Local validation of partially ordered models with an error-minimizing Bayes estimator

Kellan Moorse
California Institute of Technology
Pasadena, United States
kmoorse@caltech.edu

***Abstract*—We consider the problem of estimating the range of test conditions under which a simplified model—or set of simplified models—accurately approximates the behavior of a true system. We approach the problem by proposing a compact set of possible test conditions, and an unknown but samplable continous validity function over that set that quantifies the accuracy of the model under each possible condition. We propose a novel Bayes estimator that optimally directs function sampling to minimize the expected posterior misclassification rate of the valid set, which we call minimum posterior misclassification sampling (GP-MPM), and we show that the the method can be extended to approximate the valid sets of a partially ordered set of models, with sample complexity growing sublinearly with the number of models. In testing against several known 2-dimensional validity functions, the misclassification rate consistently reaches 0.1% in less than 50 samples—roughly an order of magnitude lower than undirected grid-based sampling.**

I. Introduction

The practice of developing a set of models to describe a system at various levels of precision is ubiquitous in science and engineering. For decades, researchers have been developing models of such high detail that they cannot feasibly be used for every application [18], [20]. But in many cases, the decision of when to dedicate the resources to use these highly detailed models, and when to settle for a simpler, cheaper one, is ill-defined.

As engineered systems like autonomous vehicles become more complex, it becomes increasingly difficult to effectively characterize their behavior; there may simply be too many behaviors of interest to test each one with the time and resources available. One solution to this problem is to check system performance through simulations, which can be run in a fraction of the time of a physical test. But high-fidelity simulations are still time- and computation-intensive, and simpified models may produce test results that are inconsistent with the behavior of the true system. It is not possible, in general, to analytically compute the conditions under which a simplified model diverges from the system's true behavior, so naively utilizing the results of the model carries a nontrivial risk of making erroneous conclusions about the behavior of the true system. However, even when model mismatch isn't analytically computable, it is generally possible to sample the model's performance relative to the true system behavior and make probabilistic statements about the mismatch.

Global Models. In this context, it is useful to separate methods for statistical modeling of system performance into two broad approaches: local and global. Global performance models assume that the system's performance is well-modeled as a random variable drawn from a single univariate continuous probability distribution. With the goal of inferring properties of this random variable, it is often useful to choose a parametric distribution with parameters θ, which is fully characterized by its cumulative distribution function $F(x|\theta)$. In particular, the noise in the measured outputs of an engineering system can generally be thought of as a sum of the noise from many component subsystems (as well as any noise inherent to the measurement itself), so by the central limit theorem, the measured noise is likely to be well-modeled by a normal distribution. This is useful because there are myriad methods for estimating the parameters of an analytical model distribution, from classical maximum likelihood es-

timation [11], to expectation-maximization [9], to further specialized methods for specific distributions [21], [23].

Once we have an estimate of the parameters of a distribution, it is trivial to find probabilistic bounds on the values taken by its corresponding random variable X directly from the definition of its cumulative distribution function $F(x|\theta)$:

$$F(x \,|\, \theta) = \mathbb{P}[X < x \,|\, \theta]$$

Further, it is generally possible to calculate confidence intervals on the parameter estimates produced by the above methods. In some simple cases, these calculations can be done analytically, but in many cases it is necessary to apply numerical methods, such as bootstrapping—where we treat the empirical distribution $\hat{F}(x)$ (i.e. the discrete set of already-drawn samples) as an approximation of the true distribution $F(x)$ and sample from $\hat{F}(x)$ to calculate summary statistics [10]. This allows us to calculate bounds of the form:

$$\mathbb{P}[F(x) < \epsilon] < \gamma$$

In other words, a bounded probability γ that a value drawn from the distribution of interest is smaller than ϵ. This is useful because it accounts for both the measurement noise in sampling the system's performance, and our uncertainty in the parameters of the distribution being sampled.

But if we are interested in this form of bounds, there are alternative methods that allow us to calculate them nonparametrically. Certain value-at-risk calculations remove the need to assume any particular model distribution, instead calculating bounds directly from the sampled values [14], [16]. For example, we can collect a finite set of n draws from an unknown distribution, call the minimum of that set x^*, and partition the distribution into two parts: $\{x < x^*\}$ and $\{x \geq x^*\}$. The newly binarized distribution is binomial, with n successes out of n trials. From the binomial PMF, we can calculate the bounds:

$$\gamma \geq \frac{ln(1-\epsilon)}{ln(n)}$$

where n is the number of samples collected, ϵ is the fraction of the original distribution's probability mass that lies below x^*, and γ is the probability that the bound on ϵ holds.

The results for all of these methods are sound, and they are useful in a wide variety of applications. But each one assumes that every observation of a system is drawn from a single univariate distribution, which means they are incapable of describing conditional behaviors. Many simplified models are designed to be conditionally valid—that is, they consistently perform well under some conditions, such as when a simplifying assumption holds, and consistently perform poorly when those conditions are not met. In order to capture and quantify these conditional properties, it is necessary to employ a more sophisticated model.

Local Models. Rather than dealing with parametric multivariate distributions, which aside from being less analytically tractable can be handled similarly to their univariate counterparts, we consider the special case of a conditional distribution $p(x|\phi)$, where samples x are drawn from different distributions depending on the conditions ϕ of the draw. For example, performance values for a linearized dynamical model will be drawn from a distribution with a higher mean if the system is near the fixed point of the linearization.

One useful analysis for a distribution of this class is to characterize its optima. There are multiple ways to efficiently sample the distribution to estimate both the location and the magnitude of the worst-case performance of the system—either with [8], [22] or without [2], [4] making any assumptions about the underlying distribution. If the task is to accept or reject a model wholesale, or to find individual failing conditions for the purpose of model development, these methods are ideal, as they efficiently pinpoint the worst-case performance of the system. But they inherently do an ineffienct job of characterizing the distribution away from its optimum. Uniform random and grid-based sampling of the distribution have the opposite problem, characterizing the whole distribution but becoming intractably sample-intensive in higher dimensions [6], [13].

Instead, we want to estimate the set of conditions ϕ under which $x > 0$ by sampling $p(x|\phi)$. Many powerful methods exist to estimate partitions over

a space by sampling it, from simple algorithms like k-means [1], to more sophisticated techniques like support vector machines and Gaussian process classifiers [5], [12], but these methods only consider discrete labels rather than real-valued samples, and they are unable to direct sampling to maximize information gain. They are designed for applications where data is plentiful and function evaluations are cheap, and they do not assume any structure in the discrete functions they are approximating.

What all of these existing methods lack is a means to efficiently sample a model's performance to determine the set of conditions under which it is a good approximator of its corresponding true system. Or, more generally, while there are methods that achieve optimal directed sampling in other regimes [15], there is no existing directed sampling method that exploits the inherent structure of continuous-valued function evalutions to efficiently estimate the set over which that function is positive. To that end, we develop a novel algorithm that is guaranteed at each step to greedily choose the action that minimizes expected posterior classification error. We further extend this algorithm to partially-ordered sets of models, generating a validity map that reports, for any given test condition, the lowest-order model that is still a good approximation of the true system, with high probability.

II. Definitions

Validity Metric. We begin by formalizing the conditions under which a model may be tested. Let Φ be a compact set in $\mathbb{R}^p$, where a vector $\phi \in \Phi$ is the set of conditions describing a given test—such as obstacle locations, road friction conditions, etc.—and let $v : \Phi \to \mathbb{R}$ be a continuous function which maps test conditions onto a scalar validity metric. This validity function is the function we seek to estimate, and it is assumed to exist for all $\phi \in \Phi$. While its values are unknown initially, its value can be sampled for any $\phi \in \Phi$.

To facilitate the later extension to multiple models, we define a pair of intermediate functions $[m^i : \Phi \to \mathbb{R}^q]_{i\in\{0,1\}}$ which map test conditions onto a vector of observations from model i. We also give a formulation of the validity function which is an explicit pairwise comparison between the observations from two models.

$$v^{01}(\phi) = v(m^0(\phi), m^1(\phi)) \tag{1}$$

One possible instantiation of this framework is that $m^0(\phi)$ and $m^1(\phi)$ correspond to traces of the position of the center of mass of an autonomous agent performing a task under some condition ϕ in experiment and in simulation, respectively. Then $v^{01}(\phi)$ may be the maximum norm difference between those position traces over the course of the task:

$$\begin{aligned} m^0(\phi) &= [x_1^0, ...x_T^0] \\ m^1(\phi) &= [x_1^1, ...x_T^1] \\ v^{01}(\phi) &= \max_{t\in\{1...T\}} \|x_t^0 - x_t^1\| \end{aligned} \tag{2}$$

In the following analysis, we use $v(\phi)$ as shorthand for $v^{01}(\phi)$, where model 0 is the true system, and model 1 is the simplified model of interest.

and it is initially unknown, but its value can be sampled for any ϕ. We define the model to be valid when $v(\phi) > 0$, and it follows that the valid set, or the set of ϕ for which the model is valid, is:

$$V = \{\phi \in \Phi \mid v(\phi) > 0\} \tag{3}$$

By this definition, the problem of set estimation is transformed into a problem of function estimation.

The validity function is sampled by running the same test on both a simplified model and the ground The function of interest $v(\phi)$ is continuous but otherwise unconstrained, which makes Gaussian process regression a strong method for estimation.

Gaussian Process Regression. We propose Gaussian processes as a model for estimating $v(\phi)$ for two reasons: first, complex engineering systems are often comprised of many interacting subsystems that each contribute to measurements of the system's properties. By the central limit theorem, uncertainty in those measurements is likely to be well-modeled by a Gaussian distribution. And second, real physical systems are generally continuous—i.e. small changes to their inputs result in small changes to their outputs. The Gaussian process model assumes these two properties, but no other structure in the

function being approximated, so we can make good approximations of a wide variety of functions.

Let $GP_\Phi(\mu(\cdot), k(\cdot,\cdot))$ be a Gaussian process, representing a set of random variables $[g(\phi)]_{\phi\in\Phi}$, such that each finite subset $[g(\phi)]_{i=1}^k$ is jointly Gaussian with mean and covariance:

$$\mathbb{E}[(g(\phi_i)] = \mu(\phi_i) \tag{4}$$

$$\mathbb{E}[(g(\phi_i) - \mu(\phi_i))(g(\phi_j) - \mu(\phi_j))] = k(\phi_i, \phi_j) \quad 1 \le i, j \le m,\ m \in \mathbb{N} \tag{5}$$

We use the prior $GP_\Phi(0, k(\cdot,\cdot))$, with mean zero and variance one, but the prior variance can be scaled to any positive value without loss of generality. Further, we use a typical squared exponential kernel:

$$k(\phi, \phi') = exp\left(\frac{\|\phi - \phi'\|_2}{2l^2}\right) \tag{6}$$

where l is a hyperparameter corresponding to the scale of the exponential. It is possible—and in many cases desireable—to estimate the hyperparameters of a Gaussian process to optimize its regression ([3], [7]), but this requires additional samples of the function, which we have assumed is the rate-limiting step of our algorithm. So for simplicity, we set l as a constant at the beginning of the procedure. We also assume normally-distributed measurement noise $\epsilon_t \sim \mathcal{N}(0, \lambda)$ added to each sample, where λ is also set as a hyperparameter.

Given a set of sample locations $A_t = (\phi_1, ..., \phi_t)$, we call the corresponding set of observed rewards $v_{1:t} = [v_1, ..., v_t]^T$. Then we define the kernel matrix and kernel vector:

$$\begin{aligned} K_t &= [k(\phi, \phi')]_{\phi,\phi'\in A_t} \\ k_t(\phi) &= [k(\phi_1, \phi), ..., k(\phi_t, \phi)]^T \end{aligned} \tag{7}$$

The observed reward vector and the true function $f(\phi)$ are then jointy Gaussian given A_t:

$$\begin{bmatrix} f(\phi) \\ v_{1:t} \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} k(\phi,\phi) & k_t(\phi)^T \\ k_t(\phi) & K_t + \lambda I \end{bmatrix}\right) \tag{8}$$

and the posterior distribution over f is $GP_\Phi(\mu_t(\cdot), k_t(\cdot,\cdot))$, where:

$$\mu_t(\phi) = k_t(\phi)^T (K_t + \lambda I)^{-1} v_{1:t} \tag{9}$$

$$k_t(\phi, \phi') = k(\phi, \phi') - k_t(\phi)^T (K_t + \lambda I)^{-1} k_t(\phi') \tag{10}$$

$$\sigma_t^2(\phi) = k_t(\phi, \phi) \tag{11}$$

III. Sampling Method

Efficient Posterior Sampling. In order to define a Bayes estimator, we need a computationally efficient representation of the posterior distribution. Below, we derive a recursive definition of the Gaussian process posterior.

Note that in eqn. 9, omitting the last data point x_t is equaivalent to removing the last elements of $k_t(\phi)$ and $v_{1:t}$, and removing the last row and column from K_t:

$$\begin{aligned} k_t(\phi)^T &= \begin{bmatrix} k(\phi_1, \phi) & \dots & k(\phi_t, \phi) \end{bmatrix} \\ k_{t-1}(\phi)^T &= \begin{bmatrix} k(\phi_1, \phi) & \dots & k(\phi_{t-1}, \phi) \end{bmatrix} \end{aligned} \tag{12}$$

$$K_t + \lambda I = \begin{bmatrix} k(\phi_1,\phi_1) & \dots & k(\phi_1,\phi_t) \\ \vdots & \ddots & \vdots \\ k(\phi_t,\phi_1) & \dots & k(\phi_t,\phi_t) \end{bmatrix}$$

$$K_{t-1} + \lambda I = \begin{bmatrix} k(\phi_1,\phi_1) & \dots & k(\phi_1,\phi_{t-1}) \\ \vdots & \ddots & \vdots \\ k(\phi_{t-1},\phi_1) & \dots & k(\phi_{t-1},\phi_{t-1}) \end{bmatrix} \tag{13}$$

$$v_{1:t} = \begin{bmatrix} v_1 \\ \vdots \\ v_t \end{bmatrix},\ v_{1:t-1} = \begin{bmatrix} v_1 \\ \vdots \\ v_{t-1} \end{bmatrix} \tag{14}$$

Using eqns. 12-14, and noting that $k(\phi, \phi) = 1$ for any ϕ, we can write each term of eqn. 9 at time t in terms of its expression at time $t-1$:

$$k_t(\phi)^T = \begin{bmatrix} k_{t-1}(\phi)^T, k(\phi_t, \phi) \end{bmatrix} \tag{15}$$

$$K_t = \begin{bmatrix} K_{t-1} & k_{t-1}(\phi_t) \\ k_{t-1}(\phi_t)^T & 1 \end{bmatrix}$$
$$K_t + \lambda I = \begin{bmatrix} K_{t-1} + \lambda I & k_{t-1}(\phi_t) \\ k_{t-1}(\phi_t)^T & 1 + \lambda \end{bmatrix} \quad (16)$$

$$v_t = \begin{bmatrix} v_{t-1} \\ v_t \end{bmatrix} \quad (17)$$

Labeling the block components of $K_t + \lambda I$, we can rewrite eqn. 9 as:

$$\mu_t(\phi) = \begin{bmatrix} k_{t-1}(\phi)^T & k(\phi_t, \phi) \end{bmatrix} R^{-1} \begin{bmatrix} v_{1:t-1} \\ v_t \end{bmatrix} \quad (18)$$

$$R = \begin{bmatrix} A & B \\ B^T & D \end{bmatrix}$$
$$A = K_{t-1} + \lambda I$$
$$B = k_{t-1}(\phi)$$
$$D = 1 + \lambda$$

And we note several useful definitions that follow from the block form of R:

$$\mu_{t-1}(\phi) = k_{t-1}(x)^T A^{-1} v_{1:t-1} \quad (19)$$
$$\frac{1}{\sigma^2_{t-1}(\phi_t) + \lambda} = (D - B^T A^{-1} B)^{-1} \quad (20)$$

Because the block matrix is Hermitian, its inverse is given in [17] to be:

$$\begin{bmatrix} A & B \\ B^T & D \end{bmatrix}^{-1} = \begin{bmatrix} E & F \\ G & H \end{bmatrix} \quad (21)$$

$$E = A^{-1} + A^{-1}B(D - B^T A^{-1} B)^{-1} B^T A^{-1}$$
$$F = -A^{-1}B(D - B^T A^{-1} B)^{-1}$$
$$G = -(D - B^T A^{-1} B)^{-1} B^T A^{-1}$$
$$H = (D - B^T A^{-1} B)^{-1}$$

Substituting eqn. 21 into eqn. 18 and multiplying out the matrices, we find a computable expression for the posterior mean $\mu(\phi)$ as a function of the prior distribution and a newly collected data point (ϕ_t, v_t):

$$\begin{aligned} \mu(\phi) &= k_{t-1}(\phi)^T A^{-1} v_{1:t-1} \\ &+ k_{t-1}(\phi)^T A^{-1} B(D - B^T A^{-1} B)^{-1} B^T A^{-1} v_{1:t-1} \\ &- k(\phi, \phi_t)(D - B^T A^{-1} B)^{-1} B^T A v_{1:t-1} \\ &- k_{t-1}(\phi)^T A^{-1} B(D - B^T A^{-1} B)^{-1} v_t \\ &+ k_(\phi, \phi_t)(D - B^T A^{-1} B) v_t \end{aligned} \quad (22)$$

Combining like terms and using eqns. 19 and 20 to convert back from block matrix to GP parameters, we arrive at a much simpler form:

$$\mu_t(\phi, \phi_t, v_t) = \mu_{t-1}(\phi) + \frac{k_{t-1}(\phi, \phi_t)}{\sigma^2_{t-1}(\phi_t) + \lambda}(v_t - \mu_{t-1}(\phi_t)) \quad (23)$$

Following an identical procedure but starting from eqn. 11, we also derive a recursive expression for the variance:

$$\sigma^2_t(\phi, \phi_t) = \frac{\lambda k_{t-1}(\phi, \phi_t)}{\sigma^2_{t-1}(\phi_t) + \lambda} \quad (24)$$

Bayes Estimator. Having derived conscise definitions for the posterior mean and variance of the Gaussian process, we propose a novel Bayes estimator which minimizes the expected posterior misclassification rate. Let $P^m_t(\phi)$ be the misclassification error rate at ϕ: the probability that the true function value $f(\phi)$ has a different sign than the posterior Gaussian process mean after t samples:

$$P^e_t(\phi) = \mathbb{P}[\text{sgn}\, f(\phi) \neq \text{sgn}\, \mu_t(\phi)] \quad (25)$$

From the Gaussian cumulative distribution function, this is equal to:

$$P^e_t(\phi) = \frac{1}{2}\text{erfc}\left(\frac{|\mu_t(\phi)|}{\sqrt{2}\sigma_t(\phi)}\right) \quad (26)$$

Now let $Z_{t-1}(\phi_t) \sim \mathcal{N}(\mu_{t-1}(\phi_t), \sigma_{t-1}(\phi_t))$ be a random variable corresponding to value observed upon sampling at ϕ_t, conditioned on the previously observed values $(A_{t-1}, v_{1:t-1})$. Then $P^e_t(\phi, \phi_t, Z_{t-1}(\phi_t))$ is the posterior misclassification error at ϕ, and its expectation is:

$$\mathbb{E}_z[P_t^e(\phi, \phi_t, Z_{t-1}(\phi_t)] = \frac{1}{2}\text{erfc}\Big(\frac{|\mu_t(\phi, \phi_t, Z_{t-1}(\phi_t)|}{\sigma_t^2(\phi, \phi_t)}\Big) \tag{27}$$

We define the global misclassification rate as the expectation of the local misclassification rate across all $\phi \in \Phi$—that is, the probability of drawing a misclassified ϕ from a uniform distribution over Φ:

$$g(\phi_t) = \frac{\int_\Phi \mathbb{E}_z[P_t^e(\phi, \phi_t, Z_{t-1}(\phi_t))]d\phi^{(1)}...d\phi^d\phi^{(p)}}{\int_\Phi d\phi^{(1)}...d\phi^d\phi^{(p)}} \tag{28}$$

The minimizer of $g(\phi_t)$ is, by definition, the point that minimizes the expected posterior global misclassification rate. However, the function $P_t^e(\phi, \phi_t, Z_{t-1}(\phi_t))$ has no assumed structure in ϕ beyond continuity, and numerically computing the nested integral over all $\phi \in \Phi$ and all $Z_{t-1}(\phi_t) \in \mathbb{R}$ is computationally expensive even in low dimensions—and prohibitively so in higher dimensions. So we implement a pair of approximations to make the computation tractable.

First, we consider the expectation over $P_t^e(\phi, \phi_t, Z_{t-1}(\phi_t))$. Because $Z_{t-1}(\phi_t)$ takes any real-numbered value, the expectation is given by:

$$\int_{\mathbb{R}} p(z)P_t^e(\phi, \phi_t, z)dz \tag{29}$$

where $p(z)$ is the probability density function over realizations z of $Z_{t-1}(\phi_t)$. We note that the expectation of a function of a gaussian random variable is equal to the function value at the mean of the random variable if (but not only if) the function is linear:

$$\begin{aligned} y(x) &= x \\ p(x) &= \frac{1}{\sigma\sqrt{2\pi}}\, exp\Big(-\frac{1}{2}\big(\frac{x-\mu}{\sigma}\big)^2\Big) \\ \mathbb{E}[y(x)] &= \int_{\mathbb{R}} p(x)y(x)dx = y(\mu) \end{aligned} \tag{30}$$

Eqn. 23 shows that $\mu_t(\phi)$ depends linearly on $Z_{t-1}(\phi_t)$, eqn. 27 shows that the argument of erfc depends linearly on $\mu_t(\phi)$ except precisely at zero, and the erfc function itself is smooth, approaching linearity as the absolute value of its argument gets large. So, if we replace $\mathbb{E}_z[P_t^e(\phi, \phi_t, Z_{t-1}(\phi_t))]$ with $P_t^e(\phi, \phi_t, \mu_{t-1}(\phi_t))$, the approximation is very good for large $|\mu_t|/\sigma_t$. The negative discontinuity in the derivative of erfc$(|\cdot|)$ at zero means that we slightly underestimate P_t^e at small $|\mu_t|/\sigma_t$, but because the Bayes estimator only considers relative values of the loss function, this has a minimal effect on its chosen actions.

Because we now assume that, in expectation, each newly collected sample at ϕ_t lands on $\mu_{t-1}(\phi_t)$, the difference term in eqn. 23 is identically zero and $\mu_{t-1}(\phi) = \mu_t(\phi)$ for all $\phi \in \Phi$. So there is no need to calculate the expected posterior mean $\mathbb{E}_z[\mu_t(\phi)]$, reducing the computation time of the expectation calculation by roughly half.

Second, we note that for our chosen kernel function $k(\phi, \phi') = exp(\|\phi - \phi'\|_2/2l^2)$, the covariance asyptotically approaches zero as the distance between the points grows large:

$$\begin{aligned} \lim_{s\to\infty} |\sigma_t^2(\phi) - \sigma_t^2(\phi')| = 0 \\ s = \|\phi - \phi'\|_2 \end{aligned} \tag{31}$$

The squared-exponential has particularly light tails, so the covariance can be treated as negligible for any points more than a small number of scale-lengths apart. This means that instead of integrating the expected posterior misclassification over all of Φ, we can integrate only over a p-ball around ϕ_t, with radius αl. α is left as a hyperparameter of the method, but $\alpha = 2$ is a reasonable choice for low-dimensional problems, capturing 95% of the kernel's mass in the one-dimensional case. In higher dimensions, the same α captures a smaller proportion of the kernel's mass, so it may be necessary to choose a larger value.

Incorporating these approximations, and omitting the denominator of eqn. 28 which does not depend on ϕ_t and only acts as a normalizing factor, we arrive at a simplified loss function whose optimizer closely approximates that of the true Bayes estimator:

$$\tilde{g}(\phi_t) = \int_B P_t^e(\phi, \phi_t, \mu_{t-1}(\phi_t)) d\phi^{(1)} ... d\phi^{(p)} \quad (32)$$
$$B = \{\phi \in \Phi \mid \|\phi - \phi_t\|_2 < \alpha l\}$$

Because this loss function is fast to compute, we can apply it iteratively to estimate the valid set of a function. However, the optimum is undefined for the prior $GP(0, k)$, so it must be initalized with at least one sample collected using a different method. In practice, the initialization is a small set of t_0 points drawn from a uniform distribution over Φ. The full procedure is shown in alg. 1.

Algorithm 1: GP-MPM: Single Model

Input: Prior GP(0,k), parameters λ, α, t_0
Set $A_{t_0} = [\phi_j \sim U(\Phi)]_{j=1}^{t_0}, v_{1:t_0} = [f(\phi_j) + \epsilon_j]_{j=1}^{t_0}$
for *t=1,2,3...T* **do**
  Choose
    $\phi_t = \underset{\phi \in \Phi}{\operatorname{argmin}} \int_B P_t^e(\phi, \phi_t, \mu_{t-1}(\phi_t)) d\phi^1 ... d\phi^p$
      where $B = \{\phi \in \Phi \mid \|\phi - \phi_t\|_2 < \alpha l\}$
  Observe validity $v_t = f(\phi_t) + \epsilon_t$
  Perform update to get μ_t and σ_t using eqns. 9–11
end for

IV. Model Sets

From the approximate Bayes estimator in eqn. 32, it is possible to extend the method to estimate valid sets for a set of models. Let $M = \{m^i\}_{i=0}^n$ be a finite set of models, each of which is a function mapping conditions to measurements:

$$[m^i : \Phi \to \mathbb{R}^q]_{i=0}^n \quad (33)$$

and we generalize the intermediate function in eqn. **??** to map measurements from any two models onto a scalar validity metric:

$$v^{ij}(\phi) = \tilde{v}(m^i(\phi), m^j(\phi)) \quad (34)$$

Because models are always compared against the true system behavior $m^0(\phi)$, we define the shorthand $v^j(\phi) = v^{0j}(\phi)$, and following the convention of the single-model case:

$$V^i = \{\phi \in \Phi \mid v^i(\phi) > 0\} \quad (35)$$

We then assume that the set of models can be organized into a partial order, where a model being greater in the order means that its validity is greater under all conditions:

$$m^i > m^j \iff v^i(\phi) > v^j(\phi) \ \forall \phi \in \Phi \quad (36)$$

This futher implies that the valid set for a given model is a superset of the valid set of each lesser model in the order:

$$m^i > v^j \implies \bigcap_{i|m^i > m^j} V^i \supset V^j \quad (37)$$

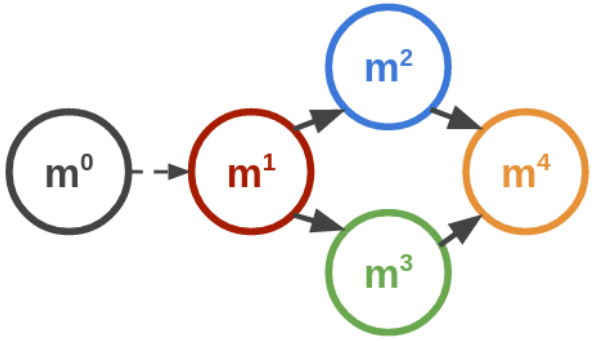

Fig. 1. An example model graph with a nontrivial order, asserting that each model performs better than the model(s) below it.

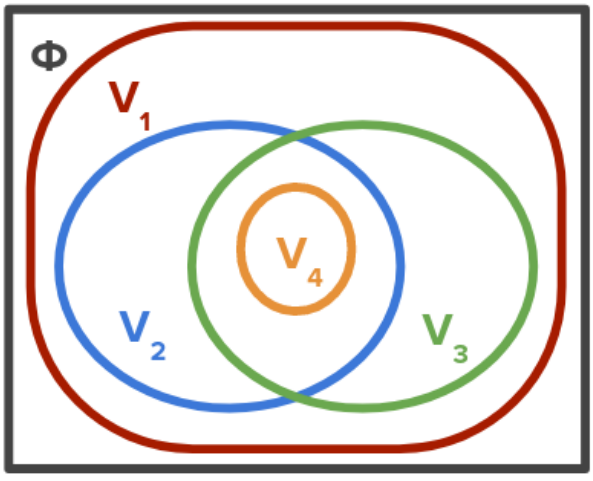

Fig. 2. A possible validity map corresponding to the ordered graph in fig. 1. Note that each model's valid set is a subset of the valid sets of any models above it in the order.

The benefits of treating the models as a set are twofold: first, the rate-limiting step of the procedure is measuring the true system m^0, and if we have

Algorithm 2: GP-MPM: Model Set

Input: Priors $[GP_i(0,k)]_{i=1}^{n}$, parameters λ, α, t_0
Set $A_{t_0} = [\phi_i \sim U(\Phi)]_{i=1}^{t_0}, [v_{1:t_0}^j = [f(\phi_i) + \epsilon_i]_{i=1}^{t_0}]_{j=1}^{n}$
for *i=1,2,3...n* **do**
  for *t=1,2,3...T* **do**
    Choose $\phi_t = \underset{\phi \in \Phi}{\operatorname{argmin}} \int_B P_t^e(\phi, \phi_t, \mu_{t-1}^j(\phi_t)) d\phi^1 ... d\phi^p$
    Observe validities $[v_t^j = f^j(\phi_t) + \epsilon_t]_{j \leq i}$
    Perform updates to get $[\mu_t^j]_{j \leq i}$ and $[\sigma_t^j]_{j \leq i}$ using eqns. 9–11
  end for
end for

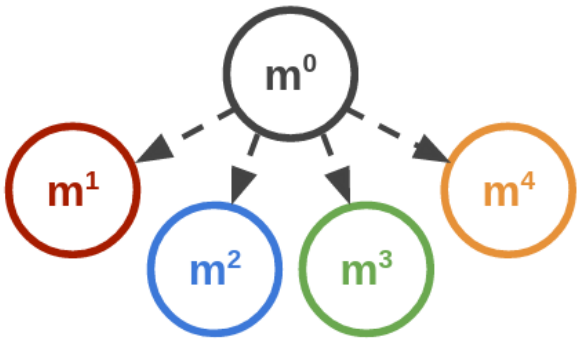

Fig. 3. An example model graph with a trivial order that makes no assumptions about the relative performance of its models

multiple models, we can re-use those measurements. Consider a pair of samples, $v^i(\phi^*)$ and $v^j(\phi^*)$. The former requires a measurement of $m^i(\phi^*)$ and the latter requires a measurement of $m^j(\phi^*)$, but both require a measurement of $m^0(\phi^*)$. In practice, it is unlikely that the GP-MPM algorithm would choose the same sample point for two different models, but each algorithm-directed measurement of $v^i(\phi)$ can be cheaply repurposed to make undirected measurements of $v^j(\phi)$, which still reduce the number of samples required to reach a desired classification rate.

Second, we can use the partial order to choose which valid sets to characterize first. Because the valid sets of lesser models are subsets of those of greater models, once we know the valid set of a greater model V^i, we can exclude everything outside of it from the search space for the valid set of a lesser model V^j—because we know that $v^j(\phi)$ is negative outside of V^i.

Fig. 1 shows an example of an informative order, which can be used to direct sampling, and fig. 2 shows a possible map of valid sets corresponding to that tree.

It is not necessary, however, to assume a strong order on the models. We can choose to assume only that the models $[m^1, ..., m^n]$ are all less than m^0, as shown in fig. 3. This maximizes generality, but sacrifices the sampling efficiency gained by iteratively constricting the search space.

V. Experiments

To quantify the performance of the GP-MPM algorithm and compare it against alternative methods, we run a series of tests against known and directly samplable validity functions. For each test, we apply three methods of approximating the valid set:

1) **GP-MPM** We draw five points from a uniform random distribution over Φ to initialize the Gaussian process, and then the algorithm runs unsupervised, with performance data collected at perfect square time steps, to match the grid search.
2) **Grid Search** We initialize a Gaussian process with the same kernel function as GP-MPM, but instead of directed sampling, the dataset consists of perfect square numbers of sample points, arranged in evenly-spaced grids.
3) **SVM** We binarize the validity function and sample it using uniform random draws from Φ, using the resulting set of labels as training data for a support vector machine. The SVM uses a radial basis function with parameters chosen by an exponential grid search to maximize the classification rate.

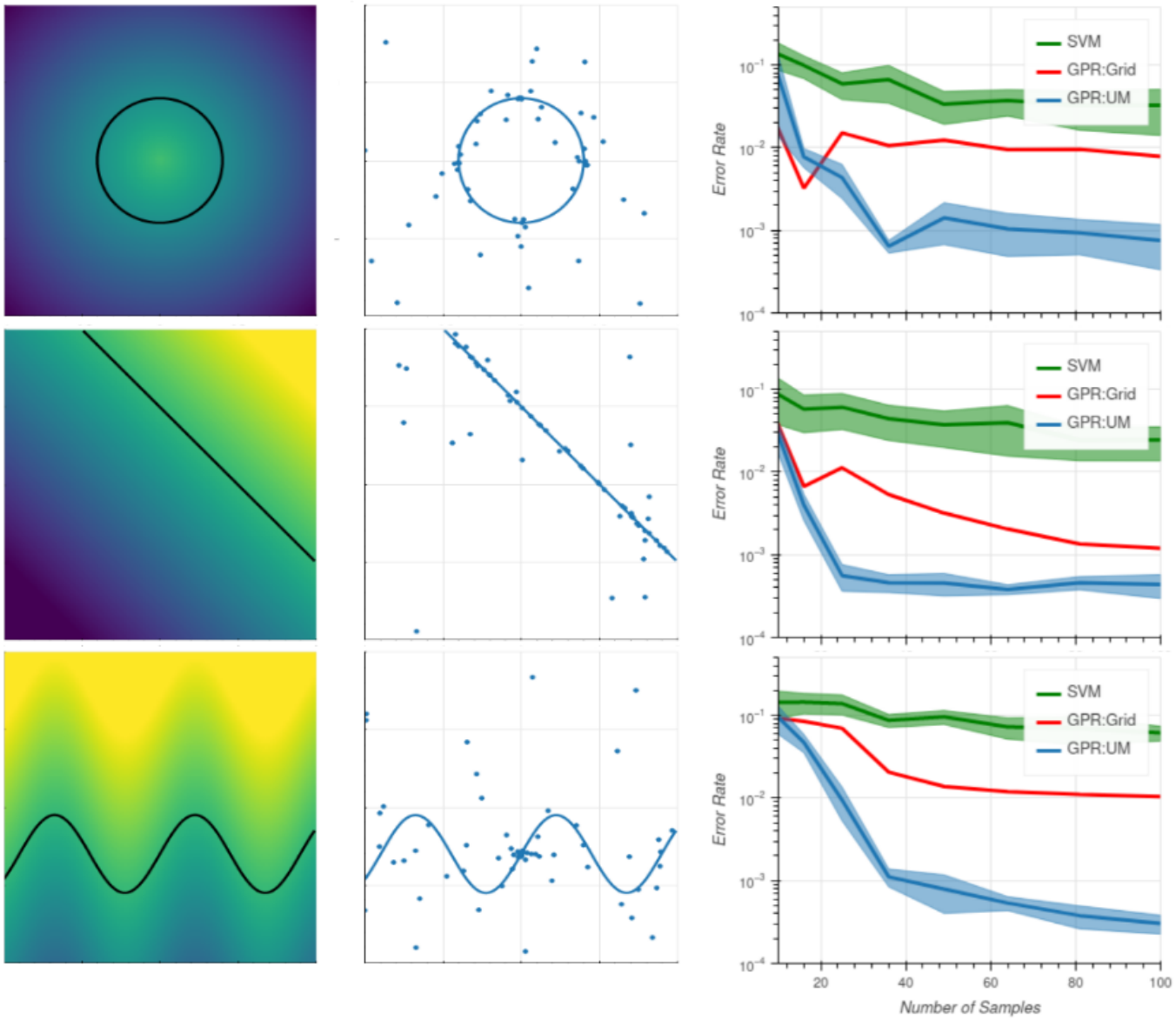

Fig. 4. Results of the GP-MPM algorithm for several known validity functions. Left: a heatmap of each validity function, with yellow corresponding to positive values, blue to negative, and the black line corresponding to $v(\phi) = 0$. Center: the algorithm's estimate of each valid set with n=64. Right: misclassification rate for several classification methods as a function of the number of samples collected.

The results in fig. 4 are striking, but perhaps unsurprising, as GP-MPM is able to exploit significantly more information about the validity function from each sample than either of the other methods. In every case, the GP-MPM algorithm reaches a 0.1% misclassification rate with less than 50 function evaluations—and in one case as little as 25 samples. At 100 samples, it outperforms the undirected grid search by anywhere from a factor of 2 to a factor of 20, depending on the complexity of the valid set being approximated.

We also ran a test on a dynamical simulation with an unknown validity function. In this test, an autonomous agent was given the task of navigating from $(0, 0)$ to $(4, 0)$ across a space obstructed by one obstacle, using a control system synthesized from a control barrier function (CBF). The condition vector ϕ consists of the x and y positions of the obstacle, which can vary freely from one test to another, In the true system case, the dynamics and the CBF are

both calculated from a double integrator, while in the simplified model, the dynamics are calculated from a double integrator, while the CBF is calculated from a single integrator–causing a mismatch. The validity function, then, is the maximum amount by which the the single integrator overestimates the output of the CBF over the course of a test, multiplied by -1:

$$v(\phi) = \max_{t \in \{1 \ldots T\}} (h_d(x,t) - h_s(x,t)) \qquad (38)$$

where x is the system state, $h_d(x,t)$ is the CBF calculated from a double integrator, and $h_s(x,t)$ is the CBF calculated from a single integrator.

Because the system is considered to be safe when the output of the CBF is positive, a negative $v(\phi)$ represents a trial where the simplified model considered itself to be safe, but the true dynamics were not. In many cases, such as the trace shown in fig. 5, this results in a collision with the obstacle. The true validity function was estimated using a Gaussian process regression over 5000 evenly-spaced sample points.

The results in fig. 6 again show that GP-MPM significantly outperforms the grid search, despite much of the valid set's border lying near the edge of Φ, which is known to be undersampled by GP-MPM. The misclassification function that GP-MPM is trying to minimize is not assumed to exist outside Φ, so it calculates less potential to reduce the function's value by sampling near the border.

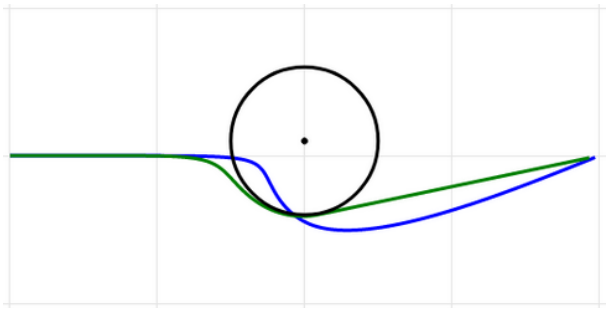

Fig. 5. A pair of example traces from the control barrier function test. Green is the true system, and blue is a model that approximates its control barrier function with simplified dynamics. In this trace, the simplified model collides with the obstacle.

VI. Further Work

Quadruped Simulations. There are several major regions of interest for continued work on this

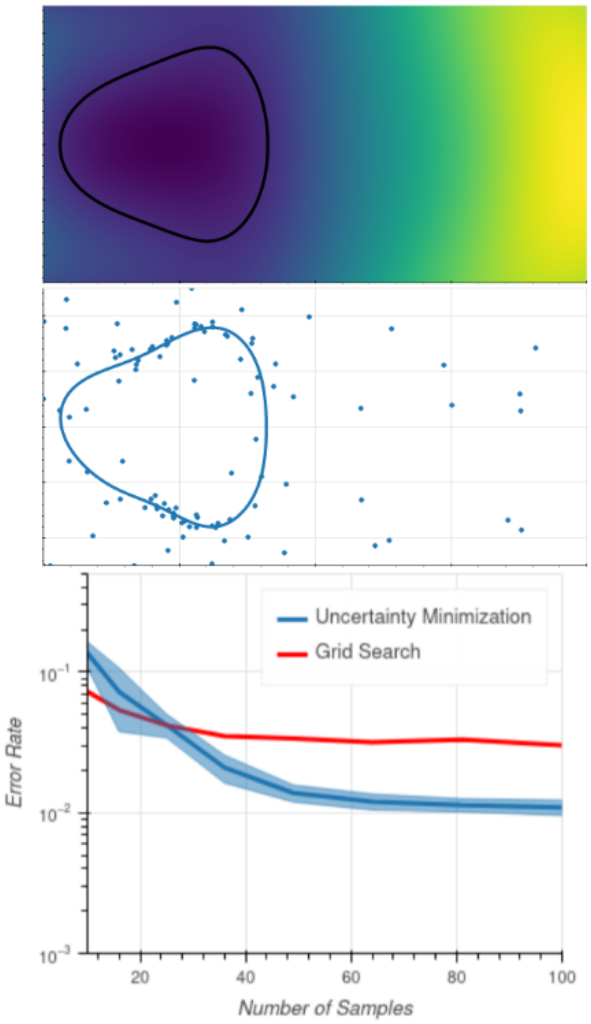

Fig. 6. Results of the GP-MPM algorithm for an unknown validity function based on control barrier function dynamics

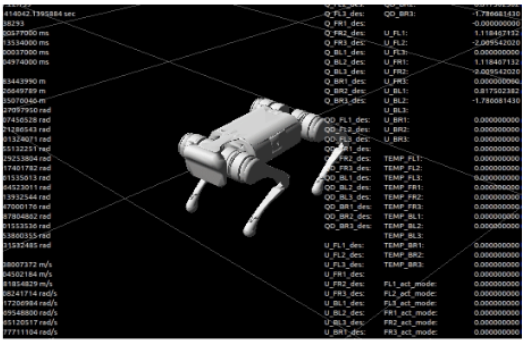

Fig. 7. A detailed kinematic simulation of a quadruped robot, to be used as a model for testing the GP-MPM algorithm

project, but the most immediate is to extend the applications of the algorithm to more interesting test cases. This includes real-world engineering problems, such as estimating regions of model mismatch for a set of models of a quadruped robot (fig. 7):

1) High-fidelity model: a detailed articulated simulation of the quadruped's dynamics, treated as the true system
2) No-slip model: a simplified version of the high-fidelity model that neglects friction, expected to display low validity when ground conditions are slippery
3) Bicycle model: a highly simplified dynamical model, expected to display low validity when the quadruped makes sharp turning maneuvers.

Unlike toy models, or even simple double integrators, models of quadruped locomotion are an active area of research. Demonstrating the capability of the GP-MPM algorithm on a set of models with direct interest to the robotics community will help to make the use cases of the algorithm clear. We have run a number of test simulations on the high-fidelity simulation and are working to choose a set of tests to best demonstrate the performance of the algorithm.

On the other hand, there are also several abstract examples that would provide more insight into the properties of the algorithm than the toy models demonstrated above. In particular, we can take advantage of the fact that the Gaussian process, while often treated as a set of jointly Gaussian random variables, is defined as a probability distribution over functions. By sampling many functions from this distribution, we can quantify the algorithm's performance over the set of all allowed functions $f(\phi)$.

Proving Properties. A second area of interest is in proving desireable properties of the algorithm. The Gaussian assumptions shared across measurement uncertainty, observation likelihood, and covariance (assuming a squared-exponential kernel) provide a lot of structure to the method, so it should be feasible to prove that it behaves optimally in more than one respect.

Since the algorithm is a Bayes estimator, it minimizes the expected posterior of its loss function by definition, but it is also common for this class of sampling algorithm to prove bounded regret. Structurally, GP-MPM shares a lot in common with GP-UCB, and that algorithm has proven regret bounds over a wide variety of kernel functions [22]. Adapting one of those proofs appears to be a promising toute proving bounded regret for GP-MPM.

Dynamic Condition Vectors. Some of the most interesting potential applications of GP-MPM are systems where the condition vector includes at least a subset of the state vector of a dynamical system. As mentioned above, control barrier functions are defined as safe when their output is positive, equivalent to GP-MPMs definition of the valid set. So it may be that GP-MPM is a very good method for generating control barrier functions from data.

Similarly, consider the case where ϕ is the state vector of an autonomous vehicle, and some unknown but measureable subset of ϕ provides positive reward. The original GP-MPM algorithm assumes that it can sample at any point in Φ each time step, without restriction, so the continuity constrants on the system state would prevent it from estimating the valid set. It is possible to modify the Bayes estimator to find paths of finite length tht minimize the expected posterior misclassification rate, but it is unclear if it is possible to do so in such a way that the estimator is still computeable.

In both of these cases, however, there is a more fundamental theoretical question that we must answer: under what conditions does including system state elements in the condition vector lead to circular logic and unsolveability? Consider the case where $\phi = d$ is the scalar distance to an obstacle, and $[V^i(\phi)]_{i=1}^n$ is a set of valid sets corresponding to the regions over which various perception models provide an accurate measurement of d. We cannot use the valid sets to decide which perception model to use at each value of d, because our knowledge of d depends on which perception model is currently in use. Is this true for all perception variables? Or is it simply a result of uncertainty in ϕ, which we assume to be known exactly in GP-MPM? It is important to determine the extent of this circularity if we want to apply the algorithm to state variables.

VII. Conclusion

We have presented GP-MPM, a novel Bayes estimator which chooses actions that iteratively minimize the expected posterior misclassification rate of a valid set by exploiting the continuous structure of the validity function underlying that valid set. In tests against known valid sets, GP-MPM significantly outperforms the misclassification rate of undirected grid-based sampling, sometimes by more than an order of magnitude. The algorithm has applications in any environment where running high-fidelity simulations is expensive, and the fidelity of low-cost alternatives is uncertain.

References

[1] M Ahmed, R Seraj, SMS Islam, "The k-means algorithm: A comprehensive survey and performance evaluation," Electronics, 9(8), 1295, 2003

[2] P Akella, A Dixit, M Ahmadi, JW Burdick, AD Ames, "Sample-based bounds for coherent risk measures: Applications to policy synthesis and verification," arXiv:2204.09833, 2022

[3] M Blum, MA Riedmiller, "Optimization of Gaussian process hyperparameters using Rprop," ESANN pp. 339-344, 2013

[4] MR Bonyadi, Z Michalewicz, "Particle swarm optimization for single objective continuous space problems: a review," Evolutionary Computation. 25 (1): 1–54, 2017

[5] BE Boser, IM Guyon, VN Vapnik, "A training algorithm for optimal margin classifiers," In Proceedings of the fifth annual workshop on Computational learning theory (pp. 144-152), 1992

[6] S Chen, J Montgomery, A Bolufé-Röhler. 2015, "Measuring the curse of dimensionality and its effects on particle swarm optimization and differential evolution," Appl. Intell. 42, 3 514–526, 2015

[7] Z Chen, B Wang, "How priors of initial hyperparameters affect Gaussian process regression models," arXiv:1605.07906, 2016

[8] SR Chowdhury, A Gopalan, "On kernelized milti-armed bandits," Proceedings of the 34th International Conference on Machine Learning, PMLR 70:844-853, 2017

[9] AP Dempster, NM Laird, DB Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," Journal of the Royal Statistical Society, Series B. 39 (1): 1–38, 1977

[10] B Efron, RJ Tibshirani, *An introduction to the bootstrap* CRC press, 1994

[11] A Gelman, JB Carlin, HS Stern, DB Dunson, A Vehtari, DB Rubin,*Bayesian data analysis*, CRC press, 2013

[12] MN Gibbs and DJC Mackay, "Variational Gaussian process classifiers," in IEEE Transactions on Neural Networks, vol. 11, no. 6, pp. 1458-1464, 2000

[13] C He, S Huang, R Cheng, K Chen Tan, Y Jin, "Evolutionary multiobjective optimization driven by generative adversarial networks (GANs)," IEEE Trans. Cybern. (2020).

[14] P Jorion, "Value at risk: the new benchmark for managing financial risk (3rd ed.)," McGraw-Hill, 2006

[15] J Kirschner, T Lattimore, C Vernade, C Szepesvari "Asymptotically Optimal Information-Directed Sampling," Proceedings of Thirty Fourth Conference on Learning Theory, 134:2777-2821, 2021

[16] K Kuester, S Mittnik, M Paolella, "Value-at-Risk Prediction: A Comparison of Alternative Strategies." Journal of Financial Econometrics. 4: 53–89, 2006

[17] T Lu, S-H Shiou, "Inverses of 2x2 block matrices," Comput Math Appl 43:119-129, 2002

[18] DT McRuer, RH Klein, "Automobile controllability – driver/vehicle response for steering control volume I," Department of Transportation DOT-HS-359-3-762, 1975

[19] F Nielsen, "K-MLE: A fast algorithm for learning statistical mixture models" IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 869–872, 2012

[20] BF Pearce, WA Johnson, RK Siskind, "Analytical study of approximate longitudinal transfer functions for a flexible airframe," Air Force Systems Command Project 8219, Task 821901, 1962

[21] T Ren, F Cui, S Sanghavi, N Ho, "Beyond EM Algorithm on Over-specified Two-Component Location-Scale Gaussian Mixtures," arXiv preprint arXiv:2205.11078, 2022

[22] N Srinivas, A Krause, SM Kakade, M Seeger, "Gaussian process optimization in the bandit setting: No regret and experimental design," arXiv preprint arXiv:0912.3995, 2009

[23] Y Tang, "Beyond EM: A faster Bayesian linear regression algorithm without matrix inversions," Neurocomputing 378:435-440, 2019