Check for updates

Received: 6 December 2023 | Revised: 25 June 2024 | Accepted: 11 September 2024

DOI: 10.1002/bit.28854

ARTICLE

BIOTECHNOLOGY AND BIOENGINEERING WILEY

Transfer learning Bayesian optimization for competitor DNA molecule design for use in diagnostic assays

Ruby Sedgwick[1,2] | John P. Goertz[1] | Molly M. Stevens[1,3] | Ruth Misener[2] | Mark van der Wilk[2,4]

[1]Department of Materials, Department of Bioengineering and Institute of Biomedical Engineering, Imperial College London, London

[2]Department of Computing, Imperial College London, London

[3]Department of Physiology, Anatomy and Genetics, Department of Engineering Science, Kavli Institute for Nanoscience Discovery, University of Oxford, Oxford, UK

[4]Department of Computer Science, University of Oxford, Oxford, UK

Correspondence
Mark van der Wilk, Department of Computer Science, University of Oxford, Oxford, UK.
Email: m.vdwilk@imperial.ac.uk

Funding information
Engineering and Physical Sciences Research Council; UK Research and Innovation, Grant/Award Number: EP/P016871/1; UKRI CDT in AI for Healthcare, Grant/Award Number: EP/S023283/1; BASF/RAEng Research Chair in Data-Driven Optimization; US National Institute of Health, Grant/Award Number: 5F32GM131594; EPSRC IRC Next Steps Plus, Grant/Award Number: EP/R018707/1; Royal Academy of Engineering Chair in Emerging Technologies, Grant/Award Number: CiET20214

Abstract

With the rise in engineered biomolecular devices, there is an increased need for tailor-made biological sequences. Often, many similar biological sequences need to be made for a specific application meaning numerous, sometimes prohibitively expensive, lab experiments are necessary for their optimization. This paper presents a transfer learning design of experiments workflow to make this development feasible. By combining a transfer learning surrogate model with Bayesian optimization, we show how the total number of experiments can be reduced by sharing information between optimization tasks. We demonstrate the reduction in the number of experiments using data from the development of DNA competitors for use in an amplification-based diagnostic assay. We use cross-validation to compare the predictive accuracy of different transfer learning models, and then compare the performance of the models for both single objective and penalized optimization tasks.

KEYWORDS
Bayesian optimization, design of experiments, DNA design, Gaussian processes, polymerase chain reaction, transfer learning

1 | INTRODUCTION

Tailoring biological sequences, such as oligonucleotides or proteins, for specific applications is a common challenge in bioengineering. These engineered molecules have a variety of uses including in biosensors (Deng et al., 2023; Goertz et al., 2023; Hua et al., 2022), medical therapeutics (Badeau et al., 2018; Blakney et al., 2019; Ebrahimi & Samanta, 2023) and bio-computing (Lv et al., 2021; Qian et al., 2011; Siuti et al., 2013). However, development often requires expensive or time consuming experiments, meaning good experimental design is necessary to optimize the biological sequences within the experimental budget (Cox & Reid, 2000). This also leads to better analysis, especially when there are interaction effects between input factors, which is common in biological experiments (Fellermann, 2019; Gilman et al., 2021; Kreutz & Timmer, 2009; Narayanan et al., 2020; Papaneophytou, 2019; Politis et al., 2017).

Iterative experimental designs have the advantage of using information from previous experiments to inform future ones. Bayesian

This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.
© 2024 The Author(s). *Biotechnology and Bioengineering* published by Wiley Periodicals LLC.

optimization is an iterative global black box optimization strategy (Shahriari et al., 2016; Snoek et al., 2012) which has proven effective for design of biomolecular experiments including vaccine production (Rosa et al., 2022), antibody development (Khan et al., 2023), design and manufacturing of proteins and tissues (Gamble et al., 2021; Mehrian et al., 2018; Narayanan et al., 2021; Romero et al., 2013), validation of molecular networks (Sedgwick et al., 2020) and extracellular vesicle production (Bader et al., 2023). In Bayesian optimization, a surrogate model, usually a Gaussian process, of the system is built using data and an acquisition function decides which data point to collect next. Gaussian processes are a powerful tool for designing biological experiments in low data regimes due to their uncertainty estimates (Hie et al., 2020).

When many similar biological sequences need to be designed, it can be harder to optimize all the sequences within the experimental budget. Optimizing each sequence from scratch discards useful information from previous tasks, meaning more experiments are required. An alternative is to use transfer learning—a technique that improves the learning of new sequences by using knowledge gained from other optimization tasks (Zhuang et al., 2021). Transfer learning is closely related to multitask learning, where information is shared between tasks that are optimized at the same time. The approach outlined here can be used for either, and we will use transfer learning as an umbrella term for both.

As we require our surrogate model to be data efficient and have uncertainty quantification, we consider four Gaussian process models: an average Gaussian process (AvgGP), the multioutput Gaussian process (MOGP), the linear model of coregionalization (LMC) and the latent variable multioutput Gaussian process (LVMOGP). The key difference between these Gaussian process models lies in their handling of correlations between outputs: from no correlation in the MOGP to nonlinear correlation in the LVMOGP.

We apply these surrogate models in conjunction with Bayesian optimization for efficient optimization of bio-molecules, as shown in Figure 1. We focus specifically on the development of a new modular diagnostic assay, based on competitive polymerase chain reaction (PCR), for measuring expression of multiple genes simultaneously, giving a single end point readout (Goertz et al., 2023). This diagnostic requires many competitor DNA sequences to be optimized to have the correct amplification properties in PCR reactions, and we believe the relationship between the responses of the competitors may be nonlinear. For optimal results, these competitors should have a predefined amplification curve rate; and a nuisance drift factor should ideally be below a certain threshold to allow for a more stable readout.

We use synthetic data experiments to compare the Gaussian process models in different settings. We then use cross-validation to verify the benefit of the LVMOGP for modeling the response of the

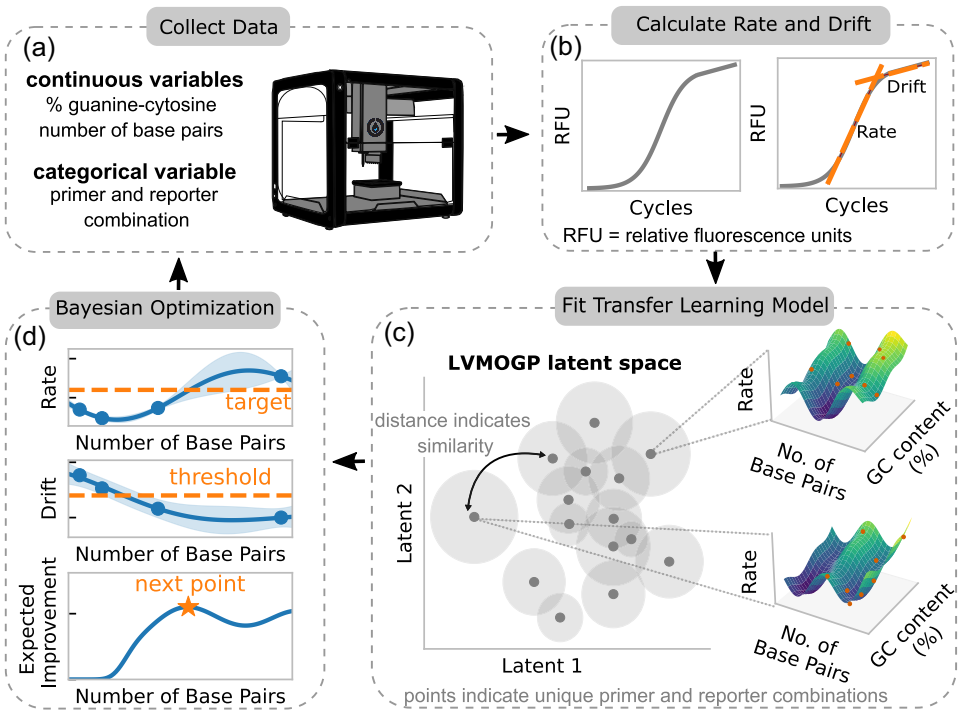


FIGURE 1 Design of experiments workflow for optimizing the competitor DNA molecules. (a) Data is collected in the lab using a DNA amplification reaction assay. (b) The rate and drift are then calculated by fitting amplification curves. (c) A transfer learning surrogate model is used the data to predict the rate and drift for each of the given competitors. The LVMOGP is introduced in Section 2.2.3. Information is shared through the latent space, with one point on the latent space for each competitor. The shaded regions indicate the uncertainty. The three-dimensional plots are predictions of the model for given competitors. (d) The Bayesian optimization algorithm, introduced in Section 2.3, combines information about the rate and drift surfaces in an acquisition function to select the experiment to run for each competitor. The solid lines in the rate and drift plots represent the mean of the Gaussian process models, while the shaded regions are 2 × standard deviation. This process is repeated until all optimal competitor sequences are found or the experimental budget is exhausted. LVMOGP, latent variable multioutput Gaussian process.

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

competitors, using data from DNA amplification experiments. We confirm that a LVMOGP surrogate model in conjunction with the design of experiments workflow speeds up optimization of the competitors both when only the single objective of rate is optimized and when rate is optimized with a penalty on drift over a given threshold.

2 | MATERIALS AND METHODS

2.1 | Gaussian process regression

A Gaussian process is a stochastic process representing an infinite collection of random variables, the joint distribution of any subset of which is a multi-dimensional Gaussian distribution (Rasmussen & Williams, 2006). A Gaussian process is fully defined by its mean $m : \mathbb{R}^D \mapsto \mathbb{R}$ and covariance $k : \mathbb{R}^D \times \mathbb{R}^D \mapsto \mathbb{R}$ functions:

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')), \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^D$ is our input. For a full nomenclature see Appendix A. We assume our output data $y(\mathbf{x}) \in \mathbb{R}$ to be noisy evaluations of $f(\mathbf{x}) \in \mathbb{R}$:

$$y(\mathbf{x}) = f(\mathbf{x}) + \epsilon, \tag{2}$$

where $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$ and σ_n^2 is the noise variance.

The choice of kernel and hyperparameter initializations for a given application depends on prior information about the system, see Appendix D for more details. Often this implies setting the mean function to zero, which is what we do here. A common kernel function is the squared exponential, which is a stationary kernel that assumes the data-generating function is smooth:

$$k(\mathbf{x}, \mathbf{x}') = \sigma_k^2 \exp\left(-\sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{2\ell_d^2}\right), \tag{3}$$

where σ_k^2 is the kernel variance and ℓ_d is the lengthscale of dimension d (Rasmussen & Williams, 2006, Chapter 4). Given a set of N training data $\mathcal{D} = \{(\mathbf{x}_i, y_i) | i = 1, \ldots, N\}$, the training inputs $\{x_n\}_{i=1}^N$ can be aggregated into the matrix $X \in \mathbb{R}^{N \times D}$ and the training observations $\{y_n\}_{i=1}^N$ aggregated into the vector $\mathbf{y} \in \mathbb{R}^N$. It is then possible to write a joint distribution of the training observations $\mathbf{y}$ and predicted function value $\mathbf{f}_*$ at prediction locations X_*. Thus, the mean and covariance of the Gaussian process at the prediction points can be calculated respectively:

$$\mu(X_*) = \mathbb{E}[\bar{\mathbf{f}}_* | X, \mathbf{y}, X_*] \tag{4}$$

$$= K(X_*, X)\left[K(X, X) + \sigma_n^2 \mathbf{I}\right]^{-1} \mathbf{y} \tag{5}$$

$$\sigma(X_*) = K(X_*, X_*) - K(X_*, X)[K(X, X) + \sigma_n^2 \mathbf{I}]^{-1} K(X, X_*). \tag{6}$$

The hyperparameters $\boldsymbol{\theta} = \{\sigma_n^2, \sigma_k^2, \ell_d\}$ are optimized by maximizing the marginal likelihood $p(\mathbf{y}|X, \boldsymbol{\theta})$, which is calculated in closed form (Rasmussen & Williams, 2006, Chapter 2).

2.2 | Gaussian processes with multiple outputs

2.2.1 | Independent Gaussian processes with shared kernel

The MOGP allows for multiple outputs such that $\mathbf{y} \in \mathbb{R}^{N \times P}$ (Álvarez et al., 2012). All outputs have the same kernel function and hyperparameters but function values on different outputs are uncorrelated. This means the kernel of the MOGP is a block diagonal with $k\left(X_p, X'_p\right) = k\left(X_p, X'_p\right)$ if $p = p'$ and $k\left(X_p, X'_p\right) = 0$ if $p \neq p'$ where p is the output index. The joint distribution for two outputs $\mathbf{f}_1$ and $\mathbf{f}_2$ evaluated at points X_1 and X_2 is given by:

$$\begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X_1, X_1) & \mathbf{0} \\ \mathbf{0} & K(X_2, X_2) \end{bmatrix}\right). \tag{7}$$

We use the MOGP to demonstrate the setting of no transfer of information about function values.

2.2.2 | Linear model of coregionalization

The LMC extends the MOGP to model linear correlations between output surfaces by assuming they are linear combinations of Gaussian process latent functions:

$$f_p(\mathbf{x}) = \mathbf{W}_p \mathbf{g}(\mathbf{x}) + \kappa_p u_p(\mathbf{x}). \tag{8}$$

where $\mathbf{W} \in \mathbb{R}^{P \times Q}$ is a vector of weights $\mathbf{g}(\mathbf{x}) = \{g_q(\mathbf{x})\}_{q=1}^Q$ are shared latent functions, $u_p(\mathbf{x})$ is a latent function that allows for some independent behavior and κ_p is a learned constant (Álvarez et al., 2012; Bonilla et al., 2007).

This leads to a Kronecker structured kernel such that the joint distribution between two functions $\mathbf{f}_1$ and $\mathbf{f}_2$ is given by:

$$\begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \sum_{q=1}^{Q} b_{11} k_q(X_1, X_1) & \sum_{q=1}^{Q} b_{12} k_q(X_2, X_2) \\ \sum_{q=1}^{Q} b_{21} k_q(X_1, X_1) & \sum_{q=1}^{Q} b_{22} k_q(X_2, X_2) \end{bmatrix}\right), \tag{9}$$

where $b_{pp'}$ is an element of $B = WW^T + \text{diag}(\boldsymbol{\kappa})$, a $P \times P$ matrix determining the similarity between functions and there are Q different covariance functions $k_q(\mathbf{x}, \mathbf{x}')$. If $Q = 1$, this is known as the intrinsic coregionalization model (Álvarez et al., 2012).

Coregionalization methods have successfully been used for Bayesian optimization (Cao et al., 2010; Swersky et al., 2013; Tighineanu et al., 2022) and applied to the optimization of synthetic genes (González et al., 2015) and chemical reactions (Taylor

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

et al., 2023). However, coregionalization methods assume the response surfaces are linear combinations of a small number of latent functions, so they can fail to fit and predict well on data with nonlinear similarity between surfaces.

2.2.3 | Latent variable multioutput Gaussian process

The LVMOGP introduced by Dai et al. (2017) can model nonlinear similarities. It does so by augmenting the input domain of a Gaussian process with a Q_H dimensional latent space $\mathcal{H}$. Each output function has a latent variable, such that the latent variables are denoted by $H = [\mathbf{h}_1, ..., \mathbf{h}_P]^T \in \mathbb{R}^{P \times Q_H}$. The LVMOGP assumes output $\mathbf{y}_p$ is generated by:

$$y_p(\mathbf{x}) = f(\mathbf{x}, \mathbf{h_p}) + \epsilon, \tag{10}$$

where $\epsilon \sim \mathcal{N}(0, \sigma_n^2 \mathbf{I})$. The latent space allows the LVMOGP to automatically transfer learn between output functions as it will cluster similar output functions together and place wildely different ones far apart on the latent space. The distance on the latent space and the latent space lengthscale determines the amount of correlation between different output functions. To account for uncertainty in the placement of the latent variables, they are treated as distributions rather than point estimates, such that $\mathbf{h_p} \sim \mathcal{N}(\boldsymbol{\mu}_{h_p}, \Sigma_{h_p})$. For more details on the implementation of the LVMOGP see Appendix B.

Similar latent variable models have been used for Bayesian optimization of material development (Zhang et al., 2020) and for transfer learning across cell lines (Hutter et al., 2021). However, these methods treat the latent variables as point estimates rather than distributions as in the LVMOGP, which can cause poor uncertainty estimates, especially at low data regimes.

2.2.4 | Comparison of Gaussian process models

In our comparisons, we include a fourth model called the average Gaussian process (AvgGP), which treats all the data as if it has come from the same response surface. Figure 2 shows predictions of the four Gaussian process models on a toy data set with linear correlation between output functions. See Appendix C for details of the data generation. As the AvgGP doesn't differentiate between surfaces, it doesn't fit any response surface well. The MOGP only shares hyperparameters but no information about function values between response surfaces, meaning it makes worse predictions and has more uncertainty on new response surfaces. The LMC has a better mean prediction than the MOGP as it shares information between response surfaces. The LVMOGP similarly has better mean prediction than the MOGP as it shares information across response surfaces through the latent space. If $Q = 1$ and B is the identity matrix, then the LMC recovers the MOGP. If a linear kernel is applied to the latent dimensions of the LVMOGP, the LMC is recovered, and by making the distance between latent variables large

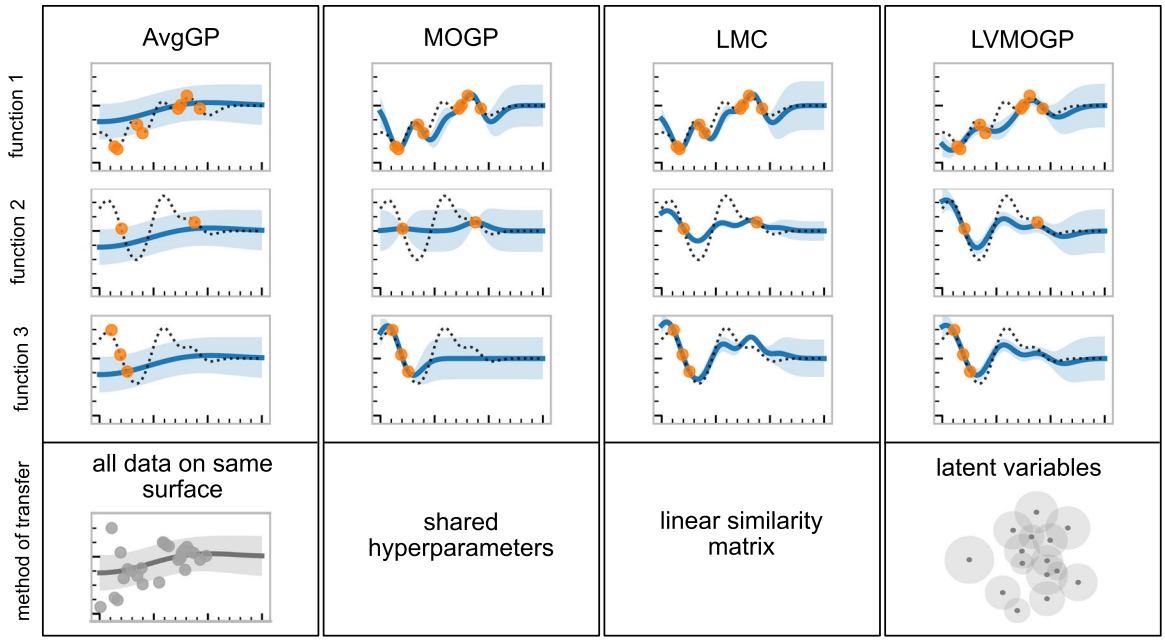


FIGURE 2 Predictions of the four Gaussian process models fitted to a toy data set with linear correlation between output surfaces. The dots are the data, the dashed line is the true function, the solid line is the Gaussian process mean prediction and the shaded region is two times the predicted standard deviation, meaning around 95% of the data points should lie within the shaded region. The bottom row explains how data is transferred between the surfaces by each model. For the average Gaussian process (AvgGP), all data is assumed to be from the same surface, for the multioutput Gaussian process (MOGP) information is only transferred about the hyperparameter values but not the function values. In the linear model of coregionalization (LMC) information is transferred via the similarity matrix **B** and in the latent variable multiouput Gaussian process (LVMOGP) it is transferred through the latent space. Theoretically, LMC and LVMOGP can learn if information can be transferred and (if so), how much.

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

relative to the lengthscale, the MOGP can be recovered too. The fact there are hyperparameter settings for the LMC and LVMOGP that recover the MOGP is promising for preventing negative transfer, as in the case where there is no correlation between response surfaces they can just revert to the MOGP. However, this is only true for large data sets—in low data regimes, we may expect some negative transfer in the no correlation case, due to uncertainty in the hyperparameter values and, in the case of the LVMOGP, a prior on the existence of correlations.

2.2.5 | Gaussian process implementation

Details of the data processing and hyperparameter initializations can be found in Appendix D. All coding was done in Python using version 3.9. The Gaussian process models were implemented using GPFlow 2.3.0 (Matthews et al., 2017). GPFlow has implementations of the standard Gaussian process, MOGP and the LMC. Our LVMOGP was implemented as a new GPflow model class, which can be accessed via the Github links in Appendix E. Other packages used include PyMC3 3.11.4 (Salvatier et al., 2016) for Bayesian parameter estimation, Numpy 1.21.4 (Harris et al., 2020), Scipy 1.7.1 (Virtanen et al., 2020) and Pandas 1.3.4 (The pandas development team, The pandas development, 2023) for data processing and Matplotlib 3.4.3 (Droettboom et al., 2015) for visualization.

2.3 | Bayesian optimization

Bayesian optimization is a sequential experimental design strategy for finding the global minimum (or maximum) of an objective function (Shahriari et al., 2016; Snoek et al., 2012). As the objective function is unknown, a surrogate model is used to represent the posterior belief of the objective function and updated every time a new data point is observed. An acquisition function is then used to select the next data point to collect. A common acquisition function is the expected improvement which trades off exploration of regions with little data and exploitation of regions which are expected to be optimal (Garnett, 2023; Jones et al., 1998). This process is repeated until the optimum has been found or the experimental budget is exhausted.

2.3.1 | Acquisition function

Rather than maximizing or minimizing the rate, as is usual in Bayesian optimization, we wish to minimize the difference between the rate, f_{rate}, and the target rate, T_{rate}:

$$\arg\min_{BP,GC} \sqrt{(f_{\text{rate}} - T_{\text{rate}})^2}. \tag{11}$$

Therefore, we use the target vector optimization acquisition function, that extends the expected improvement acquisition function to minimize the Euclidean distance between a target vector and a vector of the current predicted values (Uhrenholt & Jensen, 2019). As we are only optimizing the rate, we use their formulation with scalars instead of vectors. In this formulation, a stochastic variable is defined as $\boldsymbol{\delta}|\mathbf{x} = \|y_{\text{rate}}(\mathbf{x}) - T_{\text{rate}}\|_2^2$ where $y_{\text{rate}}(\mathbf{x})$ is the output value at input $\mathbf{x}$ and T_{rate} is our target value. The distribution of $p(\boldsymbol{\delta}|\mathbf{x})$ is modeled with the aim of minimizing δ. If the response surfaces are Gaussian processes, then $p(\boldsymbol{\delta}|\mathbf{x})$ can be approximated using a noncentral χ^2 distribution (Uhrenholt & Jensen, 2019). The expected improvement for this noncentral χ^2 distribution is expressed as:

$$\begin{aligned}\alpha_{EI} = {} & \boldsymbol{\delta}_{min} G_\lambda(\boldsymbol{\delta}_{min}/\gamma^2) \\ & -\gamma^2 \mathbb{E}[t|t < \boldsymbol{\delta}_{min}/\gamma^2] G_\lambda(\boldsymbol{\delta}_{min}/\gamma^2),\end{aligned} \tag{12}$$

where $\boldsymbol{\delta}_{min}$ is the minimum $\boldsymbol{\delta}$ observed so far, γ is root mean of the variances of each output evaluated at the training points, $t = \boldsymbol{\delta}\gamma^{-2}$, and G_λ is an approximate cumulative χ^2 distribution with non-centrality parameter λ defined in the paper (Uhrenholt & Jensen, 2019).

2.3.2 | Bayesian optimization with drift penalty

To ensure the drift value remains below, or close to the threshold, we use the *probability of feasibility* to encourage the algorithm to select points that have a high chance of being below the threshold (Schonlau et al., 1998):

$$PF(\mathbf{x}) = p(f_{\text{drift}}(\mathbf{x}) \le T_{\text{drift}}), \tag{13}$$

where $f_{\text{drift}}(\mathbf{x})$ is the value of drift function at $\mathbf{x}$, and T_{drift} is the drift threshold. We then multiply the expected improvement by the probability of feasibility to get our final acquisition function:

$$\alpha_c = PF(\mathbf{x})\alpha_{EI}(\mathbf{x}). \tag{14}$$

The probability of feasibility has been used for optimization applications including analog circuits (Lyu et al., 2018) and materials design (Sharpe et al., 2018).

2.3.3 | Performance metrics

For both the synthetic experiments and the cross-validation experiments we assessed the fit of Gaussian process models with two performance metrics: root mean squared error (RMSE):

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N^*}\left(\mu(\mathbf{x}_i^*) - y_i^*\right)^2}{N^*}}, \tag{15}$$

and negative log predictive density (NLPD):

$$\begin{aligned}\text{NLPD} &= \frac{1}{N^*}\sum_{i=1}^{N^*} \log p(y_i^*|\mathbf{x}_i^*, \mathbf{X}, \mathbf{y}, \boldsymbol{\theta}) \\ &= -\frac{1}{2N^*}\sum_{i=1}^{N^*}\left(-\log(2\pi\sigma(\mathbf{x}_i^*)^2) - \frac{\left(y_i^* - \mu(\mathbf{x}_i^*)\right)^2}{\sigma(\mathbf{x}_i^*)^2}\right).\end{aligned} \tag{16}$$

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

These are both calculated on a test set of input locations X^* of length N^*. The RMSE is useful for comparing the mean predictions of the Gaussian processes while the NLPD also indicates how good the uncertainty estimate is, both of which are important for effective exploration and exploitation. For assessing the Bayesian optimization algorithm, we use cumulative regret:

$$\text{regret} = \min_{i \in [1..N]} \left(\sqrt{(y_{\text{rate},i} - y_{\text{best}})^2} + \max(0, y_{\text{drift},i} - T_{\text{drift}}) \right), \quad (17)$$

where $y_{\text{rate},i}$ and $y_{\text{drift},i}$ are rate and drift training data, y_{best} is the data point closest to the target out of both training and candidate sets for that surface and $\max(0, (y_{\text{drift},i} - T_{\text{drift}})$ is a penalty for exceeding the drift threshold.

2.4 | Data collection

Each competitor has predefined primers and fluorescent probes and a design region where the sequence can be altered. Rather than tackling the difficult combinatorial problem of optimizing the sequence directly, we reduce the problem to two key input variables: the number of base pairs (BP) and guanine–cytosine content (GC) as in Figure 3. This converts the design space into a more manageable continuous form and reduces the input dimensions, which is beneficial when data is limited. For each BP–GC combination, chosen by an expert researcher, a PCR assay generates an amplification curve, from which rate and drift are calculated. In total, we have data on 34 different competitors and wish to optimize 16 of these. Across the 34 competitors, we have 592 data points at 327 unique input locations, with 1–6 repeats at each location. See Appendix F for a summary of the data.

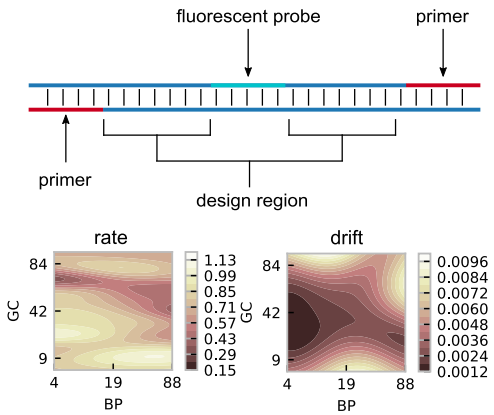


FIGURE 3 Schematic of the competitor design space. For a given competitor DNA molecule, the primers and fluorescent probe regions are fixed. We can edit the design region to ensure the sequence has a given number of base pairs and guanine-cytosine content. Changing the number of base pairs and guanine-cytosine-content affects the rate and drift of the competitor, allowing us to fine-tune to the rate and drift required for the diagnostic assay.

The rate and drift for each amplification curve were calculated using the following equations:

$$F_T = \frac{v}{1 + \frac{(v - F_0)}{F_0} \cdot e^{-r \cdot \tau}}, \quad (18)$$

$$\text{signal} = F_T \cdot \left(1 + \frac{F_T}{v} \cdot m \cdot (\ln(F_0)/r)\right), \quad (19)$$

where F_T and F_0 are the end point and starting fluorescence, v is carrying capacity, r is the rate, m is the drift and τ is cycle number.

2.4.1 | Polymerase chain reactions

To perform the PCRs, we used an Applied Biosystems QuantStudio 6 Flex using Applied Biosystems MicroAmp EnduraPlate Optical 384-well plates (Thermo Fisher Scientific). The theromcycling stages consisted of a melt step at 95°C for 3 s and an annealing step at 60°C. All reactions were performed at 10 µL and used Applied Biosystems TaqMan Fast Advanced Master Mix. Either fluorescent probes or EvaGreen dye (Biotium) were used as reporters.

2.4.2 | DNA sequences

For each BP–GC combination for a given competitor, NUPACK (Zadeh et al., 2011) was used to create a DNA sequence with the correct number of base pairs and guanine-cytosine content, as well as the correct sequences for the primer and probes. These sequences, alongside synthetic natural target analogs, were purchased from Twist Biosciences or as eBlock Gene Fragments from Integrated DNA Technologies ("IDT"). Primers and probes were also purchased from IDT.

3 | RESULTS

3.1 | Synthetic data experiments

To explore the performance of the MOGP, AvgGP, LMC, and LVMOGP, we ran experiments on synthetic data sets representing three test cases: uncorrelated, linearly correlated and horizontally offset response surfaces. All synthetic experiments had two response surfaces each with 30 points observed and 10 new response surfaces with no points observed initially. We added one random point to each new response surface every iteration and recorded the RMSE and NLPD for the Gaussian process models' predictions. Figure 4 shows the RMSEs and NLPDs of the Gaussian process models for these test settings.

For the uncorrelated test case, response surfaces were generated as independent samples of a Gaussian process prior with a $\ell = 0.3$ and $\sigma_k^2 = 2$. This test case was to check for negative transfer, where the sharing of information hinders rather than aids the learning

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

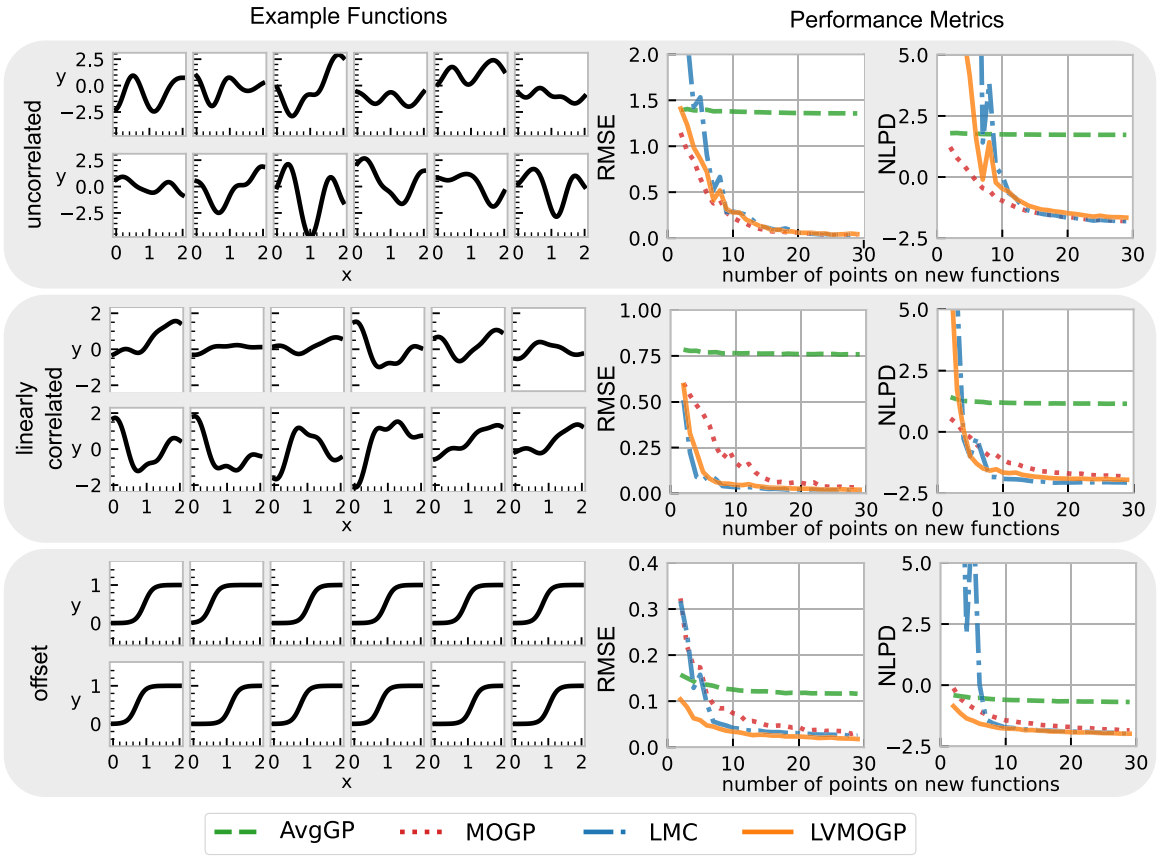


FIGURE 4 Results of experiments with synthetically-generated data. The plots on the left show example data-generating functions used for the synthetic experiments. The plots on the right show the RMSE and NLPD for the three different test response surface types for each of the Gaussian process models. New points are added randomly, and each line is the mean of five different randomly generated data sets, all generated from the same test functions. AvGP, average Gaussian process; LMC, linear model of coregionalization; LVMOGP, latent variable multiouput Gaussian process; MOGP, multiouput Gaussian process; NLPD, negative log predictive density; RMSE, root mean squared error.

process. In Figure 4, for the uncorrelated case, the MOGP outperforms the other Gaussian process models for RMSE and NLPD until approximately 10 data points, although the RMSE of the MOGP at this point is still high. We expect the LMC and LVMOGP to have some negative transfer at very low data regimes as they have a prior expectation of correlations between response surfaces. However, with enough data, they should perform the same at the MOGP, which is corroborated by the results in Figure 4. Specifically, once the MOGP gets a reasonably low RMSE of <0.25 the LMC and LVMOGP have achieved similar performance.

The response surfaces for the linearly-correlated test case were created as linear combinations of two latent functions, both generated as independent samples of a Gaussian process with squared exponential kernel, Equation (3), and $\ell = 0.3$ and $\sigma_k^2 = 2$. The LMC outperforms the other two Gaussian process models except at very low data regimes, which is likely due to overconfidence of the LMC when it has little data. The LMC and LVMOGP outperform the MOGP even at high data regimes, showing the advantage of transfer learning.

The horizontally offset test case was chosen as a simple example where the LMC struggles to fit the data. The response surfaces were generated by offsetting a sigmoid function horizontally by a random constant. In this case, the LVMOGP outperforms the other Gaussian process models for both RMSE and NLPD. This is because the LVMOGP can learn new surfaces with very few data points, as all it needs to do is to correctly predict where the sloped region is. The LMC performs worse than the LVMOGP because the offset cannot be represented by a linear combination of its latent functions, meaning it requires more data to perform as well.

Across all the test cases, the LMC has poor NLPD at low data regimes. This is likely because it cannot express uncertainty in the deterministic **B** matrix.

3.2 | Prediction of DNA amplification experiments

The performance of the proposed design of experiments workflow was validated using data from competitor DNA amplification experiments. This was done in three parts: first cross-validation was performed to compare the predictive accuracy of the Gaussian process models; then a Bayesian optimization procedure was used to optimize only the rate; finally the Bayesian optimization with drift penalty procedure was applied.

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

In cross-validation, the training set consisted of all the data from the two competitors that had the most observations as well as a random subset of the remaining data, but ensuring all competitors had at least one data point. This was repeated 70 times for each percentage of data in the training set. We set both the rank of **B** for the LMC and the latent dimensions of the LVMOGP to 10, see Appendix D.5 for a discussion on setting these parameters. Figure 5 shows the RMSE and NLPD of the Gaussian process models' predictions. The LVMOGP outperforms the other Gaussian process models for both RMSE and NLPD for both rate and drift. The LMC has poor NLPD in comparison to the other Gaussian process models, suggesting it has poor uncertainty estimates. The AvgGP model shows little improvement with increased amounts of training data. This shows the limitations of averaging the surfaces and justifies modeling each response surface separately.

3.3 | Optimization of DNA amplification experiments

Ideally, for the Bayesian optimization experiments we would integrate the algorithm into the experimental loop, collecting new data with each new recommendation of each Gaussian process model. However, due to the cost of experiments, this was infeasible. Instead, we performed retrospective Bayesian optimization using the existing competitive DNA amplification data set. The data was split into training and candidate sets, with the design of experiments algorithm only allowed to choose the next point out of the candidate set. Bayesian optimization was run iteratively until all points had been selected or up to a maximum number of iterations, whichever happened first.

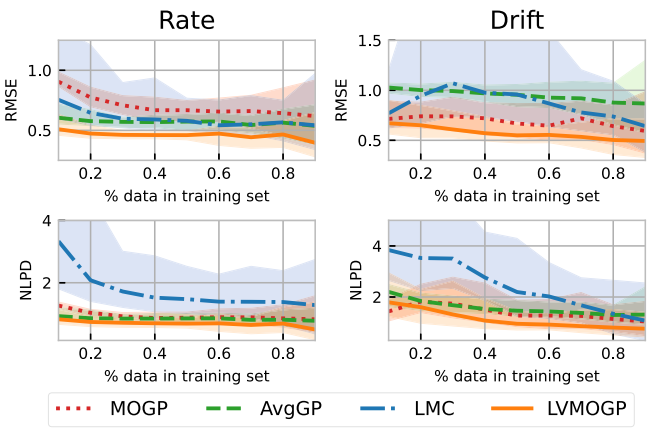


FIGURE 5 Results of cross-validation on the DNA amplification data for both rate and drift. For each cross-validation run, the training set consisted of all the data from two competitors and a random subset of the data on the remaining competitors, ensuring all competitors had at least one data point. This is repeated for different percentages of data in the training set, and for each percentage, it is repeated 70 times. AvGP, average Gaussian process; LMC, linear model of coregionalization; LVMOGP, latent variable multiouput Gaussian process; MOGP, multiouput Gaussian process.

Two learning scenarios were tested: the *learning many* scenario where all data from the two competitors with the most data were fully observed to begin with and then 16 competitors optimized in parallel; and the *one at a time* where each of the 16 competitors was optimized individually, with the 33 remaining competitors included in the training set. For a discussion of the effect of the choice of initial surfaces, see Appendix G.5. These scenarios replicate likely lab experimentation scenarios—the first for when many competitors need to be optimized at once, and the second for when many competitors are already optimized and an extra one is added. The maximum number of iterations was 15 for the rate-only optimization and 20 or 10 for the penalized optimization, depending on the learning scenario.

We also considered two methods for choosing the first experiment for a new competitor with no previously observed data. Choosing the most central data point (*center* in Figure 6) offers both maximum reduction in variance across the response surface and ensures all competitor response surfaces have a comparable point, which may help the transfer learning methods determine their similarities. It is also a reasonable approximation of what a human experimenter without prior knowledge of the response surface might do. The second method is to let the Gaussian process model choose the first point (*model's choice* in Figure 6) for a new competitor. For the AvgGP and the LVMOGP, this is possible as they can make posterior predictions on new response surface. For the LVMOGP, the latent variable of the new surface is determined as a weighted average of the latent variables of the response surfaces with data that have the same probe and at least one matching primer. If there are no surfaces with matching primers, we use a weighted average of the surfaces with the same probe. For the LMC and MOGP we have no posterior, so the first point is selected randomly.

For these experiments, we set the values of the latent hyperparameters for the LMC and LVMOGP to 2, as discussed in Appendix D.5.

3.3.1 | Single objective Bayesian optimization

The left panel of Figure 6 shows the results of optimizing rate without considering the drift penalty. The variance in the results comes from three sources. The first is the random selection of the next point when two points have the same expected improvement—this causes unavoidable variation. The second is due to the Gaussian process models optimizing to different hyperparameter values due to different initializations. The different values arise because the optimization of the nonconvex hyperparameter loss surfaces is difficult. The final source of variation is the random starting point for the MOGP and LMC.

In all cases, Figure 6 shows the LVMOGP has much lower cumulative regret than the other models. The LVMOGP also reaches the best point first the most often: 808 times across all competitors, all learning scenarios and seeds compared to 457, 498, and 484 for the MOGP, AvgGP and LMC respectively. See Appendix G: Table G1

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

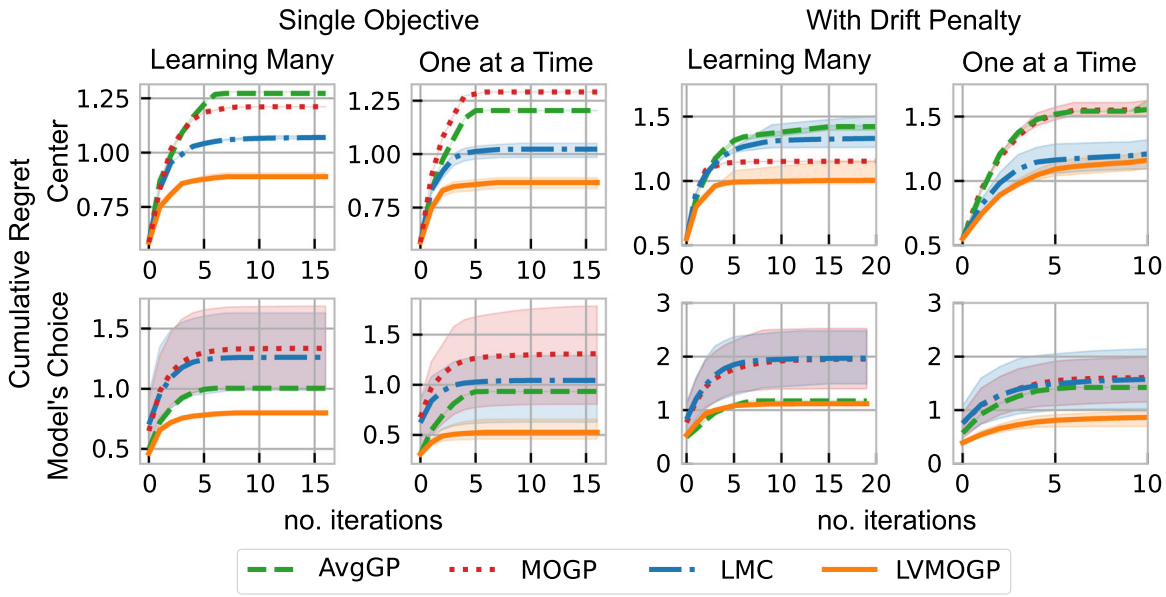


FIGURE 6 Cumulative regret of each of the Gaussian process models for single objective (left) and penalized (right) Bayesian optimization. Each line indicates the mean across 24 random seeds and all competitors, while the shaded regions indicate the upper and lower 5% quantiles by random seed. The top row is when the first point on each new surface is selected as being the center point, and the bottom is when the model is allowed to choose the first point. The *learning many* scenario is when many competitors are being optimized at the same time, and the *one at a time* scenario is when one competitor is being optimized, with all others being in the training set. AvGP, average Gaussian process; LMC, linear model of coregionalization; LVMOGP, latent variable multiouput Gaussian process; MOGP, multiouput Gaussian process.

for a breakdown of these results. The *center* start point allows us to compare the performance of the Gaussian process models without being skewed by the first point. In this case, the LMC and LVMOGP have the lowest cumulative regret, with mean values of 1.08 and 0.91, respectively at the end of optimization, compared to 1.21 and 1.28 for the MOGP and AvgGP for the *learning many* case. The ordering changes between the *center* and *model's choice* starting points, as in the latter the AvgGP and the LVMOGP are able to predict on new surfaces, giving them an advantage over the LMC and the MOGP when choosing the first point. For example, in *learning many model's choice* scenario, the mean regret of the first points selected by the LVMOGP and the AvgGP are 0.464 and 0.499 respectively compared to 0.651 and 0.703 for the MOGP and the LMC. Table 8 in Appendix G.3 lists the mean regrets of the first points.

As the *one at a time* scenario includes the data from all other competitors, the Gaussian process models start with far more data than the *learning many* scenario. This means the AvgGP, the LMC and the LVMOGP all have less regret in the *one at a time* scenario, as they are able to transfer information about the function values of competitors to improve prediction of the target competitor behavior. This is most notable for the *model's choice* start point, where the AvgGP, LMC and LVMOGP have final cumulative regrets of 0.93, 1.29, and 0.66, respectively, compared to 1.00, 1.63, and 0.80 for the *learning many* scenario. The MOGP does not transfer information about function values, so performs relatively worse than the other models for the *one at a time* scenario, with a final cumulative regret of 1.78 for the *one at a time* scenario as opposed to 1.69 for the *learning many* scenario.

Across all learning scenarios and start points, the LVMOGP has the smallest mean number of iterations to get within a tolerance of 0.05 of the value of the best point, with the LVMOGP taking a mean of 2.25 iterations, while the AvgGP, MOGP and LMC take 2.89, 3.02 and 2.93 respectively. For 16 competitors, this equates to 36 experiments needed for the LVMOGP compared to 49 for the MOGP. See Appendix G.1 for a break down by learning scenario and starting point and Appendix G.4 for box plots of the number of experiments taken by each model. The tolerance was set at 0.05 as this is approximately the level of experimental measurement uncertainty in the lab experiments Goertz et al. (2023).

3.3.2 | Bayesian optimization with drift penalty

The right-hand panel of Figure 6 shows the cumulative regret for optimization of the rate with a penalty on the drift. The LVMOGP has the lowest cumulative regret at the end for all scenarios, but doesn't outperform the other models as much as in the single objective case. In all scenarios, the MOGP, LMC, and LVMOGP fail to reach the best point for the same competitor, meaning the cumulative regret curves for these models don't completely plateau. This is because they overestimate the value of the drift at the best point, so avoid selecting it. The AvgGP does find the best point for all competitors. The LVMOGP barely outperforms the AvgGP for the *learning many* scenario with *model's choice* starting point. This may be due to negative transfer in the drift predictions at very low data regimes making the selection of the first point suboptimal.

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

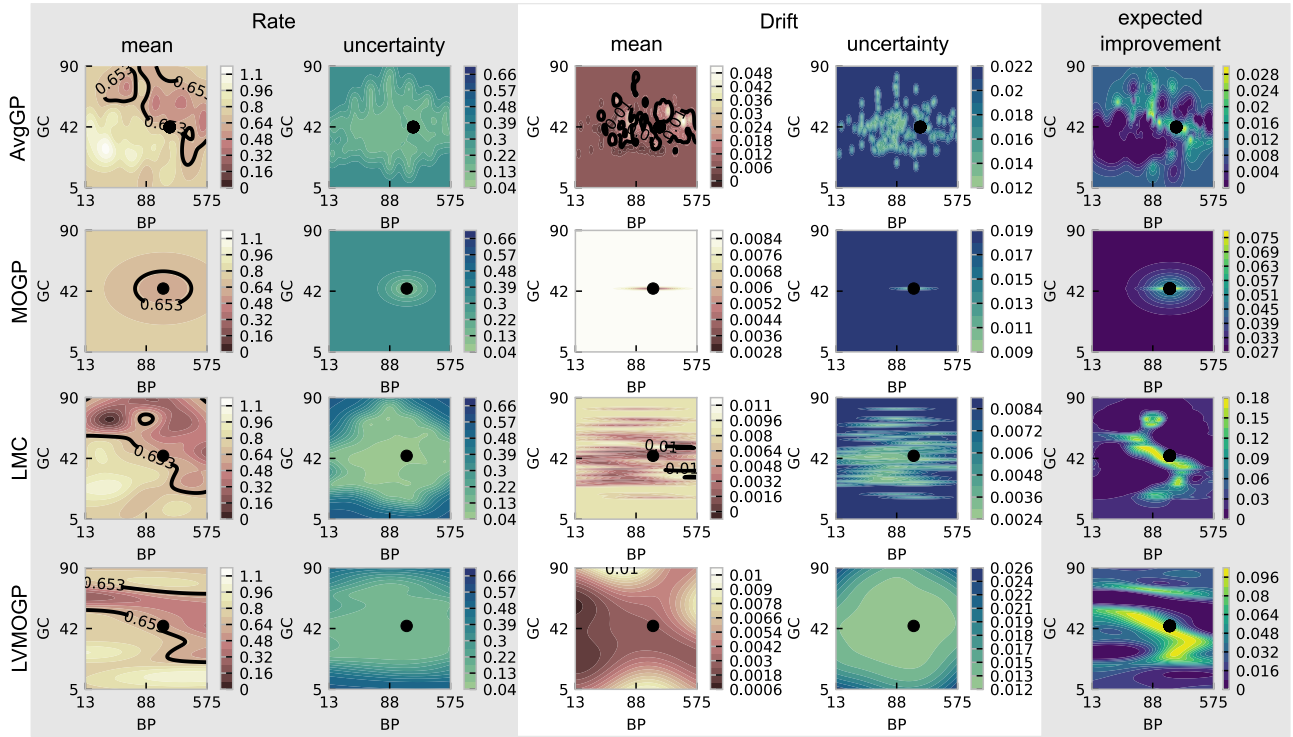


FIGURE 7 Predictions for the rate and drift for each of the Gaussian process models. The BP and GC axes are in log and logit scales respectively. These plots show the mean of the Gaussian process model predictions and the uncertainty which here is 2 × standard deviation. The expected improvement with probability of feasibility is then plotted in the final column. This is for the case where we are optimizing competitor FP005-FP004-EvaGreen and have observed one data point so far, with the models able to choose the first point. The black contour lines on the mean plots indicate the target rate and threshold drift values. BP, number of base pairs; GC, percentage guanine–cytosine content.

The ordering of the Gaussian process models is different for the *learning many* and *one at a time*, probably because the increased amount of data allows the LMC to predict comparatively better in the *one at a time* scenario than the *learning many*. The LVMOGP outperforms the other Gaussian process models the most in the *one at a time model's choice* experiment, which is likely due to the large amounts of data on all competitors, except the target, and effective transfer of information between them.

Similar to the single objective case, the LVMOGP has the smallest mean number of iterations to get within a tolerance of 0.05 of the value of the best point, with a mean of 2.26 iterations compared to 3.38, 3.54, and 3.39 for the AvgGP, the MOGP and the LMC. For 16 competitors, this equates to 37 experiments needed for the LVMOGP compared to 57 for the MOGP, see Appendix G.4 for more details. Appendix G contains further Bayesian optimization results for both the single objective and penalized optimizations.

Figure 7 shows the rate and drift predictions and expected improvement for one iteration. Most notably, the MOGP has no transfer of information, so has almost equal expected improvement for most of the candidate points. The other three models transfer information across the competitors, meaning even with one data point, they have much more complex predictions than the MOGP. We can also see how the AvgGP, MOGP and LMC fit the drift poorly. This is because the drift is of a different order of magnitude

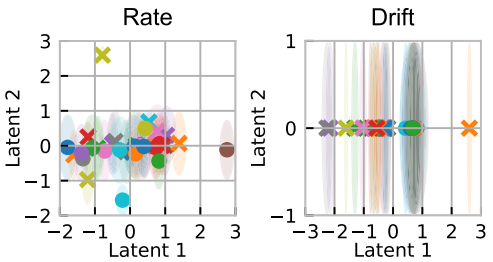


FIGURE 8 Latent space of the LVMOGP for the rate and drift. The crosses indicate competitors with probe primers and the dots indicate those with EvaGreen primers. The shaded circles indicate the uncertainty in the latent positions. LVMOGP, latent variable multiouput Gaussian process.

depending on the fluorescent probe used. Most of the Gaussian process models are unable to detect this, meaning they end up with a poor fit to the data. Figure 8, however, shows how the LVMOGP clusters the competitors in the drift latent space by the type of probe used, with crosses for probe primers and dots for EvaGreen primers. This indicates it has recognized there are two regimes for drift, despite not being explicitly told which probe a competitor uses. This improves predictive power, as for any given competitor the LVMOGP will use more information from competitors close by in the latent space, which are likely to be those with the same primer.

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

4 | DISCUSSION

Expensive and time consuming experiments require an intelligent design of experiments strategy. This study demonstrates how a transfer learning surrogate model can be used in conjunction with Bayesian optimization to optimize biological sequences. For the specific case of designing competitor DNA molecules for a new diagnostic, reducing the number and therefore cost of experiments can help it reach the affordability criteria for point of care settings (Land et al., 2019).

In Bayesian optimization, we need a surrogate function with reliable mean and uncertainty estimates to ensure a balance between exploration and exploitation when selecting new points. Our cross-validation results in Section 3.2 show the LVMOGP has better predictive accuracy than the other Gaussian process models for both rate and drift. These results also demonstrate one of the limitations of the LMC: the LMC has very high NLPD at low data regimes. This implies the LMC has poor uncertainty estimates and is overfitting, a result which has been previously observed (Dai et al., 2017).

To replicate a real-life iterative design of experiments regime, we performed Bayesian optimization on DNA amplification experimental data, but only allowing the models to select new points from existing data. For the single objective optimization case, the LVMOGP has lower cumulative regret than the other Gaussian process models for all test cases and starting points and requires fewer experiments on average to get within 0.05 tolerance of the best point. Specifically, the LVMOGP requires 13 and 20 fewer experiments than the no transfer MOGP model for the single objective and penalized cases respectively. This shows the LVMOGP transfer learning approach is useful both when optimizing multiple competitors at a time, and when using the data from all previous competitors to optimize a new one. These results also demonstrate the advantage of a surrogate model that can predict unseen surfaces—both the LVMOGP and the AvgGP see a large improvement in regret when they are allowed to select the first point, both outperforming the MOGP and LMC where the first point is chosen at random.

When optimizing new biological sequences, there are often factors we wish to keep within a certain range such as purity (Degerman et al., 2006) or biophysical properties (Khan et al., 2023). While these can be treated as constraints, sometimes we may be willing to violate them slightly if it leads to a large improvement in the objective function. In these scenarios, we can add a penalty. To apply a penalty on the nuisance drift factor, we used probability of feasibility to penalize any point predicted to be above the threshold drift value. In the penalized optimization, the LVMOGP had less cumulative regret than the other models but the difference in performance was smaller than that of the single objective optimization. This could be due to the added challenge of dealing with the penalty on drift.

There is variation in the performance of the Gaussian process models across random seeds due to the hyperparameter initialization. The LVMOGP has more variation due to its training being a harder optimization problem. While smart initialization and random restarts helped with this issue, future work could simplify the optimization procedures. The optimization of the Gaussian process models is discussed in Appendix D.3.

While the workflow outlined here will be useful for the optimization of new competitor DNA molecules, it is not specific to this application and could be used for other applications where it is necessary to optimize many similar tasks, such as engineering DNA probes (Lopez et al., 2018; Wadle et al., 2016), exploring protein fitness landscapes (Hu et al., 2023), optimizing conditions for different cell lines (Hutter et al., 2021), or inferring psuedotime for cellular processes (Campbell & Yau, 2015). We expect this method will scale well to settings with more output surfaces, and predictions will improve with more data. However, as with most Bayesian optimization approaches, it will not scale as well to high dimensional input settings (Wang et al., 2023).

We opted to use the LVMOGP to demonstrate how we can transfer information between tasks using proximity in latent space as Gaussian processes are data efficient and give good uncertainty predictions. However, we could replace Gaussian processes with any Bayesian model that gives priors over functions such as deep Gaussian processes (Damianou & Lawrence, 2013) or Bayesian neural networks (Goan & Fookes, 2020).

With the rise in lab automation, this workflow can be integrated into a design build test pipeline similar to Carbonell et al. (2018) and HamediRad et al. (2019) which can greatly reduce the time required to optimize new biomolecular components, speeding up the creation of new devices. This method could also be incorporated into hybrid models in bio-processing and chemical engineering, for decision making for systems with many similar components (Mowbray & Savage, 2021; Narayanan et al., 2023; Schweidtmann et al., 2021).

This workflow could also be extended to multi-output optimization problems by using a multi-output acquisition function or by finding Pareto optimal solutions (Belanger et al., 2019; Jablonka et al., 2021; Olofsson et al., 2018; Schweidtmann et al., 2018; Selega & Campbell, 2022). Similarly, the surrogate functions needed for multi-fidelity learning, where sources of information with varying fidelity are available, are similar to those for transfer learning, making it an easy extension (Folch et al., 2023; Sun et al., 2022).

5 | CONCLUSION

We have shown how a transfer learning design of experiments workflow can be used to optimize many competitor DNA molecules for an amplification-based diagnostics device. We used cross-validation to demonstrate that the latent variable multi-output Gaussian process has the best predictive accuracy and have shown it has the least regret when Bayesian optimization is performed on the DNA amplification data. Future improvements to the optimization of the model hyperparameters would lead to faster and more consistent performance of the algorithm. Despite this, we believe this workflow is applicable to many other biotechnology applications and should be used to reduce the experimental load when there are many similar tasks to be optimized but their similarity is a priori unknown.

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

AUTHOR CONTRIBUTIONS

John P. Goertz conducted lab experiments. Ruby Sedgwick developed code, conducted code experiments and wrote the manuscript. Ruth Misener and Mark van der Wilk supervised the project, specifically giving guidance on the machine learning aspects. John P. Goertz and Molly M. Stevens also supervised the project, specifically giving guidance on the bioengineering aspects.

ACKNOWLEDGMENTS

This work was supported by the UKRI CDT in AI for Healthcare Grant No. EP/S023283/1, UK Research and Innovation Grant No. EP/P016871/1, the BASF/RAEng Research Chair in Data-Driven Optimization, the US NIH Grant No. 5F32GM131594, the EPSRC IRC Next Steps Plus grant No. EP/R018707/1, and the RAEng Chair in Emerging Technologies award No. CiET20214.

CONFLICT OF INTEREST STATEMENT

JPG and MMS are co-founders at Signatur Biosciences, Inc., a company which seeks to commercialize the medical diagnostic technology this paper focuses on as a use-case, and they are co-inventors in a patent describing a method for amplification-based quantification of nucleic acids. The remaining authors declare no conflict of interest.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are openly available in TL_DOE_4_DNA at https://github.com/RSedgwick/TL_DOE_4_DNA.

ORCID

Ruby Sedgwick http://orcid.org/0000-0001-8267-7995
John P. Goertz https://orcid.org/0000-0003-1196-9282
Molly M. Stevens https://orcid.org/0000-0002-7335-266X
Ruth Misener https://orcid.org/0000-0001-5612-5417
Mark van der Wilk https://orcid.org/0000-0001-7947-6682

REFERENCES

Álvarez, M. A., Rosasco, L., & Lawrence, N. D. (2012). Kernels for vector-valued functions: A review. *Foundations and Trends®in Machine Learning*, *4*(3), 195–266. https://www.nowpublishers.com/article/Details/MAL-036

Badeau, B. A., Comerford, M. P., Arakawa, C. K., Shadish, J. A., & De Forest, C. A. (2018). Engineered modular biomaterial logic gates for environmentally triggered therapeutic delivery. *Nature Chemistry*, *10*(3), 251–258. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5822735/

Bader, J., Narayanan, H., Arosio, P., & Leroux, J.-C. (2023). Improving extracellular vesicles production through a Bayesian optimization-based experimental design. *European Journal of Pharmaceutics and Biopharmaceutics*, *182*, 103–114. https://www.sciencedirect.com/science/article/pii/S0939641122002983

Belanger, D., Vora, S., Mariet, Z., Deshpande, R., Dohan, D., Angermueller, C., Murphy, K., Chapelle, O., & Colwell, L. (2019). Biological sequences design using batched Bayesian optimization.

Blakney, A. K., McKay, P. F., IbarzoYus, B., Hunter, J. E., Dex, E. A., & Shattock, R. J. (2019). The skin you are in: Design-of-experiments optimization of lipid nanoparticle self-amplifying RNA formulations in human skin explants. *ACS Nano*, *13*(5), 5920–5930. https://pubs.acs.org/doi/10.1021/acsnano.9b01774

Bonilla, E. V., Chai, K., & Williams, C. (2007). Multi-task Gaussian process prediction. *Advances in Neural Information Processing Systems*, *20*, 1. https://proceedings.neurips.cc/paper/2007/hash/66368270ffd51418ec58bd793f2d9b1b-Abstract.html

Campbell, K., & Yau, C. (2015). Bayesian Gaussian Process Latent Variable Models for pseudotime inference in single-cell RNA-seq data.

Cao, B., Pan, S. J., Zhang, Y., Yeung, D.-Y., & Yang, Q. (2010). Adaptive transfer learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, *24*(1), 407–412. https://ojs.aaai.org/index.php/AAAI/article/view/7682

Carbonell, P., Jervis, A. J., Robinson, C. J., Yan, C., Dunstan, M., Swainston, N., Vinaixa, M., Hollywood, K. A., Currin, A., Rattray, N. J. W., Taylor, S., Spiess, R., Sung, R., Williams, A. R., Fellows, D., Stanford, N. J., Mulherin, P., LeFeuvre, R., Barran, P., Goodacre, R., Turner, N. J., Goble, C., Chen, G. G., Kell, D. B., Micklefield, J., Breitling, R., Takano, E., Faulon, J.-L., & Scrutton, N. S. (2018). An automated design-build-test-learn pipeline for enhanced microbial production of fine chemicals. *Communications Biology*, *1*(1), 1–10. https://www.nature.com/articles/s42003-018-0076-9

Cox, D. R., & Reid, N. (2000). *The theory of the design of experiments*. CRC Press.

Dai, Z., Álvarez, M., & Lawrence, N. (2017). Efficient modeling of latent information in supervised learning using Gaussian processes. In *Advances in Neural Information Processing Systems* (Vol. 30). Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/1680e9fa7b4dd5d62ece800239bb53bd-Paper.pdf

Damianou, A., & Lawrence, N. D. (2013). Deep gaussian processes. In *Artificial intelligence and statistics*, (pp. 207–215). PMLR.

Degerman, M., Jakobsson, N., & Nilsson, B. (2006). Constrained optimization of a preparative ion-exchange step for antibody purification. *Journal of Chromatography A*, *1113*(1), 92–100. https://www.sciencedirect.com/science/article/pii/S0021967306003013

Deng, F., Pan, J., Liu, Z., Zeng, L., & Chen, J. (2023). Programmable DNA biocomputing circuits for rapid and intelligent screening of SARS-CoV-2 variants. *Biosensors and Bioelectronics*, *223*, 115025. https://www.sciencedirect.com/science/article/pii/S095656632201065X

Droettboom, M., Hunter, J., Firing, E., Caswell, T. A., Elson, P., Dale, D., Lee, J.-J., McDougall, D., Root, B., Straw, A., Seppänen, J. K., Nielsen, J. H., May, R., Varoquaux, A., Yu, T. S., Moad, C., Gohlke, C., Würtz, P., Hisch, T., Silvester, S., Ivanov, P., Whitaker, P., Giuca, M., Thomas, I., & Mmetz, B. N., dhyams and NNemec. (2015). matplotlib: v1.4.3. https://zenodo.org/record/15423

Ebrahimi, S. B., & Samanta, D. (2023). Engineering protein-based therapeutics through structural and chemical design. *Nature Communications*, *14*, 2411. https://www.nature.com/articles/s41467-023-38039-x

Fellermann, H., Shirt-Ediss, B., Kozyra, J., Linsley, M., Lendrem, D., Isaacs, J., & Howard, T. (2019). Design of experiments and the virtual PCR simulator: An online game for pharmaceutical scientists and biotechnologists. *Pharmaceutical Statistics*, *18*(4), 402–406. https://www.ncbi.nlm.nih.gov/pmc/839

Folch, J. P., Lee, R. M., Shafei, B., Walz, D., Tsay, C., van der Wilk, M., & Misener, R. (2023). Combining multi-fidelity modelling and asynchronous batch Bayesian optimization. *Computers & Chemical Engineering*, *172*, 108194.

Gamble, C., Bryant, D., Carrieri, D., Bixby, E., Dang, J., Marshall, J., Doughty, D., Colwell, L., Berndl, M., Roberts, J., & Frumkin, M. (2021). Machine learning optimization of photosynthetic microbe cultivation and recombinant protein production. Preprint, Bioengineering, http://biorxiv.org/lookup/doi/10.1101/2021.08.06.453272

Garnett, R. (2023). *Bayesian optimization*. (pp. 127–129). Cambridge University Press.

Gilman, J., Walls, L., Bandiera, L., & Menolascina, F. (2021). Statistical design of experiments for synthetic biology. *ACS Synthetic Biology*, *10*(1), 1–18. https://doi.org/10.1021/acssynbio.0c00385

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

Goan, E., & Fookes, C. (2020). Bayesian neural networks: An introduction and survey. In K. L. Mengersen, P. Pudlo, & C. P. Robert (Eds.), *Case studies in applied Bayesian data science: CIRM Jean-Morlet Chair, Fall* (pp. 45–87). Springer.

Goertz, J. P., Sedgwick, R., Smith, F., Kaforou, M., Wright, V. J., Herberg, J. A., Kote-Jarai, Z., Eeles, R., Levin, M., Misener, R., Wilk, M. V. D., & Stevens, M. M. (2023). Competitive amplification networks enable molecular pattern recognition with PCR. Biorxiv. Advance online publication.

González, J., Longworth, J., James, D. C., & Lawrence, N. D. (2015). Bayesian optimization for synthetic gene design. arXiv:1505.01627 [stat]. http://arxiv.org/abs/1505.01627

HamediRad, M., Chao, R., Weisberg, S., Lian, J., Sinha, S., & Zhao, H. (2019). Towards a fully automated algorithm driven platform for biosystems design. *Nature Communications*, *10*(1), 5150. https://www.nature.com/articles/s41467-019-13189-z

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., delRío, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., & Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://www.nature.com/articles/s41586-020-2649-2

Hie, B., Bryson, B. D., & Berger, B. (2020). Leveraging uncertainty in machine learning accelerates biological discovery and design. *Cell Systems*, *11*(5), 461–477.e9. https://www.cell.com/cell-systems/abstract/S2405-4712(20)30364-1

Hu, R., Fu, L., Chen, Y., Chen, J., Qiao, Y., & Si, T. (2023). Protein engineering via Bayesian optimization-guided evolutionary algorithm and robotic experiments. *Briefings in Bioinformatics*, *24*(1), bbac570. https://doi.org/10.1093/bib/bbac570

Hua, Y., Ma, J., Li, D., & Wang, R. (2022). DNA-Based biosensors for the biochemical analysis: A review. *Biosensors*, *12*(3), 183. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8945906/

Hutter, C., vonStosch, M., Cruz Bournazou, M. N., & Butté, A. (2021). Knowledge transfer across cell lines using hybrid Gaussian process models with entity embedding vectors. *Biotechnology and Bioengineering*, *118*(11), 4389–4401. https://onlinelibrary.wiley.com/doi/abs/10.1002/bit.27907

Jablonka, K. M., Jothiappan, G. M., Wang, S., Smit, B., & Yoo, B. (2021). Bias free multiobjective active learning for materials design and discovery. *Nature Communications*, *12*(1), 2312. https://www.nature.com/articles/s41467-021-22437-0

Jones, D. R., Schonlau, M., & Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, *13*(4), 455–492. https://doi.org/10.1023/A:1008306431147

Khan, A., Cowen-Rivers, A. I., Grosnit, A., Deik, D.-G.-X., Robert, P. A., Greiff, V., Smorodina, E., Rawat, P., Akbar, R., Dreczkowski, K., Tutunov, R., Bou-Ammar, D., Wang, J., Storkey, A., & Bou-Ammar, H. (2023). Toward real-world automated antibody design with combinatorial Bayesian optimization. *Cell Reports Methods*, *3*(1), 100374.

Kreutz, C., & Timmer, J. (2009). Systems biology: Experimental design. *The FEBS Journal*, *276*(4), 923–942.

Lalchand, V., & Rasmussen, C. E. (2020). Approximate inference for fully bayesian gaussian process regression. In *Symposium on Advances in Approximate Bayesian Inference*, (pp. 1–12). PMLR.

Land, K. J., Boeras, D. I., Chen, X.-S., Ramsay, A. R., & Peeling, R. W. (2019). Reassured diagnostics to inform disease control strategies, strengthen health systems and improve patient outcomes. *Nature Microbiology*, *4*(1), 46–54. https://www.nature.com/articles/s41564-018-0295-3

Lopez, R., Wang, R., & Seelig, G. (2018). A molecular multi-gene classifier for disease diagnostics. *Nature Chemistry*, *10*(7), 746–754. https://www.nature.com/articles/s41557-018-0056-1

Lv, H., Li, Q., Shi, J., Fan, C., & Wang, F. (2021). Biocomputing based on DNA strand displacement reactions. *ChemPhysChem*, *22*(12), 1151–1166. https://onlinelibrary.wiley.com/doi/abs/10.1002/cphc.202100140

Lyu, W., Xue, P., Yang, F., Yan, C., Hong, Z., Zeng, X., & Zhou, D. (2018). An efficient Bayesian optimization approach for automated optimization of analog circuits. *IEEE Transactions on Circuits and Systems I: Regular Papers*, *65*(6), 1954–1967.

MacKay, D. J. (1999). Comparison of approximate methods for handling hyperparameters. *Neural Computation*, *11*(5), 1035–1068.

MacKay, D. J. C. (1998). Introduction to Gaussian processes, *NATO ASI series F: Computer and system sciences* (pp. 133–165). NATO.

Matthews, A. G., Van Der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., León-Villagrá, P., Ghahramani, Z., & Hensman, J. (2017). GPflow: A Gaussian process library using tensorflow. *The Journal of Machine Learning Research*, *18*(1), 1299–1304.

Mehrian, M., Guyot, Y., Papantoniou, I., Olofsson, S., Sonnaert, M., Misener, R., & Geris, L. (2018). Maximizing neotissue growth kinetics in a perfusion bioreactor: An in silico strategy using model reduction and Bayesian optimization. *Biotechnology and Bioengineering*, *115*(3), 617–629. https://onlinelibrary.wiley.com/doi/abs/10.1002/bit.26500

Mowbray, M., Savage, T., Wu, C., Song, Z., Cho, B. A., Del Rio-Chanona, E. A., & Zhang, D. (2021). Machine learning for biochemical engineering: A review. *Biochemical Engineering Journal*, *172*, 108054. https://www.sciencedirect.com/science/article/pii/S1369703X21001303

Narayanan, H., Dingfelder, F., CondadoMorales, I., Patel, B., Heding, K. E., Bjelke, J. R., Egebjerg, T., Butté, A., Sokolov, M., Lorenzen, N., & Arosio, P. (2021). Design of biopharmaceutical formulations accelerated by machine learning. *Molecular Pharmaceutics*, *18*(10), 3843–3853. https://pubs.acs.org/doi/10.1021/acs.molpharmaceut.1c00469

Narayanan, H., Luna, M. F., vonStosch, M., CruzBournazou, M. N., Polotti, G., Morbidelli, M., Butté, A., & Sokolov, M. (2020). Bioprocessing in the digital age: The role of process models. *Biotechnology Journal*, *15*(1), 1900172. https://onlinelibrary.wiley.com/doi/abs/10.1002/biot.201900172

Narayanan, H., vonStosch, M., Feidl, F., Sokolov, M., Morbidelli, M., & Butté, A. (2023). Hybrid modeling for biopharmaceutical processes: Advantages, opportunities, and implementation. *Frontiers in Chemical Engineering*, *5*, 1157889. https://www.frontiersin.org/articles/10.3389/fceng.2023.1157889

Olofsson, S., Mehrian, M., Calandra, R., Geris, L., Deisenroth, M. P., & Misener, R. (2018). Bayesian multiobjective optimisation with mixed analytical and black-box functions: Application to tissue engineering. *IEEE Transactions on Biomedical Engineering*, *66*(3), 727–739.

Papaneophytou, C. (2019). Design of experiments as a tool for optimization in recombinant protein biotechnology: From constructs to crystals. *Molecular Biotechnology*, *61*(12), 873–891. https://doi.org/10.1007/s12033-019-00218-x

Politis, S. N., Colombo, P., Colombo, G., & Rekkas, D. M. (2017). Design of experiments (DoE) in pharmaceutical development. *Drug Development and Industrial Pharmacy*, *43*(6), 889–901. https://doi.org/10.1080/03639045.2017.1291672

Qian, L., Winfree, E., & Bruck, J. (2011). Neural network computation with DNA strand displacement cascades. *Nature*, *475*(7356), 368–372. https://www.nature.com/articles/nature10262

Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press.

Romero, P. A., Krause, A., & Arnold, F. H. (2013). Navigating the protein fitness landscape with Gaussian processes. *Proceedings of the National Academy of Sciences of the United States of America*, *110*(3), E193–E201. https://www.pnas.org/content/110/3/E193

Rosa, S. S., Nunes, D., Antunes, L., Prazeres, D. M. F., Marques, M. P. C., & Azevedo, A. M. (2022). Maximizing mRNA vaccine production with

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

Bayesian optimization. *Biotechnology and Bioengineering*, *119*(11), 3127–3139.

Salvatier, J., Wiecki, T. V., & Fonnesbeck, C. (2016). Probabilistic programming in Python using PyMC3. *Peer Journal of Computer Science*, *2*, e55. https://peerj.com/articles/cs-55

Schonlau, M., Welch, W. J., & Jones, D. R. (1998). Global versus local search in constrained optimization of computer models, *New developments and applications in experimental design* (Vol. 34, pp. 11–26). Institute of Mathematical Statistics. https://projecteuclid.org/ebooks/institute-of-mathematical-statistics-lecture-notes-monograph-series/New-developments-and-applications-in-experimental-design/chapter/Global-versus-local-search-in-constrained-optimization-of-computer-models/10.1214/lnms/1215456182

Schweidtmann, A. M., Clayton, A. D., Holmes, N., Bradford, E., Bourne, R. A., & Lapkin, A. A. (2018). Machine learning meets continuous flow chemistry: Automated optimization towards the Pareto front of multiple objectives. *Chemical Engineering Journal*, *352*, 277–282. https://www.sciencedirect.com/science/article/pii/S1385894718312634

Schweidtmann, A. M., Esche, E., Fischer, A., Kloft, M., Repke, J.-U., Sager, S., & Mitsos, A. (2021). Machine learning in chemical engineering: A perspective. *Chemie Ingenieur Technik*, 93, 12, 2029–2039. https://onlinelibrary.wiley.com/doi/abs/10.1002/cite.202100083

Sedgwick, R., Goertz, J., Stevens, M., Misener, R., & van der Wilk, M. (2020). Design of experiments for verifying biomolecular networks. *arXiv*. Advance online publication. https://arxiv.org/abs/2011.10575

Selega, A., & Campbell, K. R. (2022). Multi-objective Bayesian optimization with heuristic objectives for biomedical and molecular data analysis workflows. *Transactions on Machine Learning Research*.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & de Freitas, N. (2016). Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, *104*(1), 148–175.

Sharpe, C., Seepersad, C. C., Watts, S., & Tortorelli, D. (2018). Design of mechanical metamaterials via constrained Bayesian optimization. In *44th Design Automation Conference* (Vol. 2A, pp. V02AT03A029), American Society of Mechanical Engineers. https://asmedigitalcollection.asme.org/IDETC-CIE/proceedings/IDETC-CIE2018/51753/Quebec%20City,%20Quebec,%20Canada/273625

Siuti, P., Yazbek, J., & Lu, T. K. (2013). Synthetic circuits integrating logic and memory in living cells. *Nature Biotechnology*, *31*(5), 448–452. https://www.nature.com/articles/nbt.2510

Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems* (Vol. 25). Curran Associates, Inc. https://proceedings.neurips.cc/paper/2012/hash/05311655a15b75fab86956663e1819cd-Abstract.html

Sun, Y., Nathan-Roberts, W., Pham, T. D., Otte, E., & Aickelin, U. (2022). Multi-fidelity Gaussian process for biomanufacturing process modeling with small data. Advance online publication. http://arxiv.org/abs/2211.14493

Swersky, K., Snoek, J., & Adams, R. P. (2013). Multi-Task Bayesian Optimization. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, & K. Q. Weinberger, (Eds.), *Advances in Neural Information Processing Systems* (Vol. 26, pp. 2004–2012). http://papers.nips.cc/paper/5086-multi-task-bayesian-optimization.pdf

Taylor, C. J., Felton, K. C., Wigh, D., Jeraal, M. I., Grainger, R., Chessari, G., Johnson, C. N., & Lapkin, A. A. (2023). Accelerated chemical reaction optimization using multi-task learning. *ACS Central Science*, *9*(5), 957–968. https://pubs.acs.org/doi/10.1021/acscentsci.3c00050

The Pandas Development team, The pandas development. (2023). pandas-dev/pandas: Pandas. https://zenodo.org/record/7979740

Tighineanu, P., Skubch, K., Baireuther, P., Reiss, A., Berkenkamp, F., & Vinogradska, J. (2022). Transfer learning with Gaussian processes for Bayesian optimization. In Proceedings of The 25th International Conference on *Artificial Intelligence and Statistics*, (pp. 6152–6181). PMLR. https://proceedings.mlr.press/v151/tighineanu22a.html

Titsias, M. (2009). Variational learning of inducing variables in sparse Gaussian processes. In *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, (pp. 567–574). PMLR. https://proceedings.mlr.press/v5/titsias09a.html

Titsias, M., & Lawrence, N. D. (2010). Bayesian Gaussian process latent variable model. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, (pp. 844–851). JMLR Workshop and Conference Proceedings. https://proceedings.mlr.press/v9/titsias10a.html

Uhrenholt, A. K., & Jensen, B. S. (2019). Efficient Bayesian optimization for target vector estimation. In K. Chaudhuri, and M. Sugiyama, (Eds.), *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics Proceedings of Machine Learning Research*, (Vol. 89, pp. 2661–2670). PMLR. https://proceedings.mlr.press/v89/uhrenholt19a.html

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … van Mulbregt, P. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, *17*(3), 261–272. https://www.nature.com/articles/s41592-019-0686-2

Wadle, S., Lehnert, M., Rubenwolf, S., Zengerle, R., & vonStetten, F. (2016). Real-time PCR probe optimization using design of experiments approach. *Biomolecular Detection and Quantification*, *7*, 1–8. https://www.sciencedirect.com/science/article/pii/S2214753515300139

Wang, X., Jin, Y., Schmitt, S., & Olhofer, M. (2023). Recent advances in Bayesian optimization. *ACM Computing Surveys*, *55*(13), 287:1–287:36. https://dl.acm.org/doi/10.1145/3582078

Warnes, J. J., & Ripley, B. D. (1987). Problems with likelihood estimation of covariance functions of spatial Gaussian processes. *Biometrika*, *74*(3), 640–642.

Zadeh, J. N., Steenberg, C. D., Bois, J. S., Wolfe, B. R., Pierce, M. B., Khan, A. R., Dirks, R. M., & Pierce, N. A. (2011). NUPACK: Analysis and design of nucleic acid systems. *Journal of Computational Chemistry*, *32*(1), 170–173. https://onlinelibrary.wiley.com/doi/abs/10.1002/jcc.21596

Zhang, Y., Tao, S., Chen, W., & Apley, D. W. (2020). A latent variable approach to Gaussian process modeling with qualitative and quantitative factors. *Technometrics*, *62*(3), 291–302. https://www.tandfonline.com/doi/full/10.1080/00401706.2019.1638834

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., & He, Q. (2021). A comprehensive survey on transfer learning. *Proceedings of the IEEE*, *109*(1), 43–76.

How to cite this article: Sedgwick, R., Goertz, J. P., Stevens, M. M., Misener, R., & ven der Wilk, M. (2025). Transfer learning Bayesian optimization for competitor DNA molecule design for use in diagnostic assays. *Biotechnology and Bioengineering*, *122*, 189–210. https://doi.org/10.1002/bit.28854

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

APPENDIX A

Accronyms

AvgGP	average Gaussian process
BP	number of base pairs
DNA	deoxyribonucleic acid
ELBO	evidence lower bound to marginal likelihood for LVMOGP
GC	percentage guanine–cytosine content
LMC	linear Model of coregionalization
LVMOGP	latent variable multioutput Gaussian process
MOGP	multioutput Gaussian process
NLPD	negative log predictive density
PCR	polymerase chain reaction
RMSE	root mean squared error

Functions

$\alpha_c(\cdot)$	acquisition function including probability of feasibility
$\alpha_{EI}(\cdot)$	expected improvement acquisition function
$\mathbf{g}(\cdot)$	latent Gaussian processes in the linear model of coregionalisation
$\mathcal{GP}$	Gaussian process
$f(\cdot)$	function of $\mathbf{x}$
$f_{\text{drift}}(\cdot)$	drift function
$f_{\text{rate}}(\cdot)$	rate function in competitor amplification
$k(\cdot,\cdot)$	Gaussian process covariance function
$K_{ii}(\cdot,\cdot)$	covariance function of the data
$K_{iu}(\cdot,\cdot)$	cross covariance function between the data and inducing points
$K_{uu}(\cdot,\cdot)$	covariance function of the inducing points
$m(\cdot)$	Gaussian process mean function
$PF(\cdot)$	probability of feasibility

Parameters and variables

ℓ_d	lengthscale of dimension d
ϵ	noise added to y where $\epsilon \sim \mathcal{N}(0, \sigma_n^2\mathbf{I})$
λ	noncentrality parameter of target vector optimization expected improvement
$\boldsymbol{\delta}$	stochastic variable defined as the squared difference between observed outputs and the target value
$\boldsymbol{\mu}_{h_p}$	mean of the p^{th} latent variable
$\mathbf{f}_*$	predictions at locations X_*
$\mathbf{h_p}$	latent variable of the p^{th} output function
$\mathbf{I}$	identity matrix
$\mathbf{u}$	inducing variables
$\mathbf{W}$	vector of weights of the latent functions in the linear model of coregionalisation
$\mathbf{x}$	input location such that $\mathbf{x} \in \mathbb{R}^D$
$\mathbf{y}_{\text{drift}}$	drift output data
$\mathbf{y}_{\text{rate}}$	rate output data
$\mu(X_*)$	predicted mean at locations X_*
v	carrying capacity
$\sigma(X_*)$	predicted covariance at locations X_*
σ_k^2	lernel variance
σ_n^2	noise variance of Gaussian process
Σ_{h_p}	variance of the p^{th} latent variable
τ	cycle number
θ	Gaussian process hyperparameters
B	coregionalization matrix in the LMC
D	dimensions of $\mathbf{x}$
F	fluorescence in DNA amplification reaction
F_0	fluorescence at the beginning of the DNA amplification reaction
F_T	fluorescence at the end of the DNA amplification reaction
H	latent variables such that $H = [\mathbf{h}_1, ..., \mathbf{h}_P]^T \in \mathbb{R}^{Q_H \times P}$
M	mean of the variational distribution on Z
P	number of output functions in multi-output Gaussian Process
$q(\cdot)$	variational distribution
Q	number of covariance matrices in the LMC
S	variance of the variational distribution on Z
t	$= \delta\gamma^{-2}$
T_{rate}	target rate
T_{drift}	drift threshold
X	training inputs of Gaussian Process $X = \{\mathbf{x}_1, ..., \mathbf{x}_N\} \in \mathbb{R}^{N \times D}$
X_*	locations to be evaluated
y	noisy evaluations of $\mathbf{x}$
y_{best}	data point which is closest to the target out of the train and test data sets for a given surface
Z	inducing points

Miscellaneous

$\mathcal{H}$	the latent space in the LVMOGP
G	an approximation to the cumulative noncentral χ^2 distribution function

APPENDIX B: LATENT VARIABLE MULTIOUTPUT GAUSSIAN PROCESS (LVMOGP) IMPLEMENTATION

Gaussian processes are normally trained by maximizing the log marginal likelihood. However, the presence of the latent variable distributions in the LVMOGP means the log marginal likelihood is no longer tractable. Instead, Dai et al. (2017) used variational inference to approximate a lower bound to this log marginal likelihood, following the method proposed by Titsias (2009) and Titsias and Lawrence (2010). In variational inference, the aim is to minimize the Kullback–Leibler divergence between an approximate posterior and a true posterior.

Our implementation of the LVMOGP takes a concatenation of the input data and their corresponding latent variables $\tilde{X} = [X, H_{:}] \in \mathbb{R}^{N \times (D+Q_H)}$ where $H_{:}$ to denotes the vector of latent inputs for each observed data point. All inputs X_p for the same output dimension will have the same latent variable, $\boldsymbol{h}_p$.

For the LVMOGP this variational lower bound is given as:

$$\begin{aligned} ELBO = &-\frac{NP}{2} log\,(2\pi\sigma_n^2) + \sum_{i=1}^{N}\left[-\frac{1}{2\sigma_n^2}y_{:i}^T y_{:i} + \frac{1}{\sigma_n^2}y_{:i}\langle K_{iu}\rangle_{q(H_{:i})}K_{uu}^{-1}M\right. \\ &-\frac{1}{2\sigma_n^2}Tr\left(K_{uu}^{-1}\langle K_{iu}^T K_{iu}\rangle_{q(H_{:i})}K_{uu}^{-1}(MM^T + S)\right) \\ &\left.-\frac{1}{2\sigma_n^2}\left(Tr\left(\langle K_{ii}\rangle_{q(H_{:i})}\right) - Tr\left(K_{uu}^{-1}\langle K_{iu}^T K_{iu}\rangle_{q(H_{:i})}\right)\right)\right] \\ &-KL\left[q(\mathbf{u})||p(\mathbf{u})\right] - \sum_{i=1}^{N} KL\left[q(H_{:i})||p(H_{:i})\right], \end{aligned} \tag{B1}$$

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

where $\langle K \rangle_{q(h_i)}$ denotes a kernel expectation over the variational distribution of the latent variable of data point i. K_{ii} and K_{uu} are the covariance functions of the data and the inducing points Z respectively, while K_{iu} is the cross covariance function between the two. Tr is the trace of a matrix. M and S are the mean and covariance of the variational distribution over inducing points $q(Z) \sim \mathcal{N}(M, S)$. The second term in this expression can be viewed as a data fit term, while the last term can be seen as a complexity penalty.

Two types of prediction are relevant using the LVMOGP. The first is when we have new input points X_* and new position on the latent space h_*. In this case, the posterior prediction can be calculated in closed form. The second, and more likely, prediction case is when we want to predict a new point X_* at a point on the latent space where we already have data with latent variable h_p. This integration is intractable, but following Titsias and Lawrence (2010), the first and second moments can be computed in closed form if using a squared exponential kernel.

APPENDIX C: TOY DATA SET CREATION

The data set used in Figure 2 was generated by creating two latent functions from samples of a Gaussian process prior with the squared exponential kernel, in Equation (3), and $\ell = 0.3$ and $\sigma_k^2 = 2$ and multiplying them by random weights to create the output functions. To ensure the output functions could generate data anywhere, a Gaussian process was fitted to the densely sampled points for each output function. Data was then generated by evaluating the mean of the Gaussian processes at varying input locations adding noise $\epsilon = \mathcal{N}(0, 0.1)$. Similarly to the competitor data set, the amount and location of data observed on each output function varies.

APPENDIX D: GAUSSIAN PROCESS IMPLEMENTATION

D.1 DATA STANDARDIZATION

For both the synthetic and competitor data sets we standardize the input and output data by subtracting the mean and dividing by the standard deviation, such that:

$$\bar{x} = \frac{x - \mu_x}{\sigma_x}, \tag{D1}$$

and similar for the output data. This is common practice for Gaussian process regression as it reduces numerical instability and allows for better interpretability of hyperparameter values, which is useful for initialization.

D.2 Choice of Gaussian process prior

When using Gaussian process models for real-world optimization tasks the Gaussian process prior should be informed by existing knowledge of the system. For example, the choice of kernel function can express belief of the smoothness or periodicity of the function and a mean function may be selected if there is a known trend in the data (Rasmussen & Williams, 2006, Chapters 2 & 4). For the competitor design task, we believed the function to be smooth, so opted for the squared exponential kernel in Equation (3). We did not have any prior information about a trend in the data so used a zero mean function.

D.3 Gaussian process hyperparameter training

Gaussian processes are generally trained using the marginal likelihood, which automatically trades off data fit and model complexity, guarding against overfitting (Rasmussen & Williams, 2006, Chapter 5). Ideally, we would perform full Bayesian inference over the Gaussian process hyperparameters, however, this is often difficult and expensive due to the need to use approximation techniques to evaluate intractable integrals (Lalchand & Rasmussen, 2020).

Instead, we use type II maximum likelihood, a common approach of maximizing the marginal likelihood with respect to the hyperparameters. With sufficient data, this approach is justified based on the Laplace approximation and because in practice the posterior for the hyperparameters tends to be highly peaked (MacKay, 1999). However, at lower data regimes, the non-convexity of the marginal likelihood surface can cause overfitting due to multiple modes (Lalchand & Rasmussen, 2020). Low data regimes can also lead to hyperparameters being weakly identified, leading to flat ridges in the marginal likelihood surface, making the optimization sensitive to starting values (Warnes & Ripley, 1987). So, at low data regimes, Gaussian processes trained with type II maximum likelihood can over fit. As more data is collected, the type II maximum likelihood approach is a reasonable approximation and the marginal likelihood will automatically trade off model complexity and data fitting, preventing overfitting (Rasmussen & Williams, 2006, Chapter 5).

The marginal likelihood optimization surface is non-convex and therefore gradient based optimizers will only find local optima meaning the result dependent on initialization (MacKay, 1998). To overcome this, and reduce overfitting, we use random restarts, along side principled methods of initialization, to fit the same Gaussian process model multiple times, and then select the hyperparameter configuration with the best log marginal likelihood. These regimes, introduced in Appendix D.4, differ slightly for the different models. We use gradient descent to optimize the marginal likelihood each time.

D.4 Hyperparameter initialization

For all model, unless otherwise states, we initialize the lengthscale randomly as $\ell \sim \text{Uniform}(0, 1)$, noise variance randomly as $\sigma_n \sim \text{Uniform}(0, 0.1)$ which is equivalent to the noise being between 0% and 10% of the data variance and kernel variance $\sigma_k = 1$. These settings are standard proactive for Gaussian process regression (Matthews et al., 2017). For the MOGP and AvgGP we did nine random restarts with these settings.

For the LMC we used three different methods for initializing $\mathbf{W}$ and $\boldsymbol{\kappa}$, with three random restarts for each:

- *Both* $\mathbf{W}$ *and* $\boldsymbol{\kappa}$ *random*. In this initialization, we initialize $\mathbf{W} \sim \text{Uniform}(0.1, 1)$ and $\boldsymbol{\kappa} \sim \text{Uniform}(0.1, 1)$.
- $\mathbf{W}$ *random and* $\boldsymbol{\kappa} = 0$. In this initialization $\mathbf{W} \sim \text{Uniform}(0.1, 1)$ and $\boldsymbol{\kappa} = 10^{-6}$. This initialization was chosen as we thought it would favor solutions with small $\boldsymbol{\kappa}$ so it would better fit the linear correlation case, where the test functions are generated as linear combinations of some linear functions.

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

- **W** *random and* $\kappa = 1$. In this initialization $\mathbf{W} \sim \text{Uniform}(0.1, 1)$ and $\kappa = 1$. We chose this initialization to favor large κ, which is useful for the uncorrelated test case, as it would encourage the output functions to behave independently of each other.

The random initializations for **W** helped the initializations for two reasons: firstly, in the GPflow implementation if **W** is not initialized it defaults to a rank of 1, and secondly by initializing to random values rather than all one value we avoid saddle points on the optimization surface.

For the LVMOGP we used three different initialization procedures, again with three random restarts for each:

- *Random*. In this initialization all hyperparameters and variational parameters were initialized randomly. the means of the latent variables were initialized as $\mu_H \sim \text{Uniform}(-1, 1)$.
- *GPy*. This is the method used in the GPy implementation of the LVMOGP (Dai et al., 2017), that has following three steps:
  1. A sparse MOGP is fitted to the data using a set of inducing points Z which are common to all outputs. The mean predictions $\mu(Z) \in \mathbb{R}^{N_U \times P}$ of the output function values at these inducing inputs is then calculated:

$$\mu(Z) = K(Z, Z)\left[K(Z, Z) + \sigma_n^2 \mathbf{I}\right]^{-1} Y. \tag{D2}$$

The sparse MOGP is used is ensure all output functions are observed at the same input locations for the functional PCA, which is necessary when data is observed at different locations on different surfaces. It also serves the purpose of smoothing the data plus the trained lengthscales are used to initialize the lengthscales of the observed dimensions of the LVMOGP.

  2. The mean predictions $\mu(Z) \in \mathbb{R}^{N_U \times P}$ are then used as inputs to functional PCA. The first Q_H eigenvectors $V \in \mathbb{R}^{N_U \times Q_H}$ and eigenvalues $\{\lambda_q\}_{q=1}^{Q_H}$ of $\mu(Z)^T\mu(Z)$ are calculated and used to project $\mu(Z)$ into latent space

$$H = \mu(Z)^T V, \tag{D3}$$

where $H \in \mathbb{R}^{P \times Q_H}$. The relative contributions of each of the eigenvalues is also calculated as:

$$\varsigma_q = \frac{\tilde{\lambda}_q}{\max\{\tilde{\lambda}_i\}_{i=1}^{Q_H}} \quad \tilde{\lambda}_q = \frac{\lambda_q}{\sum_{i=1}^{Q_H} \lambda_i} \tag{D4}$$

  3. The latent variables H from the functional PCA are used to initialize the latent variables of a Bayesian Gaussian process latent variable model. The lengthscales of the Bayesian Gaussian process latent variable model are initialized to $\left\{\frac{1}{\varsigma_q}\right\}_{q=1}^{Q_H}$. Once the Bayesian Gaussian process latent variable model is trained, the latent variables and hyperparameters of the Bayesian Gaussian process latent variable model are used to initialize those of the LVMOGP.

- *PCA*. In this initialization, the first two steps of the *GPy* initialization are followed. This means fitting a sparse MOGP to the data and performing principle component analysis (PCA) on the posterior predictions at inducing point locations. The MOGP hyperparameters were then used to initialize the LVMOGP observed lengthscale, kernel variance and noise variance. The output of the PCA was used to initialize the latent variable means and the lengthscale of the latent dimensions. This initialization was chosen as a simplified version of the *GPy* initialization.

See the github repositories in Appendix E for more details.

In the synthetic experiments, we found the method of initializing the hyperparameters affected the end log marginal likelihood, with no initialization outperforming all others for each model. Therefore, we decided to continue with all initializations for the PCR data experiments. For the PCR data experiments we did 10 random restarts for each initialization, due to the randomness of some of the initializations.

D.5 Latent dimensions

The LMC and LVMOGP have hyperparameters that need to be set for the number of latent functions and dimensions respectively.

For the LMC, the rank of the similarity matrix **B** needs to be selected. This is equivalent to the number of latent functions (Álvarez et al., 2012). If the rank is too low, the LMC will fail to explain the data well. However, if it is too high, the LMC can suffer from overfitting when data is limited. Unlike other hyperparameters, such as the lengthscale or noise variance, there is no continuous way to select this hyperparameter. Therefore, to select this parameter for a given problem, it is necessary to fit multiple LMC models with different ranks and select the best one by comparing the marginal likelihoods or cross validation. In reality, this isn't always feasible as it is computationally expensive, and data may be limited.

Figure D1 shows results of cross validation experiments introduced in Section 3.2 for the LMC with this latent dimension hyperparameters set to 2 and 10. The test setting was the same as in Section 3.2 and the cross validation was repeated 70 times. The LMC has worse NLPD with more latent functions, most likely because more hyperparameters have been introduced, increasing the chances of overfitting when the data set isn't large.

Similarly, for the LVMOGP, the dimensionality of the latent space needs to be selected. However, if a kernel that treats each dimension as independent is used, the LVMOGP can "switch off" unessential dimensions (Titsias & Lawrence, 2010). This is done by making the lengthscales of the unessential dimensions really large, so there is no variation across those dimensions. This means the LVMOGP can automatically reduce the number of dimensions to those that give a good trade off between data fit and model complexity. This effect occurs in the latent dimensions of the drift parameter in Figure 8, where all the points are lined up on a single dimension.

Figure D2 shows results of cross validation experiments introduced in Section 3.2 for the LVMOGP with 2 and 10 latent dimensions. Unlike the LMC, the LVMOGP however performs the same as it can switch off unnecessary dimensions for the 10 latent case.

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

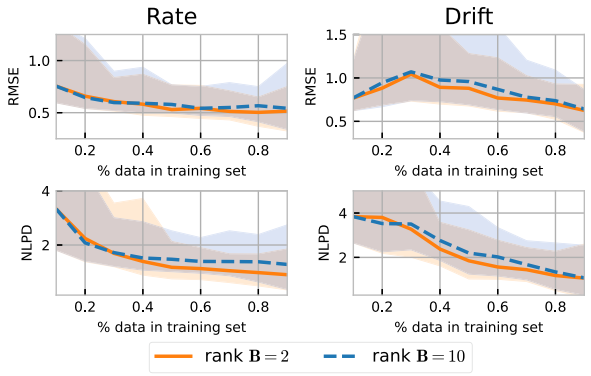


FIGURE D1 Cross validation results for the linear model of coregionalization with similarity matrices B with rank 2 and 10. For each percentage train, 70 different train test splits were used. The LMC with rank 10 matrix has worse NLPD than that with rank 2, suggesting overfitting is occurring.

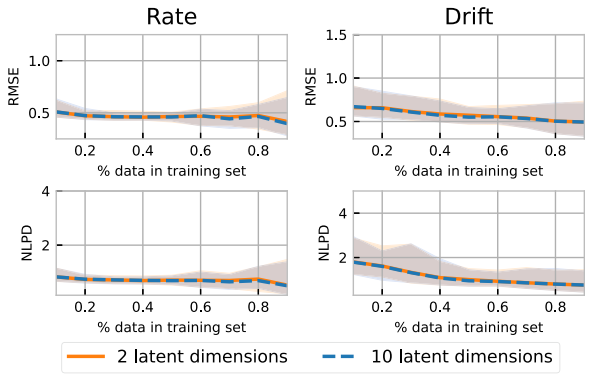


FIGURE D2 Cross validation results for the latent variable multioutput Gaussian process with 2 and 10 dimensional latent spaces. For each percentage train, 70 different train test splits were used. Due to the automatic relevance determining properties of the latent space kernel, there is very little difference in performance

Generally when conducting Bayesian optimization, data is very scarce to begin with, so it is not possible to perform model selection for the rank of the LMC **B** matrix. One option would be to fit many models with different ranks and select the one with the best marginal likelihood, but this is generally too computationally expensive. This is one of the limitations of the LMC. Therefore, for the Bayesian optimization experiments we used a rank of 2 for the LMC and 2 dimensions for the LVMOGP as we expected these settings to give enough flexibility to transfer information while limiting chances of overfitting.

APPENDIX E: DATA AND CODE AVAILABILITY

Raw data is available on request from rdm-enquiries@imperial. ac. uk.

Implementation of the methods outlined in this article requires a basic knowledge of python. We provide two github repositories, which include the methods used and jupyter notebooks demonstrations. The first repository, https://github.com/RSedgwick/TLGPs, contains code for each of the Gaussian process models, the synthetic experiments and some jupyter notebooks demonstrating the use of the models. It also contains instructions for running the code. This repository is application agnostic and could easily be transferred to other use cases.

The second repository, https://github.com/RSedgwick/TL_DOE_4_DNA, is specifically tailored to the competitor use case. This repository contains the code for the cross validation and Bayesian optimization experiments, as well as code for data processing and notebooks fodata_summaryr results analysis. This repository is more targeted to our specific use case.

APPENDIX F: DATA SUMMARY

Each competitor is defined by its primer-reporter combination. For each of these primer-pair combinations we then have data at different guanine-cytosine content and no. of base pairs combinations. Table F1 gives a summary of the number of unique locations on each of the competitors.

APPENDIX G: EXTRA BAYESIAN OPTIMIZATION RESULTS

The following tables contain extra results for the Bayesian optimization experiments. The first table in each section, Tables G1 and G4, shows counts of the first model to get to the best point on a surface for all competitors and seeds. If two models get to the best point on the same iteration, they are both counted as "winners". The second table, Tables G2 and G5 shows counts of the models with the lowest cumulative regret for each competitor and seed. The same thing applies if two models have the same cumulative regret. For the single objective optimization, Table G3 shows the average number of iterations for each model to get within tolerance of the target rate (+/− 0.05). For the penalized optimization Table G6 shows the average number of iterations for each model to get either within tolerance of the rate target with no drift penalty, or to the best point (which may have a drift penalty). For some of the runs with the drift penalty, some of the models failed to get to the best point for some surfaces within the experimental budget. In these cases, those surfaces were discarded and the average was taken for the surfaces where all the models had managed to get to the best point within the experimental budget.

G.1 Single objective optimization

Extra results for the single objective Bayesian optimization. These results demonstrate that the LVMOGP gets to the best point more often (Table G1) and has has the lowest cumulative regret (Table G2) more often than the other models. The LVMOGP also reaches the best point in the lowest number of iterations for all the learning scenarios (Table G3).

G.2 Bayesian optimization with drift penalty

Extra results for the Bayesian optimization with a penalty on drift. These results demonstrate that the LVMOGP gets to the best point more often (G4) and has has the lowest cumulative regret (Table G5) more often than the other models for most of the learning scenarios.

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

TABLE F1 Summary of the amount of data we have for each competitor design surface.

Not to be optimized Primer reporter combination	No. unique locations	To be optimized Primer reporter combination	No. unique locations
FP004-RP004-EvaGreen	28	FP004-RP004-Probe	53
FP002-RP002x-Probe	12	FP001-RP001x-EvaGreen	24
FP004-RP004x-Probe	12	FP001-RP001x-Probe	20
FP001-RP001-Probe	9	RP001x-FP002-Probe	19
FP001-RP005-Probe	8	FP002-RP002x-EvaGreen	15
FP004-RP004x-EvaGreen	8	FP005-FP001-EvaGreen	14
FP003-RP008-Probe	5	FP004-FP005-Probe	8
FP006-RP006-Probe	5	FP005-FP001-Probe	8
FP005-RP005-Probe	5	FP005-FP004-EvaGreen	8
FP002-RP002-EvaGreen	4	RP002x-FP005-Probe	8
FP002-RP006-Probe	4	RP008x-FP001-EvaGreen	8
FP057.1.0-RP003x-Probe	3	RP008x-FP005-Probe	8
FP003-RP008x-EvaGreen	3	FP001-RP004-EvaGreen	7
FP003-RP008-EvaGreen	3	RP002x-FP004-EvaGreen	6
FP002-RP002-Probe	3	FP002-RP004-EvaGreen	3
FP001-RP001-EvaGreen	2	RP002x-FP002-EvaGreen	2
FP003-RP003-Probe	1		
FP057.1.0-RP003x-EvaGreen	1		

Note: Each unique location refers to a unique GC–BP combination.

Abbreviations: BP, number of base pairs; GC, percentage guanine–cytosine content.

TABLE G1 Table showing counts of the first Gaussian process model to reach the best point on a surface for the single objective Bayesian optimization experiments.

Learning scenario	Starting point	MOGP	AvgGP	LMC	LVMOGP
Learning many	Center	124	121	144	**255**
	Model's choice	107	119	97	**147**
One at a time	Center	140	140	156	**215**
	Model's choice	86	118	87	**191**

Note: The counts are the number of times a Gaussian process model did the best on a competitor for each seed. If two Gaussian process models performed the same for a given instance, they are both counted. This is for 16 competitors and 25 random seeds. The bold values are indicate the best performance across the different models.

Abbreviations: AvGP, average Gaussian process; LMC, linear model of coregionalization; LVMOGP, latent variable multiouput Gaussian process; MOGP, multiouput Gaussian process.

TABLE G2 Table showing counts of the Gaussian process model with the lowest cumulative regret on a surface for the single objective Bayesian optimization experiments.

Learning scenario	Starting point	MOGP	AvgGP	LMC	LVMOGP
Learning many	Center	182	80	140	**197**
	Model's choice	85	94	83	**117**
One at a time	Center	129	140	131	**206**
	Model's choice	99	106	87	**159**

Note: The counts are the number of times a Gaussian process model did the best on a competitor for each seed. If two Gaussian process models performed the same for a given instance, they are both counted. This is for 16 competitors and 25 random seeds. The bold values are indicate the best performance across the different models.

Abbreviations: AvGP, average Gaussian process; LMC, linear model of coregionalization; LVMOGP, latent variable multiouput Gaussian process; MOGP, multiouput Gaussian process.

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

TABLE G3 Table showing the mean number of iterations need for the models to get within tolerance of the target rate (+/− 0.05) for the single objective optimization.

Learning scenario	Starting point	MOGP	AvgGP	LMC	LVMOGP
Learning many	Center	3.13	3.25	3.11	**2.58**
	Model's choice	3.08	2.63	3.09	**2.44**
One at a time	Center	2.94	3.06	2.85	**2.15**
	Model's choice	2.94	2.63	2.63	**1.81**

Note: This is for 16 competitors and 25 random seeds. The bold values are indicate the best performance across the different models.

Abbreviations: AvGP, average Gaussian process; LMC, linear model of coregionalization; LVMOGP, latent variable multiouput Gaussian process; MOGP, multiouput Gaussian process.

TABLE G4 Table showing counts of the first Gaussian process model to reach the best point on a surface for the penalized Bayesian optimization experiments.

Learning scenario	Starting point	MOGP	AvgGP	LMC	LVMOGP
Learning many	Center	142	157	123	**165**
	Model's choice	89	**122**	101	111
One at a time	Center	141	137	153	**217**
	Model's choice	75	102	79	**164**

Note: The counts are the number of times a Gaussian process model did the best on a competitor for each seed. If two Gaussian process models performed the same for a given instance, they are both counted. This is for 16 competitors and 24 random seeds. The bold values are indicate the best performance across the different models.

Abbreviations: AvGP, average Gaussian process; LMC, linear model of coregionalization; LVMOGP, latent variable multiouput Gaussian process; MOGP, multiouput Gaussian process.

The LVMOGP also reaches the best point in the lowest number of iterations for all the learning scenarios (Table G6).

G.3 Comparison of choice of first point

Table G7 shows the average regret of the first data point chosen by each of the models for each of the learning scenarios, for the single objective case. From this table, it is clear to see the AvgGP and the LVMOGP improve on the regret of the central point, and outperform the random selection of the MOGP and LMC. This demonstrates that having a principled method of selecting the first point is useful for reducing regret.

G.4 Comparison of number of experiments

Figure G1 shows box plots of the number of iterations taken by each model to reach the best point for each learning scenario for the experiments in Section 3.3. The distributions here are across the 16 different competitors and 24 random restarts. For the constrained optimization case, for some seeds some of the models didn't reach the best point for competitor FP004-RP004-Probe within the experimental budget. In these cases we set the number of experiments to the total experimental budget (10 or 20 depending on the scenario).

These results show the LVMOGP on average requires less experiments than the other models to select the best point, demonstrating how this approach can reduce the number of experiments

TABLE G5 Table showing counts of the Gaussian process model that had the lowest cumulative regret on a surface for the penalized Bayesian optimization experiments.

Learning scenario	Starting point	MOGP	AvgGP	LMC	LVMOGP
Learning many	Center	**180**	118	100	163
	Model's choice	85	103	84	**111**
One at a time	Center	173	118	139	**204**
	Model's choice	83	70	65	**156**

Note: The counts are the number of times a Gaussian process model did the best on a competitor for each seed. If two Gaussian process models performed the same for a given instance, they are both counted. This is for 16 competitors and 24 random seeds. The bold values are indicate the best performance across the different models.

Abbreviations: AvGP, average Gaussian process; LMC, linear model of coregionalization; LVMOGP, latent variable multiouput Gaussian process; MOGP, multiouput Gaussian process.

TABLE G6 Table showing the mean number of iterations need for the models to either get within tolerance of the target rate (+/− 0.05) without drift penalty or reach the best point (which may have a penalty) for the penalized optimization.

Learning scenario	Starting point	MOGP	AvgGP	LMC	LVMOGP
Learning many	Center	3.56	4.03	3.96	**3.47**
	Model's choice	3.78	3.00	3.66	**2.63**
One at a time	Center	3.70	**3.44**	3.51	3.56
	Model's choice	3.63	3.06	3.46	**2.52**

Note: For some runs, one or more of the models would not achieve this within the experimental budget. In these cases, the number of iterations to best point was set to the experimental budget. This is for 16 competitors and 24 random seeds. The bold values are indicate the best performance across the different models.

Abbreviations: AvGP, average Gaussian process; LMC, linear model of coregionalization; LVMOGP, latent variable multiouput Gaussian process; MOGP, multiouput Gaussian process.

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

TABLE G7 Table of the mean regret of the first data point for each of the learning scenarios for each of the models.

Learning scenario	Starting point	MOGP	AvgGP	LMC	LVMOGP
Learning many	Center	0.588	0.588	0.588	0.588
	Model's choice	0.651	0.499	0.703	0.464
One at a time	Center	0.588	0.588	0.588	0.588
	Model's choice	0.675	0.308	0.623	0.309

Note: The bold values are indicate the best performance across the different models.

Abbreviations: AvGP, average Gaussian process; LMC, linear model of coregionalization; LVMOGP, latent variable multiouput Gaussian process; MOGP, multiouput Gaussian process.

needed to optimize the competitors. All the models have some outliers, this could be due to being unlucky with the initial data it receives or suboptimal hyperparameter optimization.

G.5 Initial surfaces for Bayesian optimization

In the Bayesian optimization experiments outlined in section 3.3, we start with the two competitor surfaces with the most data fully observed, FP004-RP004-EvaGreen and FP002-RP002x-Probe. This is so the Gaussian process models have the chance to learn a reasonable prediction before the iterative Bayesian optimization process. It also replicates the case where we already have limited data on a couple of competitors and want to optimize more competitors.

To assess whether the choice of initial surfaces affects the results in our experiments, we investigate two other combinations

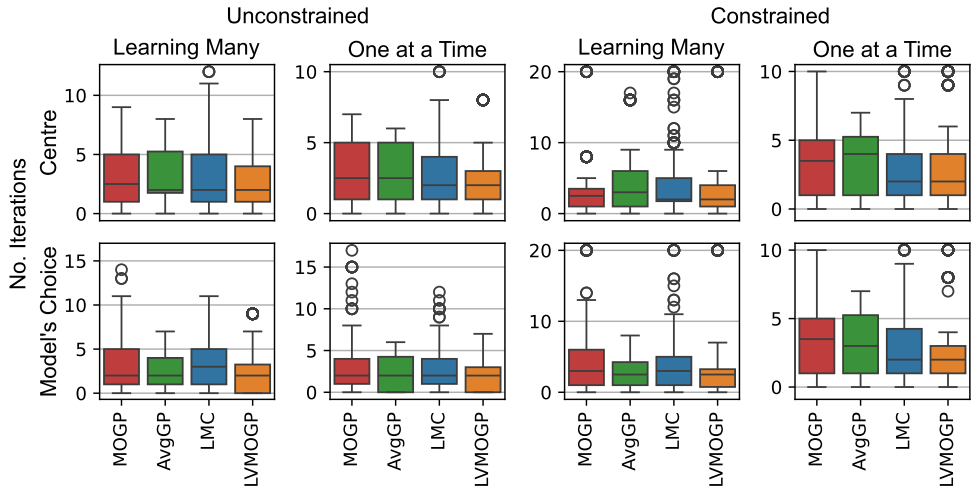


FIGURE G1 Box plots of the number of iterations needed to reach within 0.05 tolerance of the best point for all learning scenarios for the Bayesian optimization experiments outlined in Section 3.3

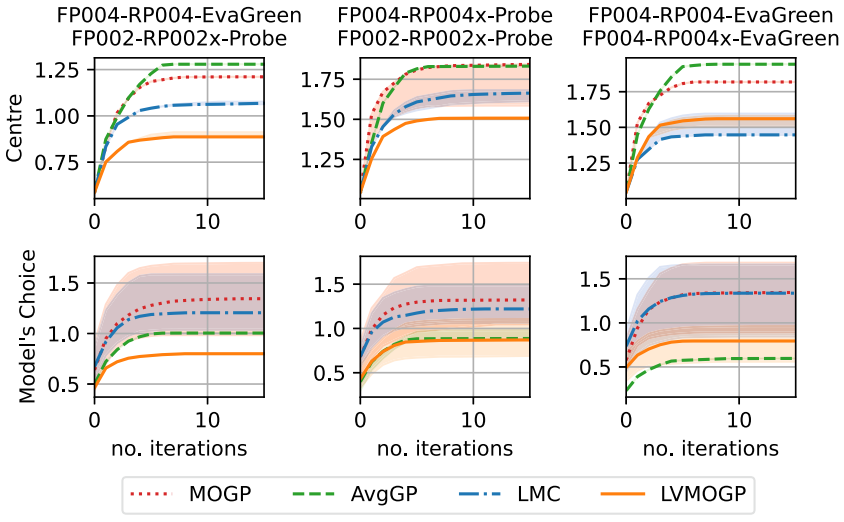


FIGURE G2 Results of single objective Bayesian optimization experiments for the *learning many* scenario with different initial two competitors. This experiment was run 20 times with different seeds. AvGP, average Gaussian process; LMC, linear model of coregionalization; LVMOGP, latent variable multiouput Gaussian process; MOGP, multiouput Gaussian process.

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

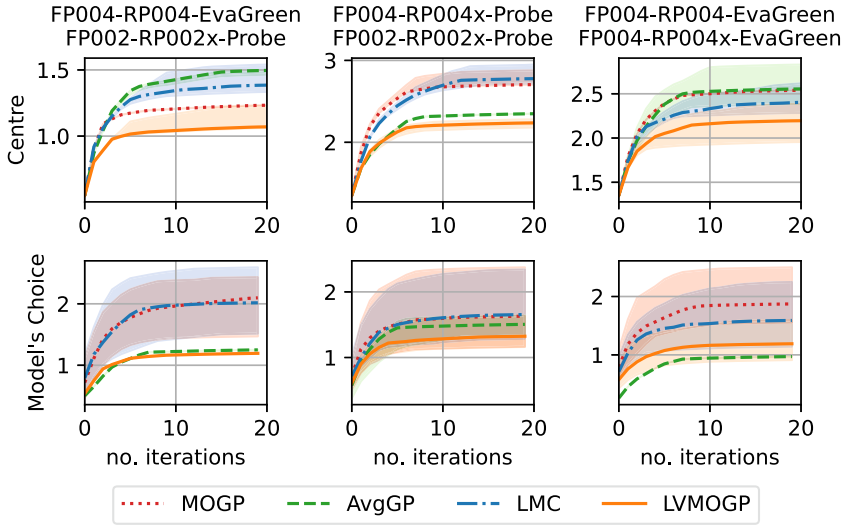


FIGURE G3 Results of penalized objective Bayesian optimization experiments for the *learning many* scenario with different initial two competitors. This experiment was run 20 times with different seeds. AvGP, average Gaussian process; LMC, linear model of coregionalization; LVMOGP, latent variable multiouput Gaussian process; MOGP, multiouput Gaussian process.

of two initial surfaces. We ran the *learning many* scenario of Bayesian optimization with 20 seeds for both the single objective and penalized optimization. We chose the *learning many* as we expect the initial surfaces to have more impact in this scenario than in the *one at a time* scenario. The results of these experiments are plotted in Figures G2 and G3. In these results, it is clear the initial surfaces make some difference to the models' performances, although the LVMOGP still performs the best most of the time. The differences in performance are most likely due to different amounts of initial data (depending on how much data we have for the initial surfaces) and differences in the similarity of the initial surfaces to the surfaces to be learned.

10970290, 2025, 1, Downloaded from https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/bit.28854 by Imperial College London, Wiley Online Library on [13/12/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License