

Multi-fidelity Batch Active Learning for Gaussian Process Classifiers

Murray Cutforth^a, Yiming Yang^b, Tiffany Fan^a, Serge Guillas^b, Eric Darve^a

^a*Mechanical Engineering, Stanford University*

^b*University College London*

Abstract

Many science and engineering problems rely on expensive computational simulations, where a multi-fidelity approach can accelerate the exploration of a parameter space. We study efficient allocation of a simulation budget using a Gaussian Process (GP) model in the binary simulation output case. This paper introduces Bernoulli Parameter Mutual Information (BPMI), a batch active learning algorithm for multi-fidelity GP classifiers. BPMI circumvents the intractability of calculating mutual information in the probability space by employing a first-order Taylor expansion of the link function. We evaluate BPMI against several baselines on two synthetic test cases and a complex, real-world application involving the simulation of a laser-ignited rocket combustor. In all experiments, BPMI demonstrates superior performance, achieving higher predictive accuracy for a fixed computational budget.

Keywords: Active Learning, Bayesian Optimization, Gaussian Process Classification, Multi-fidelity, Uncertainty Quantification

1. Introduction

In this paper, we consider the problem of optimally choosing where to run a set of simulations (given some parameter space which we wish to search over) in a bi-fidelity setting, and for binary simulation outputs. This scenario may arise across a variety of science and engineering applications, in which the simulation (or experiment) is prohibitively expensive but can be combined with a larger number of cheaper and less accurate simulations.

To be more specific, we are studying a laser-ignited methane-oxygen rocket combustor. This is a complex multi-physics problem, where each simulation yields a binary output: ignition or no ignition, which we model as a Bernoulli random variable. Our goal is to obtain the most accurate ignition probability map possible, given two simulation fidelities and a fixed computational budget which is allocated in batches. In some regions of parameter space, the simulation output exhibits stochasticity, making this a probability estimation problem rather than a classification problem. The batch active learning algorithm developed in this work is generally applicable to any multi-fidelity simulation framework with binary outcomes.

We further limit our scope to Gaussian Process classification models [1], due to their excellent built-in uncertainty quantification abilities. In many engineering applications, it is important to not only predict the quantity of interest, but also to have a robust estimate of the uncertainty on that prediction. We utilize the existing multi-fidelity Gaussian Process classification model of [2, 3] in this work.

After choosing a suitable bi-fidelity model, the challenge of intelligently exploring the parameter space remains. Given a limited computational budget, one must decide which new simulations to perform (and at which fidelity level) to maximally improve the model’s predictive accuracy. This is the problem of active learning (AL), also known as the sequential design of experiments or Bayesian optimization [4]. A key component of any AL strategy is the acquisition function, which quantifies the utility of querying a new data point. A particularly powerful class of acquisition functions is based on Mutual Information (MI), which seeks to select points that maximally reduce the uncertainty about a quantity of interest [5].

Email addresses: mcc4@stanford.edu (Murray Cutforth), darve@stanford.edu (Eric Darve)

In this work, we develop a novel active learning strategy tailored for bi-fidelity Gaussian Process classification models. Our primary contribution is a new acquisition function, termed Bernoulli Parameter Mutual Information (BPMI), which efficiently approximates the joint mutual information between sets of multi-fidelity Bernoulli parameters in a probability estimation setting. We demonstrate through numerical experiments that our proposed BPMI strategy outperforms several other mutual information and non-mutual information based active learning strategies. Our numerical experiments focus on estimating the probability map (or classification boundary) in low dimensional (2D) parameter spaces.

1.1. Related Work

Multi-fidelity Gaussian Processes. There is a large literature of multi-fidelity Gaussian Process models, starting with the widely-used autoregressive structure of [2]. However, the majority of this literature assumes continuous, Gaussian-distributed outputs. Adapting these models to classification settings is non-trivial. The model we use was proposed in [3].

Active learning. Mutual information as an active learning acquisition function was introduced in [5]. It has been widely used [6], and is generally optimal if it can be efficiently calculated [7]. There is a significant literature of batched (but single fidelity) active learning methods such as [8], but a survey of these methods is outside our scope. To our knowledge, there is only one prior batched, multi-fidelity active learning method in the literature [9], which considers real-valued simulation outputs. Our work follows from [9], and we develop an efficient algorithm which is specialized to binary outputs and Gaussian Process models.

2. Method

We first re-introduce the bi-fidelity Gaussian Process classification model of [3], before presenting the efficient mutual information-based active learning method which is the focus of this work.

2.1. Bi-Fidelity Gaussian Process Classification Model

To model the binary classification problem with data from two fidelities, we follow the non-linear auto-regressive scheme originally proposed in [2] and adapted for classification in [3]. The core of the model is built upon latent Gaussian Processes, but due to the Bernoulli likelihood used for binary data, exact inference is not possible. Unlike [3], we use a variational inference framework. This design choice was made for efficiency reasons, because the model inference step is repeated a large number of times in the active learning method we present.

2.1.1. Model Specification

Let $\mathcal{D}_L = \{(\mathbf{x}_{L,i}, y_{L,i})\}_{i=1}^{N_L}$ and $\mathcal{D}_H = \{(\mathbf{x}_{H,j}, y_{H,j})\}_{j=1}^{N_H}$ be the training datasets for the low- and high-fidelity sources, respectively, where $\mathbf{x} \in \mathbb{R}^d$ are input parameters and $y \in \{0, 1\}$ are the binary class labels. We introduce two latent functions, $f_L(\mathbf{x})$ and $f_H(\mathbf{x})$. The relationship between the observed binary labels and these latent functions is defined through a probit link function (the cumulative distribution function of the standard normal distribution, $\Phi(\cdot)$):

$$p(y_L = 1|\mathbf{x}) = \Phi(f_L(\mathbf{x})) \quad (1)$$

$$p(y_H = 1|\mathbf{x}) = \Phi(f_H(\mathbf{x})) \quad (2)$$

The bi-fidelity structure is established through an auto-regressive model connecting the two latent functions:

$$f_H(\mathbf{x}) = \rho f_L(\mathbf{x}) + \delta(\mathbf{x}) \quad (3)$$

where ρ is a learnable scalar parameter that captures the correlation between the fidelities, and $\delta(\mathbf{x})$ is a discrepancy function that models the difference between the scaled low-fidelity function and the high-fidelity function.

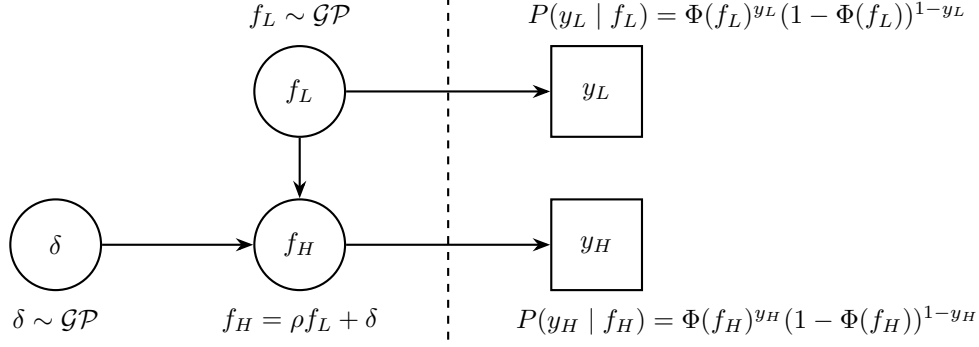


Figure 1: Bi-fidelity Gaussian Process classification model due to [3] used in this work. Variables on the left of the dashed line are hidden (latents), while y_L and y_H are the binary observed outcomes.

We place independent Gaussian Process priors on the low-fidelity latent function $f_L(\mathbf{x})$ and the discrepancy function $\delta(\mathbf{x})$:

$$f_L(\mathbf{x}) \sim \mathcal{GP}(m_L(\mathbf{x}), k_L(\mathbf{x}, \mathbf{x}')) \quad (4)$$

$$\delta(\mathbf{x}) \sim \mathcal{GP}(m_\delta(\mathbf{x}), k_\delta(\mathbf{x}, \mathbf{x}')) \quad (5)$$

In our implementation, both GPs use a constant mean function, $m(\mathbf{x}) = c$, and a scaled Radial Basis Function (RBF) kernel:

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\ell^2}\right) \quad (6)$$

where the output-scale σ^2 and the lengthscale ℓ are hyperparameters. Due to the independence of the priors on f_L and δ , the joint prior distribution over the latent functions is also a Gaussian Process. Define the vector-valued function $\mathbf{f}(\mathbf{x}) = [f_L(\mathbf{x}), f_H(\mathbf{x})]^T$. This joint process is fully specified by its vector-valued mean function $\mathbf{m}(\mathbf{x})$ and its matrix-valued covariance function (or kernel matrix) $\mathbf{K}(\mathbf{x}, \mathbf{x}')$:

$$\begin{bmatrix} f_L(\mathbf{x}) \\ f_H(\mathbf{x}) \end{bmatrix} \sim \mathcal{GP}(\mathbf{m}(\mathbf{x}), \mathbf{K}(\mathbf{x}, \mathbf{x}')), \quad (7)$$

where the mean function is:

$$\mathbf{m}(\mathbf{x}) = \begin{bmatrix} \mathbb{E}[f_L(\mathbf{x})] \\ \mathbb{E}[f_H(\mathbf{x})] \end{bmatrix} = \begin{bmatrix} m_L(\mathbf{x}) \\ \rho m_L(\mathbf{x}) + m_\delta(\mathbf{x}) \end{bmatrix}, \quad (8)$$

and the covariance function is:

$$\mathbf{K}(\mathbf{x}, \mathbf{x}') = \begin{bmatrix} k_L(\mathbf{x}, \mathbf{x}') & \rho k_L(\mathbf{x}, \mathbf{x}') \\ \rho k_L(\mathbf{x}, \mathbf{x}') & \rho^2 k_L(\mathbf{x}, \mathbf{x}') + k_\delta(\mathbf{x}, \mathbf{x}') \end{bmatrix}. \quad (9)$$

A graphical illustration of this model is provided in Figure 1.

2.1.2. Variational Inference and Training

We use a variational inference approach to find an approximate posterior, implemented in the **gpytorch** library [10]. We define variational distributions over the values of the latent functions at a set of inducing locations $\mathbf{Z}_L$ and $\mathbf{Z}_\delta$:

$$q(\mathbf{u}_L) = \mathcal{N}(\mathbf{m}_{u_L}, \mathbf{S}_{u_L}) \quad \text{where} \quad \mathbf{u}_L = f_L(\mathbf{Z}_L) \quad (10)$$

$$q(\mathbf{u}_\delta) = \mathcal{N}(\mathbf{m}_{u_\delta}, \mathbf{S}_{u_\delta}) \quad \text{where} \quad \mathbf{u}_\delta = \delta(\mathbf{Z}_\delta) \quad (11)$$

Here, $\mathbf{m}$ and $\mathbf{S}$ are the variational parameters (mean vectors and Cholesky factors) to be optimized. Training consists of maximizing the Evidence Lower Bound (ELBO), $\mathcal{L}_{\text{ELBO}}$, with respect to the variational parameters, the model hyperparameters, and the scaling factor ρ . The ELBO is given by:

$$\mathcal{L}_{\text{ELBO}} = \mathbb{E}_{q(f_L)}[\log p(\mathcal{D}_L|f_L)] + \mathbb{E}_{q(f_H)}[\log p(\mathcal{D}_H|f_H)] - \text{KL}[q(\mathbf{u}_L)||p(\mathbf{u}_L)] - \text{KL}[q(\mathbf{u}_\delta)||p(\mathbf{u}_\delta)] \quad (12)$$

where the expectation $\mathbb{E}_{q(f_H)}$ is taken over the distribution of f_H induced by $q(f_L)$ and $q(\delta)$ via Equation 3. The expected log-likelihood terms are computed using Gauss-Hermite quadrature.

Additionally, to prevent overfitting an L2 regularization term is added to the objective function, which is equivalent to placing a broad Gaussian prior on the kernel hyperparameters and minimizing the negative log-posterior:

$$\mathcal{L}_{\text{final}} = -\mathcal{L}_{\text{ELBO}} + \lambda \sum_{\theta \in \Theta} (\theta - \theta_{\text{prior}})^2 \quad (13)$$

where Θ is the set of all kernel hyperparameters and λ is a regularization coefficient (default value 10^{-2}). The model is trained by minimizing $\mathcal{L}_{\text{final}}$ using the Adam optimizer (learning rate 10^{-3}). To improve robustness against local minima, the training process is repeated for three random initializations, and the model yielding the best final ELBO is selected.

2.1.3. Prediction

After training, the model can be used to make predictions at new test locations $\mathbf{x}^*$. The approximate predictive posterior for the high-fidelity latent function, $q(f_H(\mathbf{x}^*))$, is constructed based on the autoregressive model (Equation 3):

$$\mathbb{E}[f_H(\mathbf{x}^*)] = \rho \mathbb{E}[f_L(\mathbf{x}^*)] + \mathbb{E}[\delta(\mathbf{x}^*)] \quad (14)$$

$$\text{Var}[f_H(\mathbf{x}^*)] = \rho^2 \text{Var}[f_L(\mathbf{x}^*)] + \text{Var}[\delta(\mathbf{x}^*)] \quad (15)$$

The final predictive probability for the high-fidelity class is obtained by marginalizing over the uncertainty in this latent posterior:

$$p(y_H = 1|\mathbf{x}^*, \mathcal{D}) = \int \Phi(f_H) q(f_H(\mathbf{x}^*)) df_H \quad (16)$$

This integral is again approximated numerically.

2.2. Batch acquisition function

Active learning aims to intelligently select new data points for labeling to improve model performance under a limited budget. In the *batched, multi-fidelity* setting, the goal is to select multiple queries without re-training the model, where each query is an (input location, fidelity) pair, $(\mathbf{x}, m)$, with $m \in \{L, H\}$. Let c_L and c_H be the costs associated with acquiring a low- or high-fidelity label, respectively. Given a total budget B for an active learning step, we seek to find the batch of queries $\mathcal{Q} = \{(\mathbf{x}_i, m_i)\}_{i=1}^k$ that maximizes our objective, subject to the constraint that $\sum_{i=1}^k c_{m_i} \leq B$.

2.2.1. Mutual Information as an Acquisition Function

We adopt an information-theoretic acquisition function based on Mutual Information (MI), following the batched multi-fidelity active learning approach in [9]. A variety of different MI-based acquisition functions are developed and compared in this work, and these are defined later. For now, denote the acquisition function as $\alpha(\mathcal{Q})$. Finding the optimal batch is a combinatorial NP-hard problem. However, mutual information is known to be a *submodular* function.

A set function F is submodular if it exhibits a diminishing returns property: for any sets $A \subseteq B$ and an element $x \notin B$, the marginal gain of adding x to B is no more than adding it to A , i.e., $F(A \cup \{x\}) - F(A) \geq F(B \cup \{x\}) - F(B)$. For maximizing a monotonic submodular function under a budget constraint, a simple greedy algorithm is guaranteed to find a solution that is within a constant factor $(1 - 1/e)$ of the optimal solution [11].

This allows us to construct the batch sequentially. Starting with an empty batch $\mathcal{Q}_0 = \emptyset$, at each step k , we add the single query $(\mathbf{x}^*, m^*)$ that offers the largest marginal information gain per unit cost:

$$(\mathbf{x}^*, m^*) = \arg \max_{(\mathbf{x}, m)} \frac{\alpha(\mathcal{Q}_k \cup \{(\mathbf{x}, m)\}) - \alpha(\mathcal{Q}_k)}{c_m} \quad (17)$$

This process is repeated until the budget is exhausted. As discussed in [9], in the multi-fidelity setting there is a minor caveat. The algorithm continues until a proposal is found which will exceed the budget; we select this and then terminate. This preserves the near-optimality property in the multi-fidelity setting.

2.2.2. Baseline: Latent Function Mutual Information (LFMI)

Directly adapting the approach in [9] to our bi-fidelity Gaussian Process model yields an acquisition function designed to maximize the information gained about the high-fidelity latent function f_H across the entire input domain. This is practically estimated by computing the mutual information between the latent function values at the candidate batch $\mathcal{Q}$ and the high-fidelity latent function values at a fixed set of test locations $\mathcal{X}' = \{\mathbf{x}'_j\}_{j=1}^{N'}$, which are sampled uniformly from the domain to represent the space over which we want to reduce uncertainty. The acquisition function is thus the joint mutual information:

$$\alpha_{\text{LFMI}}(\mathcal{Q}) = I(\mathbf{f}_{\mathcal{Q}}; \mathbf{f}_{H, \mathcal{X}'}), \quad (18)$$

where $\mathbf{f}_{\mathcal{Q}}$ is the vector of latent function values corresponding to the queries in $\mathcal{Q}$ (i.e., $f_L(\mathbf{x}_i)$ if $m_i = L$ and $f_H(\mathbf{x}_i)$ if $m_i = H$), and $\mathbf{f}_{H, \mathcal{X}'}$ are the latent values of the high-fidelity function at the test locations $\mathcal{X}'$.

For a joint Gaussian distribution, this MI has a convenient analytical form based on the log-determinants of the covariance matrices:

$$I(\mathbf{A}; \mathbf{B}) = \frac{1}{2} (\log \det(\mathbf{\Sigma}_{\mathbf{A}}) + \log \det(\mathbf{\Sigma}_{\mathbf{B}}) - \log \det(\mathbf{\Sigma}_{\mathbf{A}, \mathbf{B}})) \quad (19)$$

where $\mathbf{\Sigma}_{\mathbf{A}}$, $\mathbf{\Sigma}_{\mathbf{B}}$, and $\mathbf{\Sigma}_{\mathbf{A}, \mathbf{B}}$ are the covariance matrices of the random vectors $\mathbf{A}$, $\mathbf{B}$, and their joint distribution, respectively.

These quantities can be efficiently computed from our bi-fidelity GPC model as follows. We construct the joint posterior distribution over the target vector $\mathbf{f}_{\text{joint}} = [\mathbf{f}_{\mathcal{Q}}^T, \mathbf{f}_{H, \mathcal{X}'}^T]^T$ from the base independent posteriors $q(f_L)$ and $q(\delta)$. Let $\mathcal{X}_L^{\text{eval}}$ be the set of unique input locations required for all f_L evaluations, and $\mathcal{X}_{\delta}^{\text{eval}}$ be the set for all δ evaluations. We form a base random vector $\mathbf{g}$ by concatenating the latent posteriors at these unique points:

$$\mathbf{g} = \begin{bmatrix} \mathbf{f}_L(\mathcal{X}_L^{\text{eval}}) \\ \delta(\mathcal{X}_{\delta}^{\text{eval}}) \end{bmatrix} \quad (20)$$

Due to the independence of the variational posteriors for f_L and δ , the distribution of this base vector is a block-diagonal Gaussian:

$$\mathbf{g} \sim \mathcal{N} \left(\begin{bmatrix} \boldsymbol{\mu}_{f_L} \\ \boldsymbol{\mu}_{\delta} \end{bmatrix}, \begin{bmatrix} \mathbf{\Sigma}_{f_L} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_{\delta} \end{bmatrix} \right), \quad (21)$$

where $\boldsymbol{\mu}_{f_L}$ and $\mathbf{\Sigma}_{f_L}$ are the mean and covariance of $q(f_L(\mathcal{X}_L^{\text{eval}}))$, and likewise for δ .

The target vector $\mathbf{f}_{\text{joint}}$ is a linear transformation of this base vector, $\mathbf{f}_{\text{joint}} = \mathbf{T}\mathbf{g}$. The transformation matrix $\mathbf{T}$ assembles the final outputs from the base components according to Equation 3. It takes the block matrix form:

$$\mathbf{T} = \begin{bmatrix} \mathbf{S}_{L, Q} & \mathbf{0} \\ \rho \mathbf{S}_{L, H} & \mathbf{S}_{\delta, H} \\ \rho \mathbf{S}_{L, X'} & \mathbf{S}_{\delta, X'} \end{bmatrix} \quad (22)$$

Here, the $\mathbf{S}$ matrices are sparse selection matrices of 0s and 1s. For instance, $\mathbf{S}_{L, Q}$ maps the entries of $\mathbf{f}_L(\mathcal{X}_L^{\text{eval}})$ to their corresponding positions in the low-fidelity query portion of $\mathbf{f}_{\text{joint}}$. The first block row constructs the low-fidelity query outputs (f_L), while the second and third rows construct the high-fidelity outputs ($f_H = \rho f_L + \delta$) for the high-fidelity queries and test points, respectively. The resulting distribution

of the target vector is also Gaussian, $q(\mathbf{f}_{\text{joint}}) = \mathcal{N}(\mathbf{T}\boldsymbol{\mu}_g, \mathbf{T}\boldsymbol{\Sigma}_g\mathbf{T}^T)$, from which all necessary covariance matrices for the MI calculation can be extracted.

However, this approach has a significant limitation for classification and probability estimation tasks. The probit function used to transform the unbounded latents into a probability saturates if $f_H \gg 0$ or $f_H \ll 0$. This is not accounted for when measuring MI between the latent functions. The LFMI acquisition function may favor acquiring points in regions where the model is already highly confident, as the latent uncertainty can still be large. This is inefficient, as reducing uncertainty in these saturated regions does not improve knowledge of the Bernoulli parameter.

2.2.3. Proposed Method: Bernoulli Parameter Mutual Information (BPMI)

To address the saturation issue, we propose a novel acquisition function that directly approximates the information gain with respect to the classification probabilities, which we denote as Bernoulli parameters $\mathbf{p} = \Phi(\mathbf{f})$. The objective is to maximize $I(\mathbf{p}_Q; \mathbf{p}_{H, \mathcal{X}'})$. Since the probit transformation $\Phi(\cdot)$ is non-linear, the distribution of $\mathbf{p}$ is non-Gaussian, and its MI is intractable. We circumvent this issue by applying a first-order Taylor expansion of the link function around the posterior mean of the latent functions, thereby linearizing the link function and enabling an efficient approximation of $I(\mathbf{p}_Q; \mathbf{p}_{H, \mathcal{X}'})$. We now describe this procedure in more detail.

Let $\mathbf{f}_{\text{joint}} = [\mathbf{f}_Q, \mathbf{f}_{H, \mathcal{X}'}]^T$ be the joint latent vector with posterior $q(\mathbf{f}_{\text{joint}}) = \mathcal{N}(\boldsymbol{\mu}_f, \boldsymbol{\Sigma}_f)$. The corresponding vector of probabilities $\mathbf{p}_{\text{joint}}$ is approximated as:

$$\mathbf{p}_{\text{joint}} \approx \Phi(\boldsymbol{\mu}_f) + (\mathbf{f}_{\text{joint}} - \boldsymbol{\mu}_f) \odot \Phi'(\boldsymbol{\mu}_f) \quad (23)$$

where $\odot$ is the element-wise product and $\Phi'(\cdot)$ is the derivative of the probit link function, which is the standard normal probability density function, $\phi(\cdot)$. This linearization leads to a Gaussian approximation for the distribution of the Bernoulli parameters, $q(\mathbf{p}_{\text{joint}}) \approx \mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)$, where:

$$\boldsymbol{\mu}_p = \Phi(\boldsymbol{\mu}_f) \quad (24)$$

$$\boldsymbol{\Sigma}_p \approx \mathbf{D}\boldsymbol{\Sigma}_f\mathbf{D}^T \quad (25)$$

and $\mathbf{D}$ is a diagonal matrix containing the derivatives evaluated at the mean: $\mathbf{D} = \text{diag}(\phi(\boldsymbol{\mu}_f))$.

The MI can now be calculated analytically using this approximated Gaussian distribution over the probabilities. The key advantage of this formulation is that the derivative term $\phi(\boldsymbol{\mu}_f)$ approaches zero as $|\boldsymbol{\mu}_f|$ becomes large. Consequently, the elements of the transformed covariance matrix $\boldsymbol{\Sigma}_p$ will be small in regions where the classification probability is already saturated (close to 0 or 1). This naturally focuses the acquisition function on uncertain regions near the decision boundary, where the potential information gain is highest, leading to more efficient learning.

2.2.4. Sampling frequency

For problems with a Bernoulli-distributed outcome, a single observation at a location $\mathbf{x}$ provides limited information about the underlying success probability $p(\mathbf{x})$. To gain a more precise estimate of $p(\mathbf{x})$, it is advantageous to perform multiple trials at or near the same location. Our proposed BPMI method also adaptively determines the number of times a selected point should be sampled, which we refer to as the sampling frequency or number of repeats, N .

We propose a heuristic that bases the number of repeats on the model's current estimate of the aleatoric (data) uncertainty. The standard error of the sample mean from N samples of a Bernoulli random variable is $\sqrt{\frac{p(1-p)}{N}}$. To achieve a fixed standard error, N must be proportional to $p(1-p)$, which is maximised at $p = 0.5$. This further concentrates sampling in regions of maximum aleatoric uncertainty ($p \approx 0.5$).

When a point $\mathbf{x}$ is chosen by the greedy acquisition strategy, we first use the current model to predict the mean probability, $p_{\text{pred}} = \mathbb{E}[p(\mathbf{x})]$. The number of repeats N is then calculated as:

$$N(p_{\text{pred}}) = \text{round} \left(\frac{N_{\text{max}} - 1}{4} \cdot (p_{\text{pred}}(1 - p_{\text{pred}}) + 1) \right) \quad (26)$$

where $N_{\max}$ is a hyperparameter defining the maximum number of repeats allowed. Finally, as a practical implementation detail, a small amount of random jitter is added to the input coordinates of the repeated samples to avoid placing multiple query points at the exact same location, which can cause numerical issues (non-positive definiteness) in the GP model.

2.2.5. Additional baselines

We also compare our method against two non-MI baselines.

1. **Random Point Selection** is a simple strategy that does not use any information from the model. To form a batch, it sequentially selects a random point from each fidelity to query until the step's budget is spent.
2. **Maximum Uncertainty Point Selection** is a common active learning heuristic that queries points where the model is least confident. The acquisition score is a weighted sum of two uncertainty types. The first is epistemic uncertainty (model uncertainty), captured by the variance of the model's prediction for the Bernoulli success probability p . The second is aleatoric uncertainty (data uncertainty), captured by the entropy of the predicted Bernoulli distribution, which is highest when the predicted probability is 0.5. The relative importance of these two terms is controlled by a hyperparameter β (default 0.5). The strategy greedily selects one point from each fidelity until the budget is filled.

3. Numerical Experiments

3.1. Toy problems

We introduce two synthetic 2D binary classification problems to simulate multi-fidelity data. The input space for both problems is the unit square, $\mathbf{x} = (x_1, x_2) \in [0, 1]^2$. We define a probability field $p(\mathbf{x})$, and data at x is sampled according to $y(\mathbf{x}) \sim \text{Bernoulli}(p(\mathbf{x}))$. These probability fields are illustrated in Figure 2.

The low-fidelity data source, y_L , is common to both problems. It is defined by:

$$P(y_L = 1|\mathbf{x}) = \sigma(s(x_1) \cdot (x_2 - d_L(x_1))) \quad (27)$$

$$\text{where } d_L(x_1) = \frac{1}{3} \left(\cos\left(\frac{\pi x_1}{2}\right) + 1 \right) - 0.1 \quad (28)$$

$$\text{and } s(x_1) = \alpha(1 - 0.75x_1) \quad (29)$$

Here, $\sigma(z) = (1 + e^{-z})^{-1}$ is the sigmoid function and α is a base scale factor (set to $\alpha = 20$) that controls the overall sharpness of the boundary.

The high-fidelity source, y_H , is defined by transforming the LF decision boundary. We consider two scenarios:

1. *Linear Transformation.* The HF decision boundary $d_{H,\text{lin}}$ is a linear shift and scale of the LF boundary. The resulting probability is:

$$P(y_{H,\text{lin}} = 1|\mathbf{x}) = \sigma(s(x_1) \cdot (x_2 - d_{H,\text{lin}}(x_1))) \quad (30)$$

$$\text{where } d_{H,\text{lin}}(x_1) = 0.8d_L(x_1) + 0.3 \quad (31)$$

2. *Nonlinear Transformation.* The HF decision boundary $d_{H,\text{nlin}}$ is a more complex, nonlinear transformation of the LF boundary. The probability is:

$$P(y_{H,\text{nlin}} = 1|\mathbf{x}) = \sigma(s(x_1) \cdot (x_2 - d_{H,\text{nlin}}(x_1))) \quad (32)$$

$$\text{where } d_{H,\text{nlin}}(x_1) = d_L(x_1) + 0.2 \sin(3\pi x_1)(1 - x_1) + 0.1 \quad (33)$$

In our toy problems, we make the following design choices, which mimic our practical application. The cost ratio of the fidelities is set to 10 ($c_{LF} = 0.1$, $c_{HF} = 1$). We initialize our model using 50 LF and 25 HF runs, and then employ 5 active learning rounds, with a batch cost of 100 per round. Each algorithm is repeated 20 times from different initial data, which was sufficient to obtain converged statistics. The random seed is set such that for a given repeat each algorithm starts from identical data.

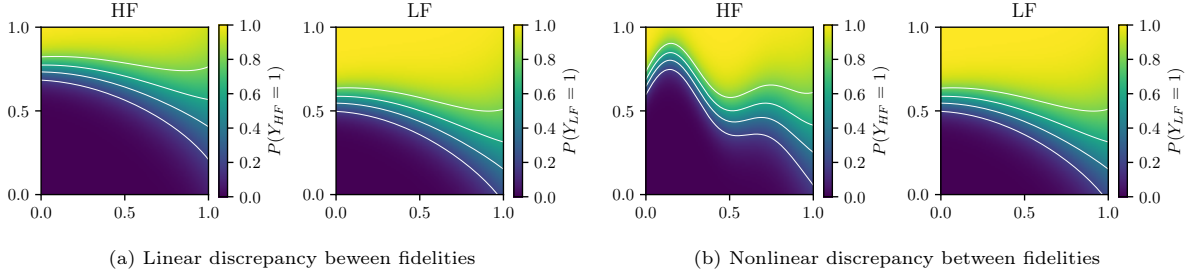


Figure 2: Toy problems used in numerical experiments, showing the probability as a function of a 2D parameter space. Observed data at a given point is the outcome of Bernoulli trial using the probability at that point illustrated above.

Metrics. We assess the performance of each active learning strategy by evaluating the model on a fixed test set of 10,000 points drawn from the high-fidelity distribution. We use two primary metrics.

First, we compute the Expected Log Predictive Probability (ELPP), a standard metric for evaluating probabilistic models. It measures the average log-likelihood of the high-fidelity test data $\mathcal{D}_{\text{test}} = \{(\mathbf{x}_i, y_{H,i})\}_{i=1}^{N_{\text{test}}}$ under the model's predictive distribution. For a binary classification model that predicts the probability $P(Y_H = 1|\mathbf{x}_i)$ as $\hat{p}_H(\mathbf{x}_i)$, the ELPP is defined as:

$$\text{ELPP} = \frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} [y_{H,i} \log \hat{p}_H(\mathbf{x}_i) + (1 - y_{H,i}) \log(1 - \hat{p}_H(\mathbf{x}_i))]. \quad (34)$$

Higher values indicated better performance.

Second, because we are in a synthetic setting where the true underlying probability field $p_{H,\text{true}}(\mathbf{x})$ is known, we can directly measure the model's accuracy in reconstructing this field. We compute the Mean Squared Error (MSE) between the model's predicted probability $\hat{p}_H(\mathbf{x})$ and the true probability $p_{H,\text{true}}(\mathbf{x})$ over the test set:

$$\text{MSE} = \frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} (\hat{p}_H(\mathbf{x}_i) - p_{H,\text{true}}(\mathbf{x}_i))^2. \quad (35)$$

Lower values indicate better performance.

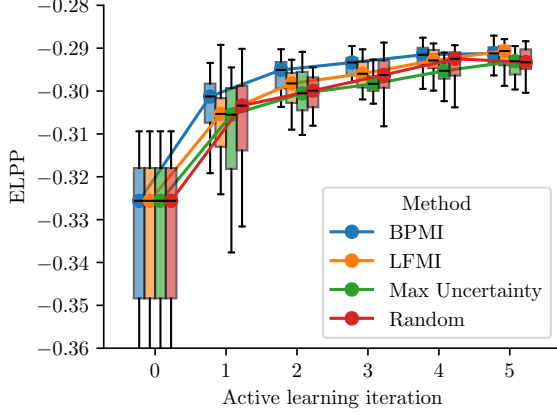
Results. Batch active learning algorithms are compared on these two toy problems in Figure 3. The proposed BPMP algorithm is consistently superior. However, in future work we plan to expand our evaluation problem set, which at present is limited.

3.2. Laser-ignited rocket combustor application

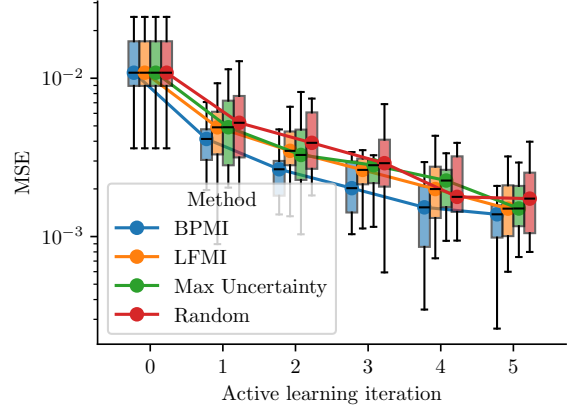
The motivating application of this work is a laser-ignited rocket combustor. Laser ignition is an ignition approach which could facilitate unlimited engine re-ignition. We consider a circular jet of oxygen with an annular flow of methane, and study the effect on ignition when the laser focal location is varied throughout the combustion chamber. In certain regions the laser spark interacts with the turbulent shear layer of the jet, and in these cases the ignition outcome can depend on the instantaneous turbulence, meaning that the ignition outcome is random (with an unknown underlying probability). We wish to quantify the ignition probability over a two-dimensional plane of symmetry through the combustor centreline.

Multi-fidelity methodology. The multi-fidelity methodology is described in [12], and briefly summarized here for completeness. Simulations are carried out using the Hypersonics Task-based Research (HTR) solver [13], modeling the compressible chemically reacting Navier-Stokes equations. The computational details concerning the solver parallelism, simulation domain, boundary conditions, and numerics can be found in previous work as part of PSAAP-III [14–16]. Note that the laser is modeled by an energy source term imposed on the fluid.

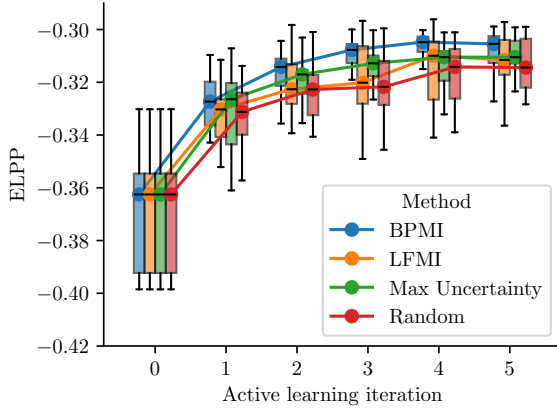
Two mesh resolutions are employed in this study. The low-fidelity mesh is composed of 2M grid points, and the high-fidelity counterpart is composed of 15M grid points. The same sub-grid scale models are



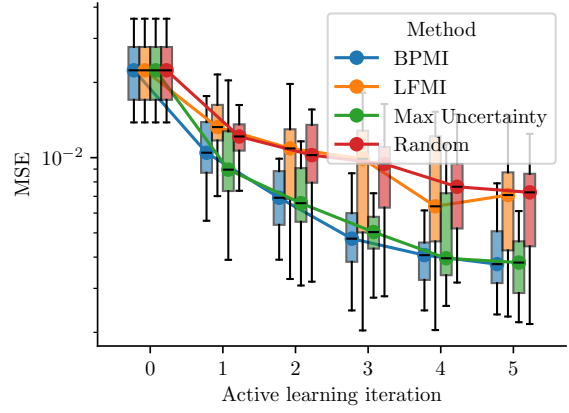
(a) Linear toy problem ELPP



(b) Linear toy problem MSE



(c) Nonlinear toy problem ELPP



(d) Nonlinear toy problem MSE

Figure 3: Comparison of active learning algorithms on the toy problems. BPMI consistently achieves higher ELPP and lower MSE than baseline methods across 20 independent runs.

employed in both cases [17]. A fully resolved direct numerical simulation would require approximately 100 billion grid points by comparison, but satisfactory agreement has been achieved between measured quantities in experiments [18] and output statistics from the present resolution used in high-fidelity simulations. The chemistry in the low-fidelity simulations is simplified to a 5-species, 3-step mechanism. Simulations were carried out on a cluster consisting of 4 Nvidia V100 GPUs per node. Each simulation on the 15M mesh took approximately 128 GPU-hours, while the 2M mesh used approximately 8 GPU-hours, a cost ratio of approximately 16.

Evaluation method. We ran 491 HF simulations, focused on the transition region where the ignition probability switches from 0 to 1. We use these observed outcomes as a test set and evaluate the ELPP given in Equation 34 using these points. Given computational constraints, we only compare our proposed BPMP active learning algorithm with our random selection baseline. We ran 3 active learning iterations, with a budget of 100 high-fidelity simulations per iteration (note that this budget is split between low- and high-fidelity simulations, so the actual number of high-fidelity runs per iteration is less than 100). Each active learning iteration therefore represents 12,800 GPU hours of compute.

Results. The results from applying these active learning problems to the laser-ignited combustor are shown in Figure 4. Once again, for a given number of active learning iterations, the proposed BPMP method results in a better performing model. The reason for this superior performance is evident in the sampling patterns shown in panels (b-e). BPMP concentrates both low- and high-fidelity queries near the transition boundary (while automatically balancing exploration of the rest of the parameter domain), whereas the Random strategy disperses them.

4. Conclusions

In this work, we introduced a novel batch active learning strategy, Bernoulli Parameter Mutual Information (BPMP), for binary data with a multi-fidelity Gaussian Process classification model. Our proposed method uses a linearization of the link function in used in Gaussian Process classification models in order to efficiently approximate a mutual information-based acquisition function. This formulation naturally focuses sampling on the decision boundary, where uncertainty in the predicted probability is highest and information gain is most valuable. We further propose a heuristic for adaptively determining the number of repeated samples at a query location, based on the current estimate of aleatoric uncertainty.

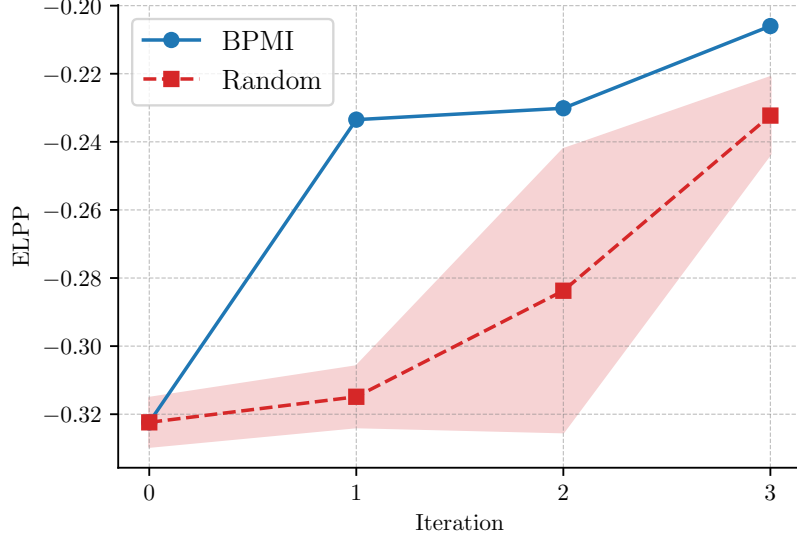
Through numerical experiments on both synthetic benchmarks and a complex, real-world application of a laser-ignited rocket combustor, we demonstrated the superior performance of BPMP. Compared to an equivalent latent-space MI approach, a maximum uncertainty heuristic, and random sampling, our method consistently produced more accurate predictive models for a given computational budget. The results confirm that BPMP more effectively allocates resources to delineate the classification boundary, leading to faster convergence and a better final model. This work provides an effective and efficient framework for the sequential design of experiments in bi-fidelity, binary-outcome problems.

Acknowledgments

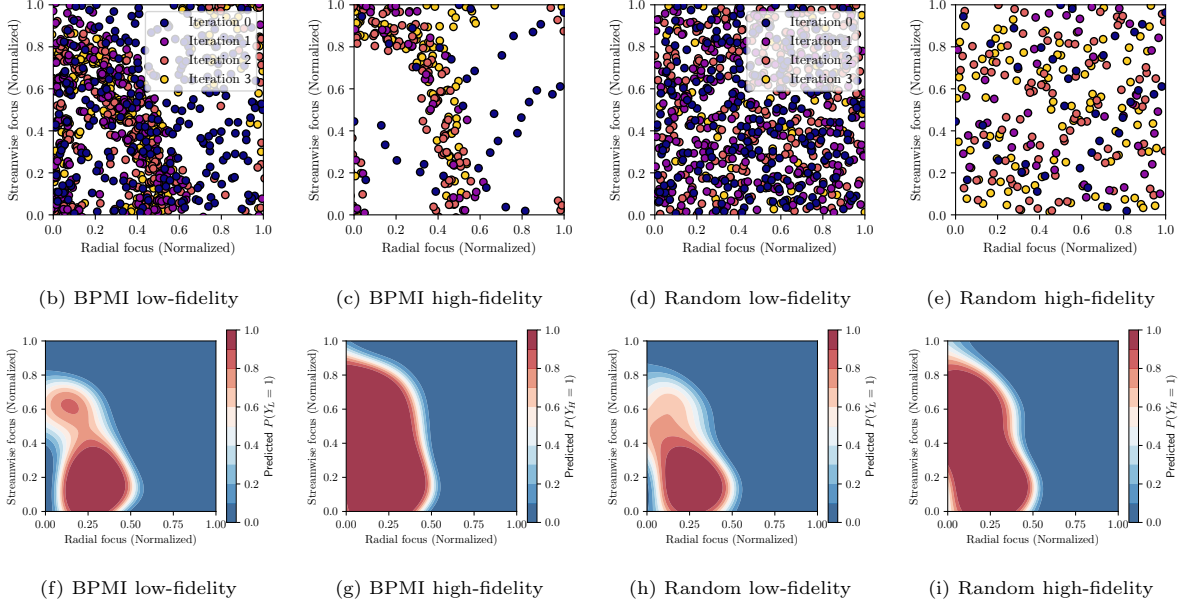
The authors acknowledge financial support from the US Department of Energy’s National Nuclear Security Administration via the Stanford PSAAP-III Center for the prediction of laser ignition of a rocket combustor (DE-NA0003968).

References

- [1] Hannes Nickisch, Carl Edward Rasmussen, et al. Approximations for binary gaussian process classification. *Journal of Machine Learning Research*, 9(10):2035–2078, 2008. 1
- [2] Marc C Kennedy and Anthony O’Hagan. Predicting the output from a complex computer code when fast approximations are available. *Biometrika*, 87(1):1–13, 2000. 1, 2



(a) Test set ELPP versus batch active learning iteration



(f) BPMI low-fidelity

(g) BPMI high-fidelity

(h) Random low-fidelity

(i) Random high-fidelity

Figure 4: Application of proposed active learning method to a complex multi-physics solver. In (a) the ELPP over a held-out test set is shown. The random strategy is repeated three times, with the shaded region representing the standard deviation of these repeats. Due to computational limitations the BPMI strategy was only run once. Panels (b) to (e) show the points selected for simulations by each approach. Panels (f) to (i) show the predicted ignition probabilities of the final model for each fidelity under the two active learning algorithms compared here.

- [3] Francisco Sahli Costabal, Paris Perdikaris, Ellen Kuhl, and Daniel E Hurtado. Multi-fidelity classification using gaussian processes: accelerating the prediction of large-scale computational models. *Computer Methods in Applied Mechanics and Engineering*, 357:112602, 2019. 1, 2, 3
- [4] Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998. 1
- [5] Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9(2), 2008. 1, 2
- [6] Joakim Beck and Serge Guillas. Sequential design with mutual information for computer experiments (mice): Emulation of a tsunami model. *SIAM/ASA Journal on Uncertainty Quantification*, 4(1): 739–766, 2016. 2
- [7] Shion Takeno, Hitoshi Fukuoka, Yuhki Tsukada, Toshiyuki Koyama, Motoki Shiga, Ichiro Takeuchi, and Masayuki Karasuyama. Multi-fidelity bayesian optimization with max-value entropy search and its parallelization. In *International Conference on Machine Learning*, pages 9334–9345. PMLR, 2020. 2
- [8] Andreas Kirsch, Joost Van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. *Advances in neural information processing systems*, 32, 2019. 2
- [9] Shibo Li, Jeff M Phillips, Xin Yu, Robert Kirby, and Shandian Zhe. Batch multi-fidelity active learning with budget constraints. *Advances in Neural Information Processing Systems*, 35:995–1007, 2022. 2, 4, 5
- [10] Jacob R Gardner, Geoff Pleiss, David Bindel, Kilian Q Weinberger, and Andrew Gordon Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. In *Advances in Neural Information Processing Systems*, 2018. 3
- [11] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978. 4
- [12] Murray Cutforth, Tiffany Fan, Tony Zahtila, Alireza Doostan, and Eric Darve. Bi-fidelity interpolative decomposition for multimodal data. *Submitted to Computer Methods in Applied Mechanics and Engineering*, 2025. 8
- [13] Mario Di Renzo, Lin Fu, and Javier Urzay. Htr solver: An open-source exascale-oriented task-based multi-gpu high-order code for hypersonic aerothermodynamics. *Computer Physics Communications*, 255:107262, 2020. 8
- [14] J Wang, M Di Renzo, C Williams, J Urzay, and G Iaccarino. Progress on laser ignition simulations of a ch4/o2 subscale rocket combustor using a multi-gpu task-based solver. *Center for Turbulence Research Annual Research Briefs*, pages 129–142, 2021. 8
- [15] Donatella Passiatore, Jonathan M Wang, Diego Rossinelli, Mario Di Renzo, and Gianluca Iaccarino. Computational study of laser-induced modes of ignition in a coflow combustor. *Flow, Turbulence and Combustion*, pages 1–25, 2024.
- [16] Tony Zahtila, Murray Cutforth, Davy J Brouzet, Donatella Passiatore, Diego Rossinelli, and Gianluca Iaccarino. Bi-fidelity data ensembles of a rocket ignition system with stochastic interpolative decomposition. In *AIAA SCITECH 2025 Forum*, page 2300, 2025. 8
- [17] Joseph Smagorinsky. General circulation experiments with the primitive equations: I. the basic experiment. *Monthly weather review*, 91(3):99–164, 1963. 10
- [18] Ryan Strelau, Mark Frederick, Will C Senior, Rohan Gejji, and Carson D Slabaugh. Modes of laser spark ignition of a model rocket combustor. In *AIAA SCITECH 2023 Forum*, page 2377, 2023. 10