What do you Mean?
The Role of the Mean Function in Bayesian Optimisation

George De Ath
g.de.ath@exeter.ac.uk
Department of Computer Science
University of Exeter
Exeter, United Kingdom

Jonathan E. Fieldsend
j.e.fieldsend@exeter.ac.uk
Department of Computer Science
University of Exeter
Exeter, United Kingdom

Richard M. Everson
r.m.everson@exeter.ac.uk
Department of Computer Science
University of Exeter
Exeter, United Kingdom

1 INTRODUCTION

Bayesian optimisation (BO) is a popular approach for optimising expensive black-box functions. The next location to be evaluated is selected via maximising an acquisition function that balances exploitation and exploration. Gaussian processes, the surrogate models of choice in Bayesian optimisation, are often used with a constant prior mean function equal to the arithmetic mean of the observed function values. We show that the rate of convergence can depend sensitively on the choice of mean function. We empirically investigate 8 mean functions (constant functions equal to the arithmetic mean, minimum, median and maximum of the observed function values, linear, quadratic polynomials, random forests and RBF networks), using 10 synthetic test problems and two real-world problems, and using the Expected Improvement and Upper Confidence Bound acquisition functions.

We find that for design dimensions \( \geq 5 \) using a constant mean function equal to the worst observed quality value is consistently the best choice on the synthetic problems considered. We argue that this worst-observed-quality function promotes exploitation leading to more rapid convergence. However, for the real-world tasks the more complex mean functions capable of modelling the fitness landscape may be effective, although there is no clearly optimum choice.

CCS CONCEPTS
• Theory of computation → Gaussian processes; Mathematical optimization; Computing methodologies → Modeling and simulation; Optimization algorithms.

KEYWORDS
Bayesian optimisation, Surrogate modelling, Gaussian process, Mean function, acquisition function

ACM Reference Format:
George De Ath, Jonathan E. Fieldsend, and Richard M. Everson. 2020. What do you Mean? The Role of the Mean Function in Bayesian Optimisation. In Genetic and Evolutionary Computation Conference Companion (GECCO ’20 Companion), July 8–12, 2020, Cancún, Mexico. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3377929.3398118
Algorithm 1: Sequential Bayesian optimisation.

Inputs:
- \( M \): Number of initial samples
- \( T \): Budget on the number of expensive evaluations

Steps:
1. \( X \leftarrow \text{SpaceFillingSampling}(X, M) \) \quad \text{Initial samples}
2. for \( t = 1 \rightarrow M \) do
3. \( f_t \leftarrow f(x_t) \) \quad \text{Expensively evaluate all initial samples}
4. \( D \leftarrow \{(x_t, f_t)\}_{t=1}^M \)
5. for \( t = M + 1 \rightarrow T \) do
6. \( \theta \leftarrow \text{TrainGP}(D) \) \quad \text{Train a GP model}
7. \( x' \leftarrow \text{argmax}_{x \in X} \alpha(x, \theta) \) \quad \text{Maximise infill criterion}
8. \( f' \leftarrow f(x') \) \quad \text{Expensively evaluate} \( x' \)
9. \( D \leftarrow D \cup \{(x', f')\} \) \quad \text{Augment training data}
10. return \( D \)

Bayesian optimisation (BO), also known as Efficient Global Optimisation (EGO), is a surrogate-assisted global search strategy that sequentially samples design space at likely locations of the global optimum, taking into account both the surrogate model’s prediction \( \mu(x) \) and the associated prediction uncertainty \( \sigma(x) \) [22]. See [7, 14, 37] for comprehensive reviews of BO. Without loss of generality, we can define the problem of finding a global minimum of an unknown objective function \( f : \mathbb{R}^d \rightarrow \mathbb{R} \) as

\[
\min_{x \in \mathcal{X}} f(x),
\]

(1)

where \( \mathcal{X} \subset \mathbb{R}^d \) is the feasible design space of interest. We assume that \( f \) is a black-box function, i.e. it has no simple closed form, but that we can have access the results of its evaluations \( f(x) \) at any location \( x \in \mathcal{X} \), although evaluating \( f(x) \) is expensive so that the number of evaluations required to locate the global optimum should be minimised.

Algorithm 1 outlines the BO procedure. It starts (line 1) with a space filling design, typically Latin hypercube sampling [27], of the feasible space. These samples \( X = \{x_i\}_{i=1}^M \) are then expensively evaluated with the function \( f_i = f(x_i) \), and a training dataset \( D \) is constructed (line 4). Then, at each iteration of the sequential algorithm, a regression model, usually a Gaussian process (GP), is constructed and trained (line 6) using the current training data. The choice of where next to expensively evaluate is determined by maximising an acquisition function (or infill criterion) \( \alpha(x) \) which balances the exploitation of regions of design space that are predicted to yield good-quality solutions and exploration of regions of space where the predictive uncertainty is high. The design \( x' \) maximising \( \alpha(x) \) is expensively evaluated and the training data is subsequently augmented (lines 7 to 9). This process is then repeated until the budget has been expended.

Two of the most popular acquisition functions are Expected Improvement (EI) [28] and Upper Confidence Bound (UCB) [39]. EI measures the positive predicted improvement over the best solution evaluated thus far: \( f^* \):

\[
\alpha_{EI}(x) = \sigma(x)(\Phi(s) + \phi(s)),
\]

(2)

where \( s = (f^* - \mu(x))/\sigma(x) \) is the predicted improvement at \( x \) normalised by the uncertainty, and \( \Phi(\cdot) \) and \( \phi(\cdot) \) are the Gaussian cumulative density and probability density functions respectively. UCB is a weighted sum of the mean prediction and its associated uncertainty:

\[
\alpha_{UCB}(x) = -\left(\mu(x) - \sqrt{\beta_t \sigma(x)}\right),
\]

(3)

where \( \beta_t \geq 0 \) is a weight that depends on the number of function evaluations \( t \) and explicitly controls the exploitation vs. exploitation trade-off. Note that both EI and UCB are presented here in the form used for minimisation. While other acquisition functions have been proposed, such as probability of improvement [25] and entropy-based methods such as predictive entropy search [17] and max-value search [42], we limit our investigation to the commonly used EI and UCB to allow the focus on the mean functions themselves.

3 GAUSSIAN PROCESSES

Gaussian processes are a common choice of surrogate model due to their strengths in uncertainty quantification and function approximation [34, 37]. A GP is a collection of random variables, and any finite number of these have a joint Gaussian distribution [34]. We can define a Gaussian process prior over \( f \) to be \( \mathcal{GP}(m(x), k(x, x' \mid \theta)) \), where \( m(\cdot) \) is the mean function and \( k(\cdot, \cdot \mid \theta) \) is the kernel function (covariance function) with hyperparameters \( \theta \). Given data consisting of \( f \) evaluated at \( t \) sampled locations \( D = \{x_n, f_n = f(x_n)\}_{n=1}^t \), the posterior distribution of \( f \) at a given location \( x \) is Gaussian:

\[
p(f \mid x, D, \theta) = N(f \mid \mu(x), \sigma^2(x))
\]

(4)
What do you Mean? The Role of the Mean Function in Bayesian Optimisation

GECCO ’20 Companion, July 8–12, 2020, Cancún, Mexico

$c = f_{\text{min}} = \min\{f_1, f_2, \ldots, f_t\}$

$c = \bar{f} = \frac{1}{t} \sum_i f_i$

$c = f_{\text{max}} = \max\{f_1, f_2, \ldots, f_t\}$

Figure 1: GP models with identical training data (blue) and kernel hyperparameters, along with different constant mean functions. The mean function used in each GP is shown (grey, dashed) and corresponds to the value of the smallest (left), arithmetic mean (centre), and largest (right) seen function values. The lower row shows the corresponding EI and its maximiser (red dashed). Note how the min and max models lead EI to prefer exploring and exploiting respectively.

Practitioners of BO usually standardise the observations $f_i$ before fitting the GP at each iteration, i.e. they subtract the mean of the observations and divide by the standard deviation of the relevant feature/variable after which the mean function is taken as the constant function equal to zero; that is, the effective mean function is the constant function equal to the arithmetic mean of the observed function values. In addition, the prior variance of the GP is matched to the observed variance of the function values. Although the standardisation is not usually discussed in the literature, it is commonplace in standard BO libraries, e.g. GPyOpt [2], BoTorch [3] and Spearmint³. It is, however, unclear whether this is the best choice for BO or whether a different mean function may be preferable. We now introduce the mean functions that will be evaluated in this work.

Here, we consider the mean function to be a set of basis functions $h(x)$ with corresponding weights $w$ [34]:

$$m(x) = h(x)^\top w.$$

A constant mean function with value $c$, for example, can be written as $h(x) = c1$ with $w = 1$. In addition to the standard constant function equal to the arithmetic mean of the data $c = \bar{f} = f^1 \sum_i f_i$, we consider three other constant values: using the best and worst seen observation’s value at each iteration, i.e. $c = f_{\text{min}} = f^* = \min\{f_1, f_2, \ldots, f_t\}$ and $c = f_{\text{max}} = \max\{f_1, f_2, \ldots, f_t\}$, and the median observed value $c = f_{\text{med}} = \text{median}\{f_1, f_2, \ldots, f_t\}$. In comparison to using the data mean $c = \bar{f}$, using $c = f_{\text{min}}$ leads to acquisition functions becoming more exploratory, as illustrated in Figure 1 (left panel). This is because locations far away from previously evaluated solutions $D$ will have predicted means $\mu(x)$ equal to the $f^*$ and with large predicted uncertainty, leading to large values of, for example, EI and UCB. Conversely, using $c = f_{\text{max}}$ will, as illustrated in Figure 1 (right panel), lead to increased exploitation due to regions far from $D$ having large uncertainty, but poor predicted values and hence small $\alpha(x)$. Interestingly, the effect of using $c = \bar{f}$ or $c = f_{\text{med}}$ will change over the course of the optimisation. When there are relatively few function evaluations $\bar{f}$ and $f_{\text{med}}$ will be approximately $(f_{\text{min}} + f_{\text{max}})/2$. However, as the number of

³https://github.com/HIPS/Spearmint
expensive function evaluations increases and the optimisation converges towards the (estimated) optimum, \( f \) and particularly \( f_{\text{med}} \) will tend to approach \( f_{\text{min}} \), thus leading to increased exploration.

We also consider linear and quadratic mean functions. Linear mean functions are defined as \( h(x) = [1, x(1), x(2), \ldots , x(d)] \), with corresponding weights \( w \in \mathbb{R}^{d+1} \) and where \( x(i) \) refers to the \( i \)th element of \( x \). Quadratic mean functions are defined similarly, with polynomial terms up to degree 2 and with weights \( w \in \mathbb{R}^q \), where \( q = \binom{d+2}{2} \). To avoid overfitting to the data, these regressions are typically trained via a regularised least-squares approximation, i.e. ridge regression (also known as Tikhonov regularization). The optimal regularised weights \( w^* \) are estimated by solving

\[
  w^* = \arg\min_w \| f - Hw \|^2 + \lambda \| w \|^2, \tag{10}
\]

where \( H = [h(x_1), h(x_2), \ldots , h(x_t)] \) and \( \lambda \geq 0 \) controls the amount of regularisation. The ordinary least squares estimator is

\[
  w^* = (H^TH + \lambda I)^{-1} H^Tf, \tag{11}
\]

In this work, the regularisation parameter \( \lambda \) was chosen via five-fold cross-validation for \( \lambda \in \{10^{-6}, 10^{-5}, \ldots , 10^1, 10^2\} \).

Another choice of basis functions are radial basis functions (RBFs), which have the property that each basis function only depends on the Euclidean distance from a fixed centre [4]. These are known as RBF networks and can be thought of as either linear neural networks using RBF activation functions [31] or as finite-dimensional Gaussian processes [4]. A commonly used set of basis functions, and the ones used in this work, are the Gaussian RBFs \( \phi_i(x) = \exp(-\gamma \| x - z_i \|) \). While any set of locations can be used as the centres \( z_i \), we place a Gaussian RBF at each of the previously-evaluated locations, i.e. \( z_i = x_i \), \( \forall i = 1, \ldots , t \), resulting in \( h(x) = [\phi_1(x), \phi_2(x), \ldots , \phi_t(x)] \). Similarly to the linear and quadratic mean functions, the regularisation parameter \( \lambda \) and length scale \( \gamma \) were chosen via a five-fold cross-validation. Values of \( \lambda \) were selected in the same range as for the linear and quadratic mean functions, and the values of \( \gamma \) were selected from \( \gamma \in \{10^{-2}, 10^{-2.5}, \ldots , 10^{1.5}, 10^2\} \). A more fine-grained selection of values was chosen following a preliminary investigation which revealed the modelling error to be more sensitive to changes in \( \gamma \) than \( \lambda \). We note here that the use of regularisation is particularly important when placing an RBF on each evaluated location because the RBF network will otherwise be able to perfectly interpolate the data. This would lead to \( f \equiv m \) and therefore (8) would reduce to \(-\frac{1}{2} \log[K]\), which can be maximised by either \( \theta_0 \to 0 \) or \( \theta_1 \to \infty \) in (7). This results in the posterior variance estimates \( \sigma^2(x) \) being over-confident and thus having small variance everywhere with predictions determined by the mean function.

Lastly, we include a non-parametric regressor, extremely randomised trees, better known as Extra-Trees (ET, [15]), a variant of Random Forests (RF, [6]). RFs are ensembles of classification or regression trees that are each trained on a different randomly chosen subsets of the data. Unlike RFs, that attempt to split the data at each cut-point of a tree optimally, the ET method instead selects the cut-point as the best from a small set of randomly chosen cut-points; this additional randomisation results in a smoother regression in comparison to RFs. Given that ETs use randomised cut-points, they typically use all the training data in each tree. However, to counteract the overfitting that this will produce, we allow each item in the training set to be resampled, instead of using all elements of the set for each tree. Note that, while not presented as such here, RFs can also be interpreted as a kernel method [10, 36].

4 EXPERIMENTAL EVALUATION

We now investigate the performance of the mean functions discussed in Section 3.1 using the EI and UCB acquisition functions (Section 2) on ten well-known benchmark functions with a range dimensionality and landscape properties, and two real-world applications. Full results of all experimental evaluations are available in the supplementary material. The mean functions to be evaluated are the constant functions, \( c = f \), \( c = f_{\text{med}} \), \( c = f_{\text{min}} \), and \( c = f_{\max} \), labelled Arithmetic, Median, Min, and Max respectively, as well as the Linear, Quadratic, Extra-Trees (RandomForest), and RBF network-based (RBF) mean functions.

A Gaussian process surrogate model with an isotropic Matérn 5/2 kernel (7) was used in all experiments. The Bayesian optimisation runs themselves were carried out as in Algorithm 1, with the additional step of fitting a mean function before training the GP (Algorithm 1, line 6) at each iteration. All test problems evaluated in this work were scaled to \([0, 1]^d\), and observations were standardised at each BO iteration, prior to mean function fitting. The models were initially trained on \( M = 2d \) observations generated by maximin Latin hypercube sampling [27], and each optimisation run was repeated 51 times with different initialisation. The same set of 51 initial observations were used for each of the mean functions to enable statistical comparisons. The hyperparameters \( \theta \) of the GP were optimised by maximising the marginal log likelihood (8) with L-BFGS-B [8] using 10 restarts. Following common practise, maximisation of the acquisition functions was carried out via multi-start optimisation; details of the full procedure can be found in [3].

The trade-off between exploitation and exploration in UCB, \( \beta_t \), is set to \( \theta_t \in [0, 1]^d \). Some additional step of fitting a mean function before training the GP was repeated 51 times with different initialisation. The same set of 51 initial observations were used for each of the mean functions to enable statistical comparisons. The hyperparameters \( \theta \) of the GP were optimised by maximising the marginal log likelihood (8) with L-BFGS-B [8] using 10 restarts. Following common practise, maximisation of the acquisition functions was carried out via multi-start optimisation; details of the full procedure can be found in [3].

Optimisation quality is measured with simple regret \( R_t \), which is the difference between the true minimum value \( f(x^*) \) and the best value found so far after \( t \) evaluations:

\[
  R_t = |f(x^*) - \min\{f_1, f_2, \ldots , f_t\}|. \tag{12}
\]

4.1 Synthetic Experiments

The mean functions were evaluated on the ten popular synthetic benchmark functions listed in Table 1 with a budget of 200 function evaluations that included the initial 2d LHS samples. These functions were selected due to their different dimensionality and landscape properties, such as the presence of multiple local or global minima (Ackley, Eggholder, Hartmann6, GoldsteinPrice, Styblinski-Tang, Shekel) deep, valley-like regions (Branin, Rosenbrock, Six-HumpCamel), and steep ridges and drops (Michalewicz).

\[\text{http://www.github.com/georgedeath/bomean}\]
The role of the mean function in Bayesian optimisation is discussed. The mean function has relatively little influence, especially when using EI, apart from the RandomForest mean function in all Bayesian optimisation tasks. Based on these results we suggest that an increase in performance may be obtained by using the Max mean function with EI. Although there does not appear to be such a clear-cut answer as to which mean function should be used in conjunction with the UCB acquisition function, we posit that this is less important because, based on these optimisation results, one would prefer the performance of EI over UCB in general.

### 4.2 Active Learning for Robot Pushing

Like [11, 12, 21, 42] we optimise the control parameters for two active learning robot pushing problems [41]; see [12] for a diagrammatic outline of the problems. In the \(d = 4\) \textsc{push4} problem, a robot should push an object towards an unknown target location and is constrained such that it can only travel in the initial direction of the object. Once the robot has finished pushing, it receives feedback in the form of the object-target distance. The robot is parametrised by its initial location, the orientation of its pushing hand and how long it pushes for. This can therefore be cast as a minimisation problem in which the four parameter values are optimised with respect to the object’s final distance from the target. The object’s initial location is fixed to the centre of the problem domain [42] and the target location is changed for each of the 51 optimisation runs, with these kept the same across mean functions. Thus, the optimisation performance is considered over problem instances rather than initialisations of a single instance.

Table 1 lists the median regret for each function using the mean function in all Bayesian optimisation tasks. Based on these results we suggest that an increase in performance may be obtained by using the Max mean function with EI. Although there does not appear to be such a clear-cut answer as to which mean function should be used in conjunction with the UCB acquisition function, we posit that this is less important because, based on these optimisation results, one would prefer the performance of EI over UCB in general.

Table 1: Synthetic functions used and their dimensionality \(d\). Formulae for all functions can be found at http://www.sfu.ca/~ssurjano/optimization.html.

| Name           | \(d\) | Name           | \(d\) |
|----------------|------|----------------|------|
| Branin         | 2    | Ackley         | 5    |
| Eggholder      | 2    | Hartmann6      | 6    |
| GoldsteinPrice | 2    | Michalewicz    | 10   |
| SixHumpCamel   | 2    | Rosenbrock     | 10   |
| Shekel         | 4    | StyblinskiTang | 10   |

Table 2 shows the median regret over the 51 repeated experiments, together with the median absolute deviation from the mean (MAD) for the mean functions using the UCB acquisition function. Due to space constraints, the corresponding table for UCB is included in the supplementary material. The method with the lowest (best) regret on each function is highlighted in dark grey, and those highlighted in light grey are statistically equivalent to the best method according to a one-sided paired Wilcoxon signed-rank test [24] with Holm-Bonferroni correction [18] (\(p \geq 0.05\)).

The convergence of the various mean functions on 8 illustrative test problems are shown using the EI (Figure 2) and UCB (Figure 3) acquisition functions. Convergence plots for the Branin and GoldsteinPrice test problems were visually similar to Eggholder and SixHumpCamel respectively; they are available in the supplementary material. As one might expect, because points are naturally less distant from one another, the choice of mean function has less impact in 2 dimensions. Although, interestingly, optimisation runs with the UCB algorithm in \(d = 2\) achieve lower regret with the constant mean functions compared to the others evaluated.

Perhaps surprisingly, the non-constant mean functions, Linear, Quadratic, RandomForest and RBF, appear to offer no advantage over the constant mean functions despite their ability to model the large scale optimisation landscape. The Quadratic model, in two dimensions where there are only three parameters to be fitted, appears to be well suited to the SixHumpCamel function which is roughly bowl-shaped (albeit with quartic terms); however, this appears to be an exceptional case.

In higher dimensions with the EI acquisition function, using the worst observation value as the constant mean function (Max) consistently provides the lowest regret on the test functions evaluated. This is consistent with recent work [12, 35] showing that being more exploitative in higher dimensions is preferable to most other strategies. However, for the UCB acquisition function this is not the case and no mean function is consistently best. We suspect that this is because the value of \(\beta\) is so large that the UCB function (3) is always dominated by the exploratory term \((\sqrt{\sigma(x)})\) so that the mean function has relatively little influence. This is in contrast to EI, which has been shown [12] to be far more exploitative than UCB.

The standard choice of using a constant mean function equal to \(\bar{f}\) the arithmetic mean of the observations (Arithmetic) is, in higher dimensions \((d \geq 5)\), only statistically equivalent to the best-performing method on one of the five test functions for both EI and UCB. This result calls into question the efficacy of the common practise of using the \(\bar{f}\) constant mean function in all Bayesian optimisation tasks.
Table 2: Mean function performance using the EI acquisition function. Median regret (column left) and median absolute deviation from the median (MAD, column right) after 200 function evaluations across the 51 runs. The method with the lowest median performance is shown in dark grey, with those statistically equivalent shown in light grey.

| Mean function | Branin (2) | Eggholder (2) | GoldsteinPrice (2) | SixHumpCamel (2) | Shekel (4) |
|---------------|-----------|--------------|-------------------|------------------|-----------|
| Arithmetic    | Median    | MAD          | Median            | MAD              | Median    |
|               | 1.35 × 10^{-5} | 1.82 × 10^{-5} | 1.58             | 1.93             | 4.18 × 10^{-2} | 5.32 × 10^{-2} | 2.51 × 10^{-5} | 2.58 × 10^{-5} | 8.13 × 10^{-2} | 1.19 × 10^{-1} |
| Median        | 9.35 × 10^{-6} | 1.17 × 10^{-5} | 3.02             | 2.67             | 5.03 × 10^{-2} | 5.59 × 10^{-2} | 1.40 × 10^{-5} | 1.49 × 10^{-5} | 1.54 × 10^{-1} | 2.24 × 10^{-1} |
| Min           | 1.18 × 10^{-5} | 1.31 × 10^{-5} | 2.82             | 3.13             | 6.25 × 10^{-2} | 8.03 × 10^{-2} | 2.25 × 10^{-2} | 2.29 × 10^{-2} | 7.02           | 9.75 × 10^{-1} |
| Max           | 7.21 × 10^{-6} | 8.81 × 10^{-6} | 2.69             | 2.48             | 6.87 × 10^{-2} | 7.40 × 10^{-2} | 1.52 × 10^{-5} | 1.84 × 10^{-5} | 7.16 × 10^{-2} | 1.05 × 10^{-1} |
| Linear        | 5.13 × 10^{-6} | 5.58 × 10^{-6} | 2.82             | 2.85             | 1.05 × 10^{-1} | 1.19 × 10^{-1} | 7.98 × 10^{-6} | 8.14 × 10^{-6} | 6.47           | 1.57           |
| Quadratic     | 1.05 × 10^{-5} | 1.09 × 10^{-5} | 3.59             | 4.03             | 5.55 × 10^{-2} | 6.03 × 10^{-2} | 6.27 × 10^{-6} | 6.52 × 10^{-6} | 6.47           | 1.30           |
| RandomForest  | 5.35 × 10^{-4} | 4.86 × 10^{-4} | 3.93             | 4.22             | 2.41           | 2.14           | 2.96 × 10^{-4} | 3.56 × 10^{-4} | 2.84           | 2.76           |
| RBF           | 1.36 × 10^{-4} | 1.07 × 10^{-4} | 3.27             | 3.45             | 1.76           | 2.39           | 3.76 × 10^{-5} | 4.63 × 10^{-5} | 7.95           | 1.05           |

Figure 2: Illustrative convergence plots for eight benchmark problems using the EI acquisition function. Each plot shows the median regret, with shading representing the interquartile range across the 51 runs and the dashed vertical line indicating the end of the initial LHS phase.
What do you Mean? The Role of the Mean Function in Bayesian Optimisation

**4.3 Pipe Shape Optimisation**

Lastly, we evaluate the mean functions on a real-world computational fluid dynamics (CFD) optimisation problem. The goal of the PitzDaily CFD problem [9] is to minimise the pressure loss between a pipe’s entrance (inflow) and exit (outflow) by optimising the shape...
of the pipe’s lower wall. The loss is evaluated by generating a CFD mesh and simulating the two-dimensional flow using OpenFOAM [20], with each function evaluation taking between 60 and 90 seconds. The decision variables in the problem are the control points of a Catmull-Clark subdivision curve that represents the lower wall’s shape; see [9] for a pictorial representation of these. In this work we use 5 control points, resulting in a 10-dimensional decision vector. The control points are constrained to lie in a polygon, rather than a hypercube for all previous problems, and, therefore, we draw initial samples uniformly from within the constrained region rather than using LHS. Similarly, we use CMA-ES [16] to optimise the acquisition functions and penalise locations that violate the constraints.

Convergence plots of the flow loss for the mean functions with EI and UCB are shown in Figure 5. The Arithmetic, Min and Max constant mean functions, when using EI, are all statistically equivalent and the best-performing. It is interesting to note the contrasting performance between the RandomForest mean function on this and the robot pushing tasks.

As shown in Figure 5, using the UCB acquisition function leads to the constant mean and Linear mean functions all having a median flow loss within $10^{-4}$ of one another, with their inter-quartile ranges rapidly decreasing. This effect can also be seen towards the end of the optimisation runs with EI. Inspection of solutions (control points) with a flow loss $\approx 0.084$ revealed that they all had distinct values but that they led to very similar sub-division curves. This implies that they all represented essentially the same inner wall shape and thus indicate the presence of either one large, valley-like global optimum or many, global optima. We suggest that this may be the actual minimum flow loss achievable for this problem. All mean functions, in combination with both EI and UCB, were able to successfully discover solutions that led to a flow loss of less than 0.0903 found by a local, gradient-based method in [30] that used approximately 500 function evaluations. This highlights the strength of Bayesian optimisation in general because the convergence rates shown in Figure 5 are far more rapid for the majority of mean functions and realise better solutions than the local, adjacent method.

![Figure 5: Convergence plots for the PitzDaily test problem using EI and UCB. Each plot shows the median regret, with shading representing the interquartile range of the 51 runs.](image)

5 CONCLUSION

We have investigated the effect of using different prior mean functions in the Gaussian process model during Bayesian optimisation when using the expected improvement and upper confidence bound acquisition functions. This was assessed by performing BO on ten synthetic functions and two real-world problems. The constant mean function Max, which uses a constant value of the worst-seen expensive function evaluation thus far, was found to consistently out-perform the other mean functions across the synthetic functions in higher dimensions, and was statistically equivalent to the best performing mean function on nine out of ten functions. We suggest that this is because this mean function tends to promote exploitation which can lead to rapid convergence in higher dimensions [12, 35] because exploration is implicitly provided through the necessarily inaccurate surrogate modelling. However, on the two, real-world problems this trend did not continue, but its performance was still statistically equivalent to the commonly-used mean equal to the arithmetic mean of the observations. For this reason we recommend using the Max mean function in conjunction with expected improvement for, at worst, the same performance as a zero mean and generally improved performance in higher dimensions.

Interestingly, the lack of consistency between mean function performance on the synthetic and real-world problems may indicate a larger issue in BO, namely that synthetic benchmarks do not always contain the same types of functional landscapes as real-world problems. In future work we would like to characterise a function’s landscape during the optimisation procedure and adaptively select the best-performing components of the BO pipeline, e.g. mean function, kernel, and acquisition function, to suit the problem structure.

This work focused on learning a mean function independent of the training of the GP. In further work we would like to jointly learn the parameters of the mean function, i.e. the weights in (9), alongside the hyperparameters of the GP itself. The efficacy of the BO approach clearly depends crucially on the ability of the surrogate model to accurately predict the function’s value in unvisited locations. We therefore look forward to evaluating a fully-Bayesian approach that marginalises over the mean function parameters and kernel hyperparameters. Although the Monte Carlo sampling required to evaluate the resulting acquisition functions may be substantial, an important area of investigation is whether fully-Bayesian models can significantly improve the convergence of Bayesian Optimisation.

ACKNOWLEDGMENTS

This work was supported by Innovate UK [grant number 104400].

REFERENCES

[1] Erdem Acar. 2013. Effects of the correlation model, the trend model, and the number of training points on the accuracy of kriging metamodels. Expert Systems 30, 5 (2013), 418–428.
[2] The GPyOpt authors. 2016. GPyOpt: A Bayesian Optimization framework in python. http://github.com/SheffieldML/GPyOpt.
[3] Maximilian Balandat, Brian Karner, Daniel R. Jiang, Samuel Daulton, Benjamin Letham, Andrew Gordon Wilson, and Eytan Bakshy. 2019. BoTorch: Programmable Bayesian Optimization in PyTorch. arXiv:1910.06403
[4] Christopher M. Bishop. 2006. Pattern Recognition and Machine Learning. Springer, Berlin, Heidelberg.
[5] B. J. N. Blight and L. Ott. 1975. A Bayesian approach to model inadequacy for polynomial regression. *Biometrika* 62, 1 (1975), 79–88.

[6] Leo Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (2001), 5–32.

[7] Eric Brochu, Vlad M Cora, and Nando De Freitas. 2010. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. arXiv:1012.2599

[8] Richard H Byrd, Pei-Huang Lu, Jorge Nocedal, and Ciyou Zhu. 1995. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing* 16, 5 (1995), 1190–1208.

[9] Steven J. Daniels, Alma A. M. Rahat, Richard M. Everson, Gavin R. Tabor, and Jonathan E. Fieldsend. 2018. A Suite of Computationally Expensive Shape Optimisation Problems Using Computational Fluid Dynamics. In *Parallel Problem Solving from Nature – PPSN XV*. Springer, 296–307.

[10] Alex Davies and Zoubin Ghahramani. 2014. The Random Forest Kernel and other kernels for big data from random partitions. arXiv:1402.4293

[11] George De Ath, Richard M. Everson, Jonathan E. Fieldsend, and Alma A. M. Rahat. 2020. γ-shooting: e-greedy Batch Bayesian Optimisation. In *Proceedings of the Genetic and Evolutionary Computation Conference*. Association for Computing Machinery.

[12] George De Ath, Richard M. Everson, Alma A. M. Rahat, and Jonathan E. Fieldsend. 2019. Greed is Good: Exploration and Exploitation Trade-offs in Bayesian Optimisation. arXiv:1911.1289

[13] Vincent Fortuin, Heiko Strafmann, and Gunnar RÅ‡isch. 2019. Meta-Learning Mean Functions for Gaussian Processes. arXiv:1901.08998

[14] Peter I Frazier. 2018. A tutorial on Bayesian optimization. arXiv:1807.02811

[15] Pierre Geurts, Damien Ernst, and Louis Wehenkel. 2006. Extremely randomized trees. *Machine Learning* 63, 1 (2006), 3–42.

[16] Nikolaus Hansen. 2009. Benchmarking a B1-population CMA-ES on the BBOB-2009 function testbed. In *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*. ACM, 2389–2396.

[17] José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. 2017. Improving Output Uncertainty Estimation and Generalization in Deep Learning via Neural Network Gaussian Processes. arXiv:1707.05922

[18] Sture Holm. 1979. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics* 6, 2 (1979), 65–70.

[19] Tomoharu Iwata and Zoubin Ghahramani. 2017. Improving Output Uncertainty Estimation and Generalization in Deep Learning via Neural Network Gaussian Processes. arXiv:1707.05922

[20] Hrvoje Jasak, Aleksandar Jemcov, and ˘Zeljko Tuković. 2007. *OpenFOAM: A C++ Library for Complex Physics Simulations*. In *International Workshop on Coupled Methods in Numerical Dynamics*. 1–20.

[21] Shali Jiang, Henry Chai, Javier González, and Roman Garnett. 2019. Efficient nonmyopic Bayesian optimization and quadrature. arXiv:arXiv:1909.04568

[22] Donald R. Jones, Matthias Schonlau, and William J. Welch. 1998. Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization* 13, 4 (1998), 455–492.

[23] Marc C. Kennedy and Anthony O’Hagan. 2001. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63, 3 (2001), 425–464.

[24] Joshua D. Knowles, Lothar Thiele, and Eckart Zitzler. 2006. *A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers*. Technical Report TK1-214. Computer Engineering and Networks Laboratory, ETH Zurich, Zurich, Switzerland.

[25] Harold J. Kushner. 1964. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal Basic Engineering* 86, 1 (1964), 97–106.

[26] Marius Lindauer, Matthias Feurer, Katharina Eggensperger, André Biedenkapp, and Frank Hutter. 2019. Towards Assessing the Impact of Bayesian Optimisation’s Own Hyperparameters. In *IJCAI 2019 DSO Workshop*.

[27] Michal D McKay, Richard J. Beckman, and William J. Conover. 2000. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics* 42, 1 (2000), 55–61.

[28] Jonsh Mokeien, Vytautas Tieis, and Antanas Žilinskas. 1978. The application of Bayesian methods for seeking the extremum. *Towards Global Optimization* 2, 1 (1978), 117–129.

[29] Tammo Mukhopadhyay, S Chakraborty, S Dey, S Adhikari, and R Chowdhury. 2017. A critical assessment of Krigeing model variants for high-fidelity uncertainty quantification in dynamics of composite shells. *Archives of Computational Methods in Engineering* 24, 3 (2017), 495–518.

[30] Ulf Nielsen, Daniel Lindblad, and Olivier Petit. 2014. Description of adjointShapeOptimizationFoam and how to implement new objective functions. Technical Report. Chalmers University of Technology, Gothenburg, Sweden.

[31] Mark J. L Orr. 1996. *Introduction to radial basis function networks*. Technical Report University of Edinburgh.

[32] Pramodita Saria Palar and Koji Shimoyama. 2019. Efficient global optimization with ensemble and selection of kernel functions for engineering design. *Structural and Multidisciplinary Optimisation* 59, 1 (2019), 93–116.

[33] Pramodita S Palar, Lavi R. Zabal, Timidle Chough, and Alma Rahat. 2020. On the Impact of Covariance Functions in Multi-Objective Bayesian Optimization for Engineering Design. In *AAIA Scitech 2020 Forum*. American Institute of Aeronautics and Astronautics.

[34] Carl Edward Rasmussen and Christopher K I Williams. 2006. *Gaussian processes for machine learning*. The MIT Press, Boston, MA.

[35] Frederik Rehbach, Martin Zaefferer, Boris Nautjoks, and Thomas Bartz-Beielstein. 2020. Expected Improvement versus Predicted Value in Surrogate-Based Optimisation. arXiv:2001.02957

[36] Erwan Scornet. 2016. Random Forests and Kernel Methods. *IEEE Transactions on Information Theory* 62, 3 (2016), 1485–1500.

[37] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. 2016. Taking the human out of the loop: A review of Bayesian optimization. Proc. IEEE 104, 1 (2016), 148–175.

[38] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. 2012. Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2951–2959.

[39] Niranjani Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. 2010. Gaussian process optimization in the bandit setting: no regret and experimental design. In *Proceedings of the 27th International Conference on Machine Learning*. Omnipress, 1015–1022.

[40] Michael L. Stein. 2012. *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media.

[41] Zi Wang, Caelan Reed Garrett, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. 2016. Taking the human out of the loop: A review of Bayesian optimization. *IEEE Transactions on Evolutionary Computation* 20 Companion, July 8–12, 2020, Cancún, Mexico. 2019. *Towards Global Optimization*. Proceedings of the 34th International Conference on Machine Learning. PMLR, 3627–3635.

[42] Zi Wang, and Stefanie Jegelka. 2017. *Max-value entropy search for efficient Bayesian optimization*. In *Proceedings of the 34th International Conference on Machine Learning*. PMLR, 3627–3635.

[43] Daniel Williamson, Adam T Blaker, Charlotte Hampton, and James Salter. 2015. Identifying and removing structural biases in climate models with history matching. *Climate dynamics* 45, 5 (2015), 1299–1324.

[44] David H. Wolpert and William G. Macready. 1997. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation* 1, 1 (1997), 67–82.
A ADDITIONAL RESULTS

In this section we show convergence plots for the Branin and GoldsteinPrice synthetic test functions for both the EI and UCB acquisition functions, as well as the results tables for UCB on the synthetic functions, and EI and UCB on the three real-world problems.

Fig. 1. Convergence plots for the Branin and GoldsteinPrice test problem using EI (left) and UCB (right). Each plot shows the median regret, with shading representing the interquartile range across the 51 runs.
Table 1. Mean function performance using the UCB acquisition function. Median regret (column left) and median absolute deviation from the median (MAD, column right) after 200 function evaluations across the 51 runs. The method with the lowest median performance is shown in dark grey, with those statistically equivalent shown in light grey.

| Mean function | Branin (2) | Eggholder (2) | GoldsteinPrice (2) | SixHumpCamel (2) | Shekel (4) |
|---------------|------------|---------------|---------------------|------------------|-----------|
|               | Median     | MAD           | Median              | MAD              | Median    |
| Arithmetic    | 1.27 × 10⁻⁴| 1.47 × 10⁻⁴   | 2.47                | 2.77             | 1.37      |
| Median        | 1.11 × 10⁻⁴| 1.44 × 10⁻⁴   | 3.21                | 2.76             | 1.21      |
| Min           | 1.27 × 10⁻⁴| 1.57 × 10⁻⁴   | 4.09                | 3.09             | 1.12      |
| Max           | 1.46 × 10⁻⁴| 1.55 × 10⁻⁴   | 3.31                | 3.01             | 1.04      |
| Linear        | 9.20 × 10⁻⁵| 8.31 × 10⁻⁵   | 2.66                | 2.34             | 1.36      |
| Quadratic     | 3.05 × 10⁻³| 4.36 × 10⁻³   | 3.08                | 4.15             | 1.78      |
| RandomForest  | 4.30 × 10⁻³| 4.55 × 10⁻³   | 2.96                | 3.30             | 2.18      |
| RBF           | 1.59 × 10⁻³| 2.21 × 10⁻³   | 5.05                | 4.05             | 5.36      |

| Mean function | Ackley (5) | Hartmann6 (6) | Michalewicz (10) | Rosenbrock (10) | StylblinskiTang (10) |
|---------------|------------|---------------|-------------------|-----------------|---------------------|
|               | Median     | MAD           | Median             | MAD             | Median              |
| Arithmetic    | 3.53       | 6.53 × 10⁻¹   | 9.93 × 10⁻²        | 1.22 × 10⁻¹     | 7.90 × 10⁻²        |
| Median        | 2.37       | 5.60 × 10⁻¹   | 1.94 × 10⁻²        | 2.78 × 10⁻²     | 4.84 × 10⁻²        |
| Min           | 7.06       | 1.49          | 3.03 × 10⁻¹        | 1.28 × 10⁻¹     | 1.74                |
| Max           | 2.21       | 4.51 × 10⁻¹   | 4.40 × 10⁻²        | 6.48 × 10⁻²     | 3.92 × 10⁻¹        |
| Linear        | 3.35       | 8.75 × 10⁻¹   | 6.31 × 10⁻²        | 8.03 × 10⁻²     | 9.27 × 10⁻²        |
| Quadratic     | 4.75       | 1.71          | 1.87 × 10⁻¹        | 1.84 × 10⁻¹     | 6.54 × 10⁻²        |
| RandomForest  | 4.42       | 5.62 × 10⁻¹   | 8.03 × 10⁻²        | 6.30 × 10⁻²     | 1.51 × 10⁻¹        |
| RBF           | 3.67       | 1.05          | 1.33 × 10⁻¹        | 6.83 × 10⁻¹     | 7.21 × 10⁻¹        |

Table 2. Mean function performance using the EI (left table) and UCB (right table) acquisition functions. Median regret (column left) and median absolute deviation from the median (MAD, column right) after 200 function evaluations across the 51 runs are shown in paired columns. The method with the lowest median performance is shown in dark grey, with those statistically equivalent shown in light grey.

| Mean function | PUSH4 (4) | PUSH8 (8) |
|---------------|-----------|-----------|
|               | Median    | MAD       | Median    | MAD       |
| Arithmetic    | 1.53 × 10⁻¹| 1.16 × 10⁻¹| 2.77      | 1.93      |
| Median        | 1.54 × 10⁻¹| 6.53 × 10⁻²| 2.19      | 1.58      |
| Min           | 2.39 × 10⁻¹| 1.60 × 10⁻¹| 3.53      | 1.44      |
| Max           | 1.28 × 10⁻¹| 9.75 × 10⁻²| 2.66      | 1.83      |
| Quadratic     | 1.92 × 10⁻¹| 1.28 × 10⁻¹| 3.16      | 1.83      |
| RandomForest  | 6.56 × 10⁻²| 5.71 × 10⁻²| 3.07      | 2.24      |
| RBF           | 1.64 × 10⁻¹| 1.19 × 10⁻¹| 3.35      | 2.29      |

| Mean function | PUSH4 (4) | PUSH8 (8) |
|---------------|-----------|-----------|
|               | Median    | MAD       | Median    | MAD       |
| Arithmetic    | 3.42 × 10⁻¹| 2.18 × 10⁻¹| 3.95      | 1.57      |
| Median        | 5.01 × 10⁻³| 3.06 × 10⁻³| 3.64      | 2.05      |
| Min           | 4.36 × 10⁻¹| 3.69 × 10⁻¹| 4.37      | 1.72      |
| Max           | 4.25 × 10⁻¹| 3.20 × 10⁻¹| 4.57      | 1.74      |
| Quadratic     | 4.69 × 10⁻¹| 4.37 × 10⁻¹| 3.79      | 1.94      |
| RandomForest  | 1.05 × 10⁻¹| 6.92 × 10⁻²| 2.36      | 2.11      |
| RBF           | 1.39 × 10⁻¹| 9.29 × 10⁻²| 3.66      | 2.43      |
Table 3. Mean function performance using the EI (left table) and UCB (right table) acquisition functions. Median regret (column left) and median absolute deviation from the median (MAD, column right) after 200 function evaluations across the 51 runs are shown in paired columns. The method with the lowest median performance is shown in dark grey, with those statistically equivalent in light grey.

| Mean function        | PitzDaily (10) |          | Mean function | PitzDaily (10) |          |
|----------------------|----------------|----------|---------------|----------------|----------|
|                      | Median         | MAD      |               | Median         | MAD      |
| Arithmetic           | $8.39 \times 10^{-2}$ | $1.12 \times 10^{-3}$ | Arithmetic | $8.41 \times 10^{-2}$ | $3.92 \times 10^{-4}$ |
| Median               | $8.60 \times 10^{-2}$ | $3.23 \times 10^{-3}$ | Median       | $8.40 \times 10^{-2}$ | $1.96 \times 10^{-4}$ |
| Min                  | $8.43 \times 10^{-2}$ | $1.39 \times 10^{-3}$ | Min          | $8.39 \times 10^{-2}$ | $2.01 \times 10^{-4}$ |
| Max                  | $8.38 \times 10^{-2}$ | $1.38 \times 10^{-3}$ | Max          | $8.40 \times 10^{-2}$ | $3.07 \times 10^{-4}$ |
| Linear               | $8.50 \times 10^{-2}$ | $2.24 \times 10^{-3}$ | Linear       | $8.41 \times 10^{-2}$ | $3.98 \times 10^{-4}$ |
| Quadratic            | $8.57 \times 10^{-2}$ | $3.07 \times 10^{-3}$ | Quadratic    | $8.43 \times 10^{-2}$ | $9.23 \times 10^{-4}$ |
| RandomForest         | $8.84 \times 10^{-2}$ | $6.38 \times 10^{-3}$ | RandomForest | $8.77 \times 10^{-2}$ | $4.69 \times 10^{-3}$ |
| RBF                  | $8.79 \times 10^{-2}$ | $6.23 \times 10^{-3}$ | RBF          | $8.96 \times 10^{-2}$ | $8.23 \times 10^{-3}$ |

### B GP MODELLING ERROR USING DIFFERENT MEAN FUNCTIONS

In this section we present a short investigation into the modelling capability of the Gaussian process (GP) when combined with different mean functions. In order to quantify the modelling capability of a GP using a prior mean function on a test function $f: \mathbb{R}^d \mapsto \mathbb{R}$, we created GP models of $f$ using training data taken from the optimisation runs, evaluated a set of $N = 1000$ test locations, and calculated the model’s normalised root mean squared error (NRMSE).

More precisely, for a given test problem $f$, a GP model using the mean function to be evaluated was created using the first 100 locations (including initial LHS samples) for each of the mean function’s 51 optimisation runs using the EI acquisition function. A set of $N = 1000$ test locations $X = \{x_n\}_{n=1}^N$ were chosen via Latin hypercube sampling and the model’s prediction NRMSE between the true function values $f_n$ and predicted values from the surrogate model $\hat{f}_n$ were calculated for each of the 51 models. The NRMSE is defined as

$$\text{NRMSE} = \sqrt{\frac{1}{N} \sum_{n=1}^{N} (f_n - \hat{f}_n)^2} / f_{\text{max}} - f_{\text{min}},$$

where $f_{\text{max}}$ and $f_{\text{min}}$ are the largest and smallest values of the test locations evaluated using the real function. The test locations $X$ were paired across models corresponding to optimisation runs using the same training data so that a statistical comparison could be undertaken.

Table 4 shows the median NRMSE over the 51 GP models created using each mean function for the synthetic and robot pushing test problems. The method with the lowest median NRMSE on each function is highlighted in dark grey, and those highlighted in light grey are statistically equivalent to the best method according to a one-sided paired Wilcoxon signed-rank test with Holm-Bonferroni correction ($p \geq 0.05$). The RBF mean function provides the best (or equivalent to the best) modelling error for the evaluated test locations on 10 of the 12 evaluated functions. Interestingly, this does not correspond the same performance in Bayesian optimisation (recall Table 1 in the main paper). We suspect this is because the experiments are evaluating modelling of the entire function, whereas BO only requires the GP model to reflect the true function within the region that the minima reside.

Further work in quantifying the modelling error in BO could be carried out by only calculating the NRMSE within a small hypercube surrounding the global optimum. However, the size of the hypercube required to accurately calculate the modelling error of the important features for BO may vary between problems and therefore additional evaluations may be required of sets of test data sampled from hypercubes of different sizes.
| Function          | Arith | Med   | Min   | Max   | Lin   | Quad | RF   | RBF |
|-------------------|-------|-------|-------|-------|-------|------|------|-----|
| Branin            | 0.271 | 0.271 | 0.271 | 0.263 | 0.270 | 0.263 | 0.121 | 0.114 |
| Eggholder         | 0.217 | 0.165 | 0.216 | 0.177 | 0.161 | 0.163 | 0.198 | 0.166 |
| GoldsteinPrice    | 0.136 | 0.136 | 0.136 | 0.136 | 0.136 | 0.136 | 0.069 | 0.136 |
| SixHumpCamel      | 0.232 | 0.231 | 0.231 | 0.223 | 0.215 | 0.216 | 0.142 | 0.228 |
| Shekel            | 0.265 | 0.215 | 0.494 | 0.058 | 0.083 | 0.094 | 0.142 | 0.053 |
| Ackley            | 0.663 | 0.716 | 0.316 | 0.899 | 0.107 | 0.378 | 0.334 | 0.117 |
| Hartmann6         | 0.222 | 0.394 | 0.184 | 0.154 | 0.140 | 0.185 | 0.195 | 0.097 |
| Michalewicz       | 0.271 | 0.311 | 0.417 | 0.316 | 0.205 | 0.193 | 0.360 | 0.190 |
| Rosenbrock        | 0.362 | 0.368 | 0.371 | 0.270 | 0.257 | 0.151 | 0.355 | 0.157 |
| StyblinskiTang    | 0.325 | 0.347 | 0.360 | 0.151 | 0.267 | 0.199 | 0.220 | 0.168 |
| **PUSH4**         | 0.202 | 0.200 | 0.256 | 0.254 | 0.153 | 0.152 | 0.222 | 0.151 |
| **PUSH8**         | 0.205 | 0.202 | 0.238 | 0.168 | 0.136 | 0.129 | 0.191 | 0.125 |

Table 4. Median NRMSE modelling prediction error taken over across 51 GP models using the the 8 evaluated mean functions on the ten synthetic and two robot pushing functions. The models with the lowest NRMSE are shown in dark grey, with those statistically equivalent to them shown in light grey.