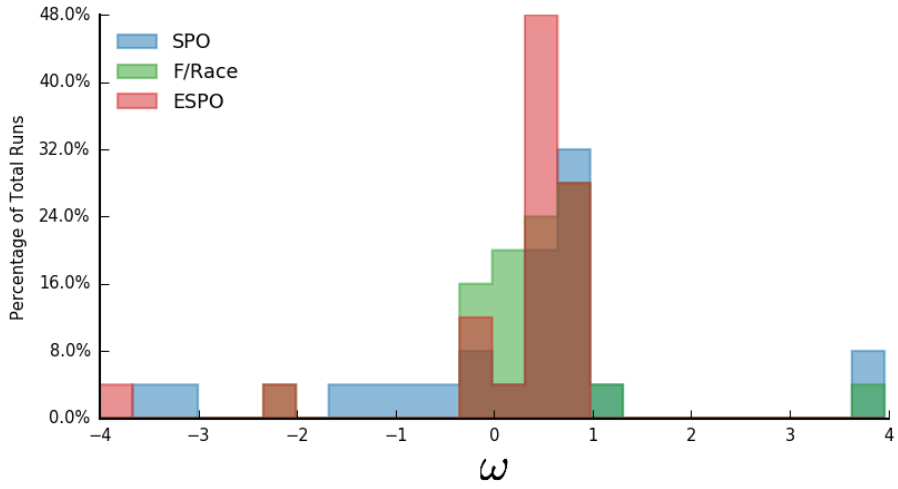
the PSO algorithm was tuned for ten different test problems the same conclusion as before is drawn, that is ESPO performs orders better than the closest rival which is F/Race.

5. Conclusion

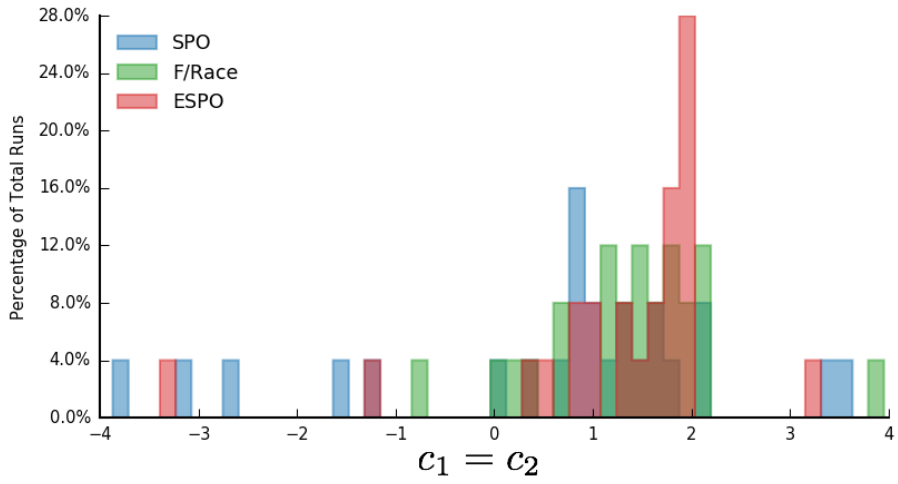
Real parameter tuning algorithms require individual parameter vectors to be sampled multiple times, i.e. $N_{tests} >> 1$, to estimate the expected performance of the respective parameter vectors required to compute the Expected Meta-Optimisation Fitness (EMOF). Statistical screening, ranking and selection or statistical racing methods are mostly used to reduce the associated cost to compute EMOF, but unfortunately, the cost remains prohibitively expensive when practical engineering problems are considered subject to restricted meta-evaluation budgets well below 100 evaluations in total.

This study proposed an alternative approach to reduce the computational cost by never sampling an individual parameter vector more than once, i.e. $N_{tests} = 1$. This is achieved by resolving the spatial location of the optimal parameter vector in the meta-design domain and its associated performance by spatial clustering of parameter vectors the meta-design domain. This is achieved by fitting a regression based response surface through the parameter vectors with each vector sampled only once. The regression surface is then optimised to estimate the spatial location of the optimal parameter vector and its mean performance. We refer to this fitness function as *Statistical Meta-Optimisation Fitness (SMOF)*.

A regression RBF response surface approach is proposed, namely Efficient Sequential Parameter Optimisation (ESPO), as an effective model based tuning



(a)



(b)

Figure 17: Distribution of the (a) ω and (b) $c_1 = c_2$ PSO parameters obtained by 100 runs of the different tuning techniques on the ten test problems listed in Table 2.

approach to construct and minimise SMOF under restricted tuning budgets. We showed that the ESPO parameters are insensitive and therefore fairly robust by conducting a sensitivity study in addition to considering its performance on two distinct test problem sets. We considered a moderate fitness budget of 5 000 function evaluations per meta-function evaluation, and a meta-evaluation budget of 100 evaluations in total. This means that just to tune a stochastic optimisation algorithm we require 500 000 function evaluations. If each function evaluation were to take 60 seconds, then the entire tuning process would take just short of a year on a single core computer. Clearly this becomes prohibitively expensive. If you want to tune anything besides academic test problems then you must do so very efficiently with a restricted meta-evaluation budget.

We compared ESPO to both F/Race and SPO on two sets of test problems ranging from 15 to 30 dimensional problems. On both test sets, ESPO, found superior performing parameters to F/Race and SPO using the same meta-evaluation budget. A line search, depicted in Figure 14, showed these parameters are indeed locally optimal, which was then verified to be globally optimal. Finally, it found a parameter vector that outperformed the values published by Clerc (2006). It seems that ESPO should be the tuning method of choice for low utility budgets. In addition, ESPO should not be viewed as a competitor to SPO but as an improvement to the concepts introduced by SPO that makes it more efficient in restricted tuning budget scenarios. This is due to two favourable properties of the method. Firstly, it only evaluates a parameter vector once and then uses regression to get an accurate model of the expected utility landscape. This means that it uses much fewer points to build a regression model than SPO, which evaluates parameter vectors multiple times. Secondly, ESPO uses formal optimisation techniques to optimise the regression model instead of following SPO's strategy to sample the regression model using design of experiments. This accelerates the quality of the solutions obtained from the regression model.

Additional research is required to asses the robustness of the ESPO parameters on practical problems. In addition to considering heuristic optimisation algorithms with more parameters, as well as parameters that are defined over different length scales. It would be of significant practical benefit to include discrete parameters.

Lastly, a reminder that tuning is not always practical. Even with low utility budgets tuning is still a significant computational cost. When practitioners are constantly optimising unique problems once-off it makes little sense to perform tuning before optimising the problem directly. There is always a balance to be kept between optimising with sub-optimal parameters versus tuning and then optimising with "optimal" parameters. Tuning really comes to its own in situations where similar problems have to be solved multiple times. For example, constantly solving vehicle routing problems for different scenarios. Here, a tuned algorithm can save valuable time by conducting a tuning study once-off for a set of representative scenarios and then using the optimal parameters to solve new routing scenarios. Here, again having to compute each parameter vector only once during the tuning process may save significant resources in contrast

to repeatedly sampling individual parameter vectors.

References

[1] Bachmann, P., 1894. Die Analytische Zahlentheorie. B. G. Teubner Verlag, Leipzig.

[2] Balaprakash, P., Birattari, M., Stützle, T., 2007. Improvement Strategies for the F-Race Algorithm: Sampling Design and Iterative Refinement. In: Bartz-Beielstein, T., Aguilera, M. J. B., Blum, C., Naujoks, B., Roli, A., Rudolph, G., Sampels, M. (Eds.), Hybrid Metaheuristics. Lecture Notes in Computer Science. Springer, pp. 108–122.

[3] Bartz-Beielstein, T., Lasarczyk, C., Preuss, M., 2005. Sequential parameter optimization. In: IEEE Congress on Evolutionary Computation. Vol. 1. IEEE, pp. 773–780.

[4] Bartz-Beielstein, T., Parsopoulos, K. E., Vrahatis, M. N., 2004. Design and Analysis of Optimization Algorithms Using Computational Statistics. Applied Numerical Analysis & Computational Mathematics 1, 413–433.

[5] Bartz-Beielstein, T., Preuss, M., 2006. Considerations of budget allocation for sequential parameter optimization (SPO). In: Workshop on Empirical Methods for the Analysis of Algorithms, Proceedings. pp. 35–40.

[6] Bartz-Beielstein, T., Preuß, M., 2014. Experimental analysis of optimization algorithms: Tuning and beyond. In: Theory and Principled Methods for the Design of Metaheuristics. Springer, pp. 205–245.

[7] Basudhar, A., Missoum, S., May 2013. Reliability-Based Design Optimization using Efficient Global Reliability Analysis. In: 10th World Congress on Structural and Multidisciplinary Optimization. pp. 1–15.

[8] Bellman, R., 1957. Dynamic programming. Princeton University Press.

[9] Bichon, B., Mahadevan, S., Eldred, M., May 2009. Reliability-Based Design Optimization using Efficient Global Reliability Analysis. In: 50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference. AIAA, pp. 1–12.

[10] Birattari, M., Stützle, T., Paquete, L., Varrentrapp, K., 2002. A Racing Algorithm for Configuring Metaheuristics. In: Langdon, W. B., Cantú-Paz, E., Mathias, K. E., Roy, R., Davis, D., Poli, R., Balakrishnan, K., Honavar, V., Rudolph, G., Wegener, J., Bull, L., Potter, M. A., Schultz, A. C., Miller, J. F., Burke, E. K., Jonoska, N. (Eds.), GECCO. Morgan Kaufmann, pp. 11–18.

[11] Birattari, M., Yuan, Z., Balaprakash, P., Stützle, T., 2010. F-race and iterated F-race: An overview. In: Experimental methods for the analysis of optimization. Springer, pp. 311–336.

[12] Chae, Y., Wilke, D. N., 2017. Heuristic linear algebraic rank-variance formulation and solution approach for efficient sensor placement. Engineering Structures 153 (Supplement C), 717–731.

[13] Chakraborty, D., Pal, N. R., 2008. Selecting useful groups of features in a connectionist framework. IEEE Transactions on Neural Networks 19 (3), 381–396.

[14] Che, J., Yang, Y., Li, L., Bai, X., Zhang, S., Deng, C., 2017. Maximum relevance minimum common redundancy feature selection for nonlinear data. Information Sciences 409-410 (Supplement C), 68–86.

[15] Clerc, M., 2006. Particle Swarm Optimization. iSTE.

[16] Eiben, A., Hinterding, R., Michalewicz, Z., 1999. Parameter control in evolutionary algorithms. IEEE Transactions on Evolutionary Computation 3 (2), 124–141.

[17] Eiben, A. E., Smit, S. K., 2011. Parameter tuning for configuring and analyzing evolutionary algorithms. Swarm and Evolutionary Computation 1 (1), 19–31.

[18] Evensen, G., 2007. Statistical optimization. Springer, pp. 139–155.

[19] Fasshauer, G. E., Zhang, J. G., 2007. On choosing "optimal" shape parameters for RBF approximation. Numerical Algorithms 45, 345–368.

[20] Fisher, R., 1925. Statistical Methods for Research Workers. Oliver & Boyd.

[21] Fisher, R., 1937. The Design of Experiments. Oliver & Boyd.

[22] Ho, S., Yang, S., Ni, G., Wong, K., 2006. An Improved Particle Swarm Optimization Method with Application to Multimodal Functions of Inverse Problems. In: Electromagnetic Field Computation, 2006 12th Biennial IEEE Conference on. pp. 125–125.

[23] Hu, Z., Du, X., 2015. Mixed efficient global optimization for time-dependent reliability analysis. Journal of Mechanical Design 137, 051401–051401–9.

[24] Hutter, F., Hoof, H., Leyton-Brown, K., Stützle, T., 2009. ParamILS: An Automatic Algorithm Configuration Framework. Journal of Artificial Intelligence Research 36, 267–306.

[25] Iman, R. L., 2014. Latin Hypercube Sampling. John Wiley & Sons, Ltd.

[26] Jones, D., Schonlau, M., Welch, W., 1998. Efficient global optimization of expensive black-box functions. Journal of Global Optimization 13, 455–492.

[27] Kennedy, J., Eberhart, R. C., November 1995. Particle Swarm Optimization. In: Proceedings of IEEE International Conference on Neural Networks. Vol. 4. IEEE Computer Society, Washington, DC, USA, pp. 1942–1948.

[28] Khuri, A. I., Mukhopadhyay, S., 2010. Response Surface Methodology. WIREs Computational Statistics 2, 128–149.

[29] Liu, D. C., Nocedal, J., 1989. On the limited memory BFGS method for large scale optimization. Mathematical programming 45 (1), 503–528.

[30] Maron, O., Moore, A. W., 1997. The racing algorithm: Model selection for lazy learners. Artificial Intelligence Review 11 (1), 193–225.

[31] McKay, M. D., Beckman, R. J., Conover, W. J., 1979. Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics 21 (2), 239–245.

[32] Montero, E., Riff, M.-C., Neveu, B., 1986. Optimization of control parameters for genetic algorithms. IEEE Transactions on Systems, Man and Cybernetics 16, 122–128.

[33] Montero, E., Riff, M.-C., Neveu, B., 2014. A beginner's guide to tuning methods. Applied Soft Computing 17, 39–51.

[34] Nannen, V., Eiben, A., 2006. A method for parameter calibration and relevance estimation in evolutionary algorithms. In: Proceedings of the 8th annual conference on Genetic and evolutionary computation (GECCO'06). pp. 183–190.

[35] Pedersen, M. E. H., 2010. Tuning & Simplifying Heuristical Optimization. Ph.D. thesis, University of Southampton.

[36] Pedersen, M. E. H., Chipperfield, A. J., 2010. Simplifying Particle Swarm Optimization. Applied Soft Computing 10 (2), 618–628.

[37] Rice, J., 2007. Mathematical Statistics And Data Analysis, 3rd Edition. Duxbury Advanced Series. Thomson/Brooks/Cole.

[38] Rippa, S., 1999. An algorithm for selecting a good value for the parameter c in radial basis function interpolation. Advances in Computational Mathematics 11, 193–210.

[39] Shan, S., Wang, G., 2006. Failure surface frontier for reliability assessment on expensive performance function. Journal of Mechanical Design 128, 1227–1235.

[40] Smit, S., Eiben, A., September 2009. Comparing parameter tuning methods for evolutionary algorithms. In: IEEE Congress on Evolutionary Computation. IEEE Press, pp. 399–406.

[41] Smit, S., Eiben, A., 2010. Parameter tuning of evolutionary algorithms: generalist vs. specialist. In: Proceedings of the 2010 international conference on Applications of Evolutionary Computation (EvoApplicatons'10). pp. 542–551.

[42] Snyman, J., Fatti, L., 1987. A multi-start global minimization algorithm with dynamic search trajectories. Journal of Optimization Theory and Applications 54 (1), 121–141.

[43] Stone, C., 1982. Optimal global rates of convergence for non-parametric estimators. Annals of Statistics 10, 1340–1353.

[44] Strang, G., 1993. Introduction to linear algebra, 2nd Edition. Wellesley-Cambridge Press Wellesley, MA.

[45] Streuber, G., 2017. A gradient-based multistart algorithm for multimodal aerodynamic shape optimization problems based on free-form deformation. Master's thesis, University of Toronto.

[46] Taguchi, G., Yokoyama, Y., 1993. Taguchi methods: design of experiments. Taguchi Methods Series. ASI Press.

[47] Wolpert, D. H., Macready, W. G., 1997. No free lunch theorems for optimization. Evolutionary Computation, IEEE Transactions on 1 (1), 67–82.

[48] Yuan, B., Gallagher, M., 2007. Combining meta-EAs and racing for difficult EA parameter tuning tasks. In: Lobo, F., Lima, F., Michalewicz, C. (Eds.), Parameter Setting in Evolutionary Algorithms. Springer, pp. 121–142.

[49] Zhu, C., Byrd, R. H., Lu, P., Nocedal, J., 1994. L-BFGS-B: a limited memory FORTRAN code for solving bound constrained optimization problems. EECS Department, Northwestern University, Evanston, IL, Technical Report No. NAM-11.