Bayesian Active Structure Learning for Gaussian Process Probabilistic Programs

by

Gloria Z. Lin

S.B. Computer Science and Engineering
Massachusetts Institute of Technology, 2022

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2022

Author . . . . . .
Department of Electrical Engineering and Computer Science
January 14, 2022

Certified by . . . . . .
Vikash Mansinghka
Principal Research Scientist
Thesis Supervisor

Co-Supervised by . . . . . .
Tan Zhi-Xuan
PhD Student
Thesis Supervisor

Accepted by . . . . . .
Katrina LaCurts
Chair, Master of Engineering Thesis Committee

Bayesian Active Structure Learning for Gaussian Process Probabilistic Programs

by

Gloria Z. Lin

Submitted to the Department of Electrical Engineering and Computer Science
on January 14, 2022, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

What data should we gather to learn about the underlying structure of the world as quickly as possible, especially in cases where data is sparse or expensive to acquire? Structure learning techniques for Gaussian process (GP) probabilistic programs provide a rich framework for inferring qualitative structure in data. In this thesis, we improve the data-efficiency of probabilistic GP structure learning by extending it to the active learning setting. We present a sequential Monte Carlo algorithm for Bayesian active learning for GPs with a novel objective function, Kernel Information Gain (IG-K), to reduce uncertainty over model structure and parameters. As a baseline for comparison, we also formulate a second objective function, Predictive Information Gain (IG-P), that reduces uncertainty over the posterior predictive distribution.

We empirically validate that active learning with our novel IG-K objective is able to more accurately infer the structure of synthetic datasets using fewer datapoints than active learning with IG-P. We also validate the underlying active learning inference algorithm using simulation-based calibration. Finally, we test our active learning algorithm on a real-world dataset with complex structure. Collectively, the results provide a deeper understanding of the benefits and limitations of active structure learning using Gaussian processes, revealing that an active selection strategy suited for inferring the model structure and parameters may not favorable for providing accurate predictions. These findings suggest directions for future active learning approaches which combine the IG-K and IG-P objectives, leveraging the advantages of each objective to efficiently discover structure in data and provide accurate predictions.

Thesis Supervisor: Vikash Mansinghka
Title: Principal Research Scientist

Thesis Supervisor: Tan Zhi-Xuan
Title: PhD Student

Acknowledgments

This work would not have been possible without the support and mentorship of Tan Zhi-Xuan, who has been a constant source of advice, encouragement, and warmth throughout this program. Discussions with Xuan have not only furthered my understanding of Bayesian inference and probabilistic computing, but also helped me to gain a better appreciation for what it takes to do research. I thank my thesis supervisor, Vikash Mansinghka, for his deeply insightful guidance that helped drive my research. I am grateful to Feras Saad, Ulrich Schaechtle, Ben Zinberg, and Marco Cusumano-Towner, whose work on Bayesian GP structure learning directly enabled the existence of this project. I also thank Rachel Paiste and Amanda Brower for their support and positivity over the last year.

Finally, I am profoundly grateful to my parents Yi Lin and Angela Yeung, who have raised me with the greatest care and invested so much to help me succeed. I am also grateful to my wonderful friends, whose unfailing love and encouragement make the tough times easier and the good times better.

This research was supported under contract number CW30HR0011-20-C-0042, Compositionally Organized Learning To Reason About Novel Experience (COLTRANE), for the DARPA SAIL-ON program, and Intel's Probabilistic Programming with Fast, Verified, Programmable Inference award.

Contents

List of Figures

Chapter 1

Introduction

Exponential gains in available data and computational capabilities are changing the ways we understand the world and approach challenging predictive problems, such as traffic patterns [10], pandemic spread, and financial markets. The ability to statistically model data-rich processes provides a framework to perform optimization, forecasting, and other critical predictive tasks. In some cases, however, it is expensive or difficult to obtain data from the process we wish to model. Examples include hyperparameter tuning for machine learning when the objective functions are computationally expensive to evaluate [18], environmental pollution monitoring via costly in-situ sampling [10], and composite fuselage design where obtaining experimental samples is time-consuming and expensive [21]. In cases like these, we might choose to select observations carefully so as to learn more information about the underlying process. Assuming observations are selected sequentially, we can frame the problem as *active learning* [10] or *optimal experimental design* [20], where the goal is to formulate a sequential policy for selecting the most informative locations to observe, given information from previous observations and our beliefs about the underlying process. Formulating a sequential policy can also be considered as formulating an *objective* function; the sequential policy selects points which maximize the objective. By using active learning, we can maximize the amount of information we gain about the underlying process given a limited budget of observations.

A common model choice for active learning problems is the Gaussian process (GP) [9, 13], since GPs do not make strong (parametric) assumptions about the statistical processes they model. Additionally, it is easy to sample from and evaluate posterior likelihoods of GPs. Existing GP active learning algorithms [10, 17] maximize objective functions of predictive entropy and reduction in predictive variance to actively select observations. However, these algorithms assume that the GP covariance kernel is known either at the structure level [10] (e.g. whether it is squared exponential or periodic) or at the param-

eter level [17] (e.g. a fixed period of a periodic kernel), which are often overly strong assumptions for real-world applications. In such cases, structure and parameter mismatch of the kernel with the data has been shown to significantly decrease the predictive ability of GP regression [6, 17]. Additionally, the structure of the covariance kernel determines almost all the generalization properties of a GP model, so fixing the kernel structure does not guarantee it is the most appropriate choice to model features of interest in the data [6].

To overcome this limitation, Duvenaud et al. [6] propose structure learning for GPs by performing search over a rich space of GP kernels which are hierarchically composed from simpler kernels [6, 11]. Since searching over the infinite space of compositional kernels is intractable, Duvenaud et al. propose a greedy algorithm for kernel search. Work by Schaechtle et al. [16] and Saad et al. [14] improve upon the greedy search technique by reformulating kernel search as probabilistic program synthesis: representing GP models as probabilistic programs and using Bayesian inference to synthesize these GP programs, conditioned on observed data. Because this approach quantifies uncertainty over possible kernel structures, this suggests that it can be combined with active learning to maximize information gain about model structure, while also overcoming the predictive limitations of existing GP active learning algorithms. It is the combination of active learning with this rich GP model class that we investigate in this thesis.

1.1 Key Contributions

This thesis makes the following contributions:

1. **Novel objective for optimizing information gain over model structure and parameters.** We formulate a novel GP active learning objective, Kernel Information Gain (IG-K), to maximize information gain on GP kernel structure and parameters, defining information gain as reduction in entropy over a GP model class introduced by Duvenaud et al. [6]. As a baseline for comparison, we also formulate a second objective, Predictive Information Gain (IG-P), to maximize information gain on predictive posteriors. IG-P is based on ideas from the Active Learning Cohn (ALC) objective implemented by Seo et al. in [17].

2. **Monte Carlo approximation for information gain objectives.** To tractably compute the IG-K and IG-P objectives, we formulate Monte Carlo estimators of IG-K and IG-P. To justify discrete approximation of the model entropy for the IG-K estimator, we show that model entropy decomposes into the sum of entropies over the model kernel structure and parameters separately.

3. **Bayesian active learning algorithm for GPs using SMC.** We define a Bayesian active learning algorithm for GPs using the estimators of the IG-K and IG-P objectives. The active learning algorithm extends work by Schaechtle et al. [16] and Saad et al. [14] on Bayesian synthesis of GP probabilistic programs to the active learning setting using a sequential Monte Carlo (SMC) procedure [1].

4. **Empirical Validation.** We present empirical results for inferring the correct GP model and predicting new data using active learning with the IG-K objective versus the IG-P objective. We also present empirical validation of the underlying SMC algorithm using simulation-based calibration (SBC) [5] and results of active learning with IG-K and IG-P on a real-world dataset. We present qualitative analysis of these results which provides insight into the benefits and limitations of using GPs for active structure learning.

Chapter 2 provides background on Bayesian synthesis of GP probabilistic programs [14] and sequential Monte Carlo (SMC) [5]. Chapter 3 presents the SMC algorithm for Bayesian active structure learning for GPs with the IG-K and IG-P objectives. It also presents formulation of the objectives and their Monte Carlo approximations. Chapter 4 presents results of experiments testing active learning with IG-K and IG-P objectives on synthetic datasets using a grid approximated posterior over constrained GP model classes. Chapter 5 presents results of simulation-based calibration (SBC) validation of the underlying SMC algorithm, as well as results on a real-world dataset of airline travel data. We conclude with a discussion of our overall findings and potential future work in Chapter 6.

1.2 Related Work

The section reviews previous work in structure discovery and active learning using GPs. 1.2.1 describes work in automating kernel selection for GP regression via greedy search over a space of compositional covariance kernels [6, 11]. 1.2.2 describes prior work that reformulates the GP kernel search problem and model class introduced in [6] as probabilistic program synthesis, allowing for more robust Bayesian search over kernel structures [14]. 1.2.3 and 1.2.4 describe entropy-based GP active learning algorithms [17, 20] which inform the information gain objectives developed in this thesis. 1.2.5 describes an approximate algorithm for nonmyopic GP active learning [10] which addresses parameter (but not structure) uncertainty in the GP model, suggesting it can be combined with the myopic active learning algorithm presented in this thesis to take advantage of both structure search and nonmyopic planning.

1.2.1 Structure Discovery via Compositional GP Kernel Search

Choosing an appropriate parametric form of the covariance kernel used in GP regression can require expertise and trial and error. To automate this choice, Duvenaud et al. present a greedy kernel search technique: Automatic Bayesian Covariance Discovery (ABCD) [6]. Duvenaud et al. formulate a space of kernel structures defined compositionally in terms of sums and products of four base kernel structures: squared exponential (SE), periodic (PER), linear (LIN), and rational quadratic (RQ). Examples of possible kernel structures in this space include PER, LIN $\times$ LIN $\times$ LIN, and SE + RQ. To search over the space of structures, ABCD greedily chooses the highest scoring kernel at each iteration, then expands it by applying all possible search operators: adding a base kernel, multiplying by a base kernel, and replacing an existing base kernel with another base kernel type. To score kernels, the algorithm uses marginal likelihood, which can be computed analytically when conditioned on the kernel structure with particular parameter values. However, evaluating a kernel structure means integrating over possible kernel parameters, which is intractable. ABCD approximates this integral with the Bayesian information criterion (BIC), which trades off model fit and complexity, on models whose kernel parameters are optimized by conjugate gradient descent. In experiments, ABCD is validated on real-world and synthetic datasets and slightly outperforms standard baselines at extrapolation and high-dimensional prediction.

The kernel structures learned by ABCD often yield decompositions of the data into diverse and interpretable components. Later work by Lloyd et al. [11] expands on this interpretability to present an automated statistician which performs ABCD and interprets the learned kernel to describe the data using natural-language text and figures. The system is evaluated on univariate timeseries datasets from real-world applications and successfully discovers complex temporal patterns in the data. To adapt to this use case, the system includes modifications to ABCD as presented in [6]. Changes include adding white noise (WN) and constant (C) base kernel types, introducing the changepoint operator, and removing the rational quadratic (RQ) kernel, since it is difficult to interpret.

1.2.2 GP Kernel Search as Probabilistic Program Synthesis

Probabilistic programming languages provide a rich paradigm for specifying models and performing inference. The work presented in this thesis builds upon a line of previous research in modeling GPs as probabilistic programs, including prior work by Zinberg [22], Schaechtle et al. [16], and Saad et al. [14] in developing a framework for performing Bayesian GP structure learning. This framework, presented in detail in chapter 2, refor-

mulates the GP kernel structure search problem and model class in [6] as Bayesian synthesis of GP probabilistic programs, allowing for more robust Bayesian search over kernel structures. The Bayesian active structure learning algorithm presented in this thesis is implemented in the Gen probabilistic programming language [4], leveraging Gen's ability to perform automated involutive Markov Chain Monte Carlo (MCMC) [3] to search over GP covariance kernels of the kind introduced in Duvenaud et al. [6].

1.2.3 Variance-Based Active Learning with GPs

Using sequential design methods suggested by MacKay [12] and Cohn [2], Seo et al. implement and evaluate two active selection strategies for GP regression based on the variance of the GP predictive posterior over target unobserved values [17]. The first strategy, Active Learning MacKay (ALM), chooses the next point to maximize predictive variance, which approximates a maximum entropy design. The second strategy, Active Learning Cohn (ALC), chooses the next point which causes the greatest reduction in predictive variance averaged over a user-specified range of target values. Both ALM and ALC rely heavily on the assumption that the correct model, i.e. GP covariance kernel and noise, is known, which is frequently untrue for real-world data applications. Consequently, ALM and ALC are able to reduce predictive mean squared error (MSE) on held-out data better for synthetic data than real-world data, due to model mismatch between the assumed kernel and real-world data. ALC outperforms ALM at reducing predictive variance and MSE, and both outperform a random selection strategy. The predictive information gain (IG-P) objective presented in this thesis (section 3.3) is based on the ALC policy.

1.2.4 Active Bayesian Causal Discovery using GP Networks

Kügelgen et al. present a Bayesian GP active learning approach for causal discovery through targeted interventions [20]. The goal of the algorithm is to learn the structural causal model over d real-valued observable variables through experiments, i.e. targeted interventions on one of the variables, fixing its value, and observing the resulting values of the other variables to find evidence of causality. The structural causal model is assumed to be a directed acyclic graph where the relationships between parent and child variables are modeled by non-linear functions with additive noise. The structure and parameterization of such causal graphs can be modeled as GPs, allowing the marginal likelihood and posteriors of observed data and causal graphs to be calculated in closed form. To decide which experiment to perform next, the authors propose a Bayesian experimental design approach, selecting an experiment ξ which maximizes a given utility function $U(y|\xi)$ that describes

the usefulness of observing the outcome y after performing experiment ξ.

To learn the causal model, the authors suggest defining utility as information gain over model uncertainty. This requires integration over the possible outcomes of particular interventions, which is intractable and approximated using a Monte Carlo estimator which draws samples from the predictive interventional distribution. The authors also suggest using Bayesian optimization as a way to find the globally optimal intervention with respect to maximizing utility. The kernel information gain policy (IG-K) presented in this thesis (section 3.2) is also based on the idea of maximizing information gain over model uncertainty, with distinct notions of structure and parameter uncertainty. However, IG-K approximates outcome integrals via gridding rather than sampling over the interventional distribution, and it searches for the optimal intervention through enumeration rather than Bayesian optimization.

1.2.5 Nonmyopic Bayes-Optimal Active Learning of GPs

To address the model mismatch problem of other GP active learning algorithms, Hoang et al. present ϵ-Bayes-optimal active learning (ϵ-BAL), a nonmyopic selection policy which jointly optimizes selecting locations informative for both spatial prediction and model estimation [10]. Though ϵ-BAL still assumes that the correct covariance kernel structure (a squared exponential kernel) is known, it does not assume that the lengthscale parameter is known, instead maintaining a Bayesian belief over possible lengthscales. Given a planning horizon of N observations, ϵ-BAL recursively selects a sequence of N observations which maximize joint entropy with respect to all possible induced sequences of future beliefs, hence the term "nonmyopic". Like ALM, this follows a maximum entropy design. Unlike ϵ-BAL, which uses a planning horizon of N > 1, all other active learning policies discussed so far use a horizon of N=1, including the policies presented in this thesis. However, as a direction for future exploration, the policies presented in this thesis may be combined with ideas from ϵ-BAL to take advantage of the former's search over kernel structures (ϵ-BAL assumes kernel structure is fixed) and the latter's nonmyopic planning horizon.

Due to an uncountable number of candidate observations and unknown model parameters, the stage-wise entropy and expectation quantities required for ϵ-BAL cannot be evaluated precisely and are approximated within a loss bound $\epsilon > 0$ using a truncated sampling procedure. Even so, the cost of deriving the ϵ-optimal policy is exponential with respect to planning horizon N. To ease this computational burden, the authors present $\langle\alpha, \epsilon\rangle$-BAL, an anytime algorithm based on ϵ-BAL. At each stage, $\langle\alpha, \epsilon\rangle$-BAL only explores the next states with uncertainty exceeding α to prune unnecessary exploration while still guaran-

teeing policy optimality. Using root mean squared prediction error as the performance metric, $\langle \alpha, \epsilon \rangle$-BAL outperforms state-of-the-art myopic, entropy-based GP active learning algorithms on simulated GP data and real-world traffic data, and the performance gain is especially high when the sample budget is small (5-10 observations). In the experiments, the planning horizon N does not exceed 5, and 7 candidate values of the lengthscale parameter are considered.

Chapter 2

Background

This chapter provides an overview of the Bayesian inference framework for GP probabilistic programs by Saad et al. [14] (built on previous work by Schaechtle et al. [16]) and provides background on SMC [1], which will allow us to extend this inference framework to the active learning setting.

Saad et al. [14] reformulate the kernel search problem introduced by Duvenaud et al. [6] as probabilistic program synthesis [14] to allow for fully Bayesian search over the space of covariance kernels, making structure discovery more robust and generalizable. The probabilistic program synthesis approach relies on three key ideas:

1. Representing a class of GP models (from Duvenaud et al. [6]) as probabilistic programs using a domain-specific language (DSL).

2. Defining prior and posterior distributions over the GP model class, conditioned on observed data.

3. Performing Bayesian search over the GP model class by (approximately) sampling from the model posterior.

GP models defined in the DSL are translated to executable code for obtaining GP predictions and likelihoods conditioned on observed data. GP programs approximately sampled from the model posterior provide a good fit for observed data. Analysis of these sampled GP programs allow us to quantify uncertainty over possible kernel structures.

Section 2.1 describes the DSL for representing GPs and defines prior and posterior likelihoods for GP models defined in the DSL. Section 2.2 describes Markov Chain Monte Carlo (MCMC) techniques which allow for sampling of GP models from the approximate model posterior given observed data, i.e. Bayesian synthesis of GP programs. Section 2.3 provides an overview of SMC algorithms.

2.1 GP Models as Probabilistic Programs

2.1.1 GP Model Class

Gaussian processes are distributions over functions $f : \mathcal{X} \to \mathcal{Y}$ such that any finite set of function evaluations have a jointly Gaussian distribution. Following the notation of [9], we formalize a GP $f \sim \text{GP}(m, k)$ with mean function $m : \mathcal{X} \to \mathcal{Y}$ and covariance function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ as follows: f is a collection of random variables $\{f(x) : x \in \mathcal{X}\}$. Given locations $\{x_1, ..., x_n\}$ the vector of random variables $[f(x_1), ..., f(x_n)]$ is jointly Gaussian with mean vector $[m(x_1), ..., m(x_n)]$ and covariance matrix $[k(x_i, x_j)]_{1 \leq i,j \leq n}$. The prior mean is typically set to zero, and the functional form of the covariance kernel k defines essential features of the unknown function f which allow the GP to fit structural patterns in observed data and make forecasts. To express the space of compositional GP covariance kernels presented in ABCD [6, 11], Saad et al. [14] define a DSL for expressing the kernel of a specific GP. The implementation of Bayesian program synthesis used in this thesis uses a slightly different DSL, which is presented below:

$$\begin{aligned} K \in \text{Kernel} \;\coloneqq\; & \texttt{(C } \upsilon\texttt{)} \mid \texttt{(LIN } \upsilon\texttt{)} \mid \texttt{(SE } \upsilon\texttt{)} \mid \texttt{(PER } \upsilon_1\ \upsilon_2\texttt{)} \\ & \mid \texttt{(+ } K_1\ K_2\texttt{)} \mid \texttt{(}\times\ K_1\ K_2\texttt{)} \end{aligned}$$

The base kernels are constant (`C`), linear (`LIN`), squared exponential (`SE`), and periodic (`PER`). Each base kernel has one or more numeric parameters υ, e.g. `PER` has a lengthscale and period. The composition operators are sum (+), product ($\times$). Every GP program in this DSL also has a numeric noise parameter η representing diagonal noise added to covariance matrices calculated using the GP kernel. We omit the changepoint `CP` operator and the white noise `WN` kernel presented in the original DSL [14]; `WN` is effectively replaced by η.

2.1.2 GP Model Prior

Given the DSL grammar, Saad et al. implement a probabilistic context-free grammar (PCFG) which assigns a production probability (structure prior) to each covariance structure in the grammar. This makes it possible to sample specific GP programs from the DSL according to this structure prior. For the implementation of the DSL used in this thesis, the production probabilities are 0.2 for each base kernel (`C, LIN, SE, PER`) and 0.1 for each operator (+, $\times$), unless otherwise specified in experiment descriptions. Under this prior, the probability of sampling a GP program with linear + periodic covariance structure (marginalized over all values of parameters υ) is:

$$
\begin{aligned}
P(\texttt{(+ (LIN) (PER))}) &= P(+) \cdot P(\texttt{LIN}) \cdot P(\texttt{PER}) \\
&= 0.1 \cdot 0.2 \cdot 0.2
\end{aligned}
$$

All numeric kernel parameter values υ are sampled independently from a uniform prior over the range [0,1], and the noise parameter η is sampled from a Gamma prior with $\alpha = 1, \beta = 1$. Using these priors, we can analytically calculate the prior probability of any specific GP program in the DSL.

2.1.3 GP Model Posterior

Given a the covariance function of a GP specified in the DSL, the predictive distribution over output values $f(x_1), ..., f(x_n)$ at locations $x_1, ..., x_n$ is a standard multivariate normal which can be calculated analytically using definitions from [9]. Using this key property of GPs, we can translate any specific GP program in the DSL to executable Gen probabilistic programs which calculate the likelihood of observed data (x, y) (2.1) and the predictive posterior of unknown $\tilde{y}$ at location $\tilde{x}$ under the GP θ (2.2), the latter of which allows us to perform forecasting and interpolation.

$$P(y \mid \theta; x) \tag{2.1}$$

$$P(\tilde{y} \mid \theta; \tilde{x}) \tag{2.2}$$

Using properties of conditional multivariate normal distributions, we can calculate (2.1) and (2.2) conditioned on previously observed data (in addition to the model θ), which is useful for the active learning setting where we select data sequentially. Multiplying the likelihood (2.1) by the prior probability of the GP program $P(\theta)$ as defined in section 2.1.2 gives the model posterior conditioned on observed data (2.3).

$$P(\theta \mid y; x) = P(y \mid \theta; x) \cdot P(\theta) \tag{2.3}$$

Thus, we can analytically calculate the posterior probability of any specific GP program in the DSL by translating the DSL program to Gen and performing the calculations above.

2.2 Bayesian Synthesis of GP Probabilistic Programs

The GP model class, prior, and conditional likelihood defined in section 2.1 make it possible to analytically calculate the posterior likelihood of particular GP models given observed data. Thus, it is possible to perform Bayesian inference over the model class to learn (*maximum a posteriori*) GP models which explain the structure of observed data well, i.e. Bayesian synthesis of GP programs. However, since the model class consists of an infinite number of compositional GP kernels (section 2.1.1), calculating the posterior over all possible models is intractable. Therefore, approximate inference is necessary, and Saad et al. propose an inference strategy which simulates a Markov chain whose target distribution is the posterior over the GP model class, conditioned on observed data [14]. For searching over GP structures (base kernels and operators), they propose a Metropolis-Hastings (MH) algorithm with a transition operator that stochastically replaces a random sub-expression in the compositional covariance kernel of a GP program and stochastically accepts or rejects the mutation depending on how much it increases or decreases the likelihood of the GP program. Similarly, for searching over parameters of GP structures (e.g. lengthscale values for a squared exponential kernel), they propose a MH algorithm where the transition operator perturbs kernel parameter values.

For the overall Bayesian inference strategy, Saad et al. alternate between performing a MH step over the program structure and a MH step over program parameters. Saad et al. show that GP programs approximately sampled from the posterior using this technique provide a good fit for observed data, formally proving that the inference procedure is sound. Analysis of sampled GP programs allow us to infer qualitative properties of the underlying model. For example, the presence of the periodic kernel in a majority of the sampled GP programs would suggest the presence of a periodic trend in the data. Being able to quantify uncertainty over possible kernel structures using this inference method suggests that it can be combined with active learning to maximize information gain about model structure.

2.3 Sequential Monte Carlo

Bayesian synthesis of GP probabilistic programs [14] performs inference on all available data at once. In the active learning setting explored in this thesis, data is available sequentially: we select data one observation at a time, update the posterior belief based on this new information, and select the next observation based on the updated posterior. MCMC methods such as the one presented in section 2.2 are not suitable for active learning, since active learning involves a sequence of posteriors $\pi_1, ..., \pi_t$ as more observations are collected. This would require generating a different chain run for each posterior π_i, which is computationally expensive, and does not take into account the previous posterior π_{i-1}. Thus, to extend the Bayesian synthesis framework to the sequential active learning setting, we turn to more efficient iterative strategies known as Sequential Monte Carlo (SMC) or particle filter methods to draw samples from π_i by transforming the previous posterior π_{i-1} via reweighting operations and importance sampling [1].

As defined by Chopin [1], a particle system is a sequence (θ_i, w_i) of weighted random variables in Θ, θ_i being a particle with weight w_i, which targets a distribution of interest π over Θ in the sense that

$$\lim_{k \to +\infty} \frac{\sum_{i=1}^{k} w_i h(\theta_j)}{\sum_{i=1}^{k} w_i} = E_\pi\{h(\theta)\}$$

almost surely, for any measurable h such that $E_\pi\{h(\theta)\}$ exists. In order to draw particles from an unknown distribution π, we use importance sampling: a technique where we draw particles independently from a known distribution g, and weights proportional to $\pi(\theta_i)/g(\theta_i)$ provide a particle system with target distribution π.

An SMC (particle filter) algorithm provides consistent inferences from a sequence of distributions via the following procedure, where π_0 is the prior distribution, and π_i is the i-th posterior distribution in the sequence, conditioned on the sequence of observations $d_{1:i}$.

Algorithm 1 Sequential Monte Carlo

```
1:  procedure SMC
2:      parameters: k, number of particles
3:      θ_i ~ g for i ∈ [1, k]                          ▷ Sample k particles
4:      w_i ⟵ π_0(θ_i)/g(θ_i) for i ∈ [1, k]            ▷ Compute importance weights
5:      for τ ∈ [1, n] do
6:          w_i ⟵ w_i · π_τ(θ_i)/π_{τ-1}(θ_i) for i ∈ [1, k]   ▷ Reweight: update particle weights
7:          (θ_i, w_i) ⟵ (θ_i^r, 1) for i ∈ [1, k]             ▷ Resample: prune low-weight particles
8:          (θ_i, w_i) ~ REJUVENATE(θ_i^r, w_i) for i ∈ [1, k]   ▷ Rejuvenate: perturb particles according to π_t
9:      end for
10:     return [(θ_1, w_1), ..., (θ_k, w_k)]           ▷ Return particle system for inference
11: end procedure
```

Lines 3-4 initialize the particle system via importance sampling. Line 6 is the reweight-

ing step, which relies on the key intuition that a simple reweighting operation $w_i' = w_i \cdot \pi_2(\theta_i)/\pi_1(\theta_i)$ shifts the target of a particle system from π_1 to π_2. This allows us to perform reweighting iteratively for each $\pi_1, ..., \pi_t$ when the distribution of interest is evolving through time, like in active learning. A common problem with reweighting is weight collapse: as π_t moves away from π_1, fewer particles retain significant weights and some particles have high weights. To address this, we can resample particles (line 7), a technique where each particle θ_i is replaced by a number n_i (possibly zero) of its replicates according to some resampling scheme. Each new particle is assigned unit weight. In resampling schemes, n_i and w_i are positively correlated, effectively pruning low-weight particles. Resampling also replaces single high-weight particles with numerous lower weight copies that can evolve in different directions (as opposed to one) when resampling is combined with a rejuvenation step (line 8). Rejuvenation increases particle diversity by perturbing resampled particles according to a Markov chain transition operation, effectively replacing identical replicates of a single particle with fresh values.

In the SMC procedure presented above, data is passively, rather than actively, selected. At each timestep of the procedure, the target distribution π is moved by incorporating the next observation from the known sequence of observations $d_1, ..., d_n$. To extend the SMC procedure to the active learning setting, at each timestep, before reweighting, we select the next observation according to some policy. The next chapter presents an SMC procedure for active learning of GP probabilistic programs and formulates two data selection policies based on different information gain criteria.

Chapter 3

Bayesian Active Learning of GP Probabilistic Programs

This chapter presents the algorithm for Bayesian active learning of GP probabilistic programs, extending prior work by Schaechtle et al. [16] and Saad et al. [14]. This algorithm uses SMC and active data selection policies that maximize one of two objective functions: **(1) Kernel Information Gain (IG-K)** or **(2) Predictive Information Gain (IG-P)**. These objectives measure information gain on (1) the unknown GP model covariance kernel and (2) the GP predictive posteriors, respectively, where information gain is the decrease in entropy conditioned on the observing the data at a chosen location. Section 3.1 presents the SMC algorithm for Bayesian active learning of GP programs, and sections 3.2 and 3.3 present formulations and pseudocode of the IG-P and IG-K objectives.

3.1 Active Learning as SMC

The particle system for SMC active learning of GP programs consists of particles θ_i with corresponding importance weights w_i, where a particle θ_i represents a GP covariance kernel from the DSL (section 2.1.1) with kernel structure s_i, parameters υ_i, and noise η_i.

$$\theta_i := (s_i, \upsilon_i, \eta_i)$$

T is the sampling budget of observations (x_i, y_i) where a new observation location x_t is selected from a set of unobserved locations $\mathcal{X}$ at each timestep t. Let π_0 represent the covariance prior (section 2.1.2), and π_i be the posterior conditioned on the first i observations $(x_1, y_1), ..., (x_i, y_i)$. Below is the SMC procedure for active learning.

Algorithm 2 Sequential Monte Carlo for Active Learning

1: **procedure** SMC-AL
2: **parameters**: k, number of particles; c, resampling threshold; T, sampling budget
3: $\theta_i \sim \pi_0$ for $i \in [1,k]$ ▷ Sample k GPs θ from the covariance prior
4: $w_i \longleftarrow \pi_0(\theta_i)$ for $i \in [1,k]$ ▷ Compute prior probabilities of sampled GPs
5: **for** $t \in [1,T]$ **do**
6: **if** EFFECTIVE-SAMPLE-SIZE$(w_1, ..., w_k)/k < c$ **then**
7: $(\theta_i, w_i) \sim$ RESAMPLE$([(\theta_i, w_i)]_{1:k})$ for $i \in [1,k]$ ▷ Resample if ESS is low
8: **end if**
9: $(\theta_i, w_i) \sim$ REJUVENATE(θ_i) for $i \in [1,k]$ ▷ Rejuvenate s_i, υ_i, η_i
10: $x_t \longleftarrow \underset{\tilde{x} \in \mathcal{X}}{\text{argmax}}$ OBJECTIVE$(\tilde{x}, [(\theta_i, w_i)]_{1:k}, [(x_j, y_j)]_{1:t-1})$ ▷ Select next observation location x_t
11: $y_t \longleftarrow$ OBSERVE(x_t)
12: $w_i \longleftarrow w_i \cdot \pi_t(\theta_i)/\pi_{t-1}(\theta_i)$ ▷ Reweight using new posterior π_t
13: **end for**
14: **return** $[(\theta_i, w_i)]_{1:k}$ ▷ Return particle system for inference
15: **end procedure**

Lines 3-4 initialize the particle system by sampling GP programs from covariance prior presented in section 2.1.2 and computing their prior probabilities. For each timestep, we first calculate the effective sample size (ESS) (line 6), a measure of the inference quality of a particle system based on the variance of its particle weights. ESS is the sample size required to attain the same precision as with particles drawn directly from the target distribution [1]. To save unnecessary resampling, we resample (line 7) only when the ESS falls below a certain proportion c of the number of particles k. The resampling scheme used is residual resampling, which first allocates each particle θ_i a number of replicates equal to the integer floor of the expected number of θ_i particles drawn if we sampled k times from the particle system according to weights. If this procedure results in fewer than k replicates, the remaining particles are sampled at random from the particle system according to weights.

At every timestep, we rejuvenate the particles (line 9) using the following steps with the hyperparameters $\alpha, \beta, \delta, \gamma$ for each particle.

1. Rejuvenate kernel structure s using the MH move presented in section 2.2 which may replace a random sub-expression of the kernel structure.

2. Rejuvenate kernel parameters υ using a MH move that randomly selects a numeric parameter p with value v in the kernel, and may replace p with a new value sampled from the truncated normal with mean v, standard deviation δ, and range $[0, \gamma]$ (support of p). Repeat this procedure α times.

3. Rejuvenate noise parameter η using a MH move which may replace η with a new value from the noise prior.

4. Repeat steps 1-3 for β iterations.

Following rejuvenation, we select the next observation location $x_t \in \mathcal{X}$ which maximizes an objective function, either IG-K or IG-P (lines 10-11). Conditioning on the new observation (x_t, y_t) changes the target distribution (posterior), so we reweight the particle system according to the new posterior π_t (line 12). The rest of this chapter presents derivations and pseudocode of the IG-K and IG-P objectives in sections 3.2 and 3.3 respectively. For readability in 3.2 and 3.3, we will omit the noise parameter η, treating it in calculations as another numeric parameter included in υ.

3.2 Kernel Information Gain (IG-K) Objective

Active learning with the kernel information gain (IG-K) objective aims to reduce uncertainty in the inferred GP model by selecting observations to maximize information gain of the model.

Let Θ be a random variable representing the unknown model, a GP covariance kernel drawn from the model distribution over all kernels specifiable by the DSL (section 2.1). Thus, the entropy $H(\Theta)$ quantifies model uncertainty. Let Y_x be a random variable representing the unknown observed value at x, so the conditional entropy $H(\Theta|Y_x)$ quantifies model uncertainty after observing the value at x.

$$\begin{aligned} H(\Theta) &= -\int P(\theta) \log P(\theta)\, d\theta \\ H(\Theta|Y_x) &= \int f(Y_x = y) \cdot H(\Theta|Y_x = y)\, dy \end{aligned}$$

To reduce model uncertainty, IG-K selects an observation location $x* \in \mathcal{X}$ to maximize the information gain of Θ, given an observation at $x*$.

$$\begin{aligned} I(\Theta; Y_x) &= H(\Theta) - H(\Theta|Y_x) \\ x* &= \operatorname*{argmax}_{x \in \mathcal{X}} I(\Theta; Y_x) \end{aligned}$$

At each timestep of the particle filter algorithm, we calculate information gain conditioned on a sequence of observations. To condition on observations Y_{x_1}, Y_{x_2} sequentially, we can the apply chain rule of mutual information as follows.

$$\begin{aligned} I(\Theta; Y_{x_1}) &= H(\Theta) - H(\Theta|Y_{x_1}) \\ I(\Theta; Y_{x_2}|Y_{x_1}) &= H(\Theta|Y_{x_1}) - H(\Theta|Y_{x_2}, Y_{x_1}) \end{aligned}$$

The rest of this section describes how to estimate the kernel entropy $H(\Theta)$ and kernel conditional entropy $H(\Theta|Y_x)$ needed to calculate the IG-K information gain objective $I(\Theta; Y_x)$. Section 3.2.1 shows that $H(\Theta)$ and $H(\Theta|Y_x)$ decomposes to the sum of discrete and differential entropies over kernel structure and parameterization, which justifies using a discrete sample-based approximation to estimate kernel entropy. 3.2.2 gives formulations of estimators for $H(\Theta)$ and $H(\Theta|Y_x)$ using particle system approximation. Section 3.2.3 provides the pseudocode for estimating the IG-K objective.

3.2.1 Kernel Entropy Decomposition

A particular GP kernel θ can be characterized by its structure s and parameterization υ. As an example, a kernel θ defined in the DSL as

```
(+ (PER 0.5 0.1) (LIN 0.3))
```

has periodic + linear structure s and a parameterization υ of $\{0.5, 0.1, 0.3\}$. And we can calculate its probability according to the covariance prior (section 2.1.2) as

$$\begin{aligned} P(\theta) &= P(s) \cdot f(\upsilon|s) \\ &= P(+) \cdot P(\texttt{PER}) \cdot P(\texttt{LIN}) \cdot f(0.5) \cdot f(0.1) \cdot f(0.3) \end{aligned}$$

Let $S \in \mathcal{S}$ be a discrete random variable representing the unknown kernel structure, a structure s sampled from the posterior over the discrete space $\mathcal{S}$ of all kernel structures in the DSL. Let U be a continuous random variable representing the unknown parameterization υ. In the following propositions, we will prove that the entropy of the unknown model $H(\Theta)$ can be decomposed into the discrete entropy of S and differential entropy of U.

Proposition 3.2.1 (Kernel Entropy)**.** The kernel entropy $H(\Theta)$ decomposes into the discrete entropy of the structure S and conditional differential entropy of the parameterization U:

$$H(\Theta) = H(S) + h(U|S)$$

Proof.

$$\begin{aligned} H(\Theta) &= -\int P(\theta) \log P(\theta)\, d\theta \\ &= -\sum_{s \in \mathcal{S}} \int P(s) f(\upsilon|s) \log(P(s) f(\upsilon|s))\, d\upsilon \end{aligned}$$

$$
\begin{aligned}
&= -\sum_{s\in\mathcal{S}} P(s) \int f(\upsilon|s)\cdot(\log P(s) + \log f(\upsilon|s))\; d\upsilon \\
&= -\sum_{s\in\mathcal{S}} P(s)\cdot\left(\log P(s)\int f(\upsilon|s)\,d\upsilon + \int f(\upsilon|s)\log f(\upsilon|s)\,d\upsilon\right) \\
&= -\sum_{s\in\mathcal{S}} P(s)\cdot\left(\log P(s) + \int f(\upsilon|s)\log f(\upsilon|s)\,d\upsilon\right) \\
&= -\sum_{s\in\mathcal{S}} P(s)\log P(s) - \sum_{s\in S} P(s)\int f(\upsilon|s)\log f(\upsilon|s)\,d\upsilon \\
&= H(S) + h(U|S)
\end{aligned}
$$

□

Proposition 3.2.2 (Kernel Conditional Entropy)**.** The kernel conditional entropy $H(\Theta|Y_x)$ decomposes into the conditional discrete entropy of the structure S and conditional differential entropy of the parameterization U:

$$H(\Theta|Y_x) = H(S\,|Y_x) + h(U|S, Y_x)$$

Proof.

$$
\begin{aligned}
H(\Theta|Y_x) &= \int f(Y_x = y)\cdot H(\Theta|Y_x = y)\,dy \\
&= -\int f(y)\int P(\theta\,|\,y)\log P(\theta|y)\,d\theta \\
&= -\int f(y)\sum_{s\in\mathcal{S}}\int P(\theta\,|\,y)\;\log\;P(\theta\,|\,y)\,d\upsilon\,dy \\
&= -\int f(y)\sum_{s\in\mathcal{S}}\int f(s,\upsilon\,|\,y)\;\log\;f(s,\upsilon\,|\,y)\,d\upsilon\,dy \\
&= -\int f(y)\sum_{s\in\mathcal{S}} P(s|y)\int f(\upsilon|s,y)\log\left(P(s|y)\cdot f(\upsilon|s,y)\right)d\upsilon\,dy \\
&= -\int f(y)\sum_{s\in\mathcal{S}} P(s|y)\int f(\upsilon|s,y)\cdot(\log P(s|y) + \log f(\upsilon|s,y))\;d\upsilon\,dy \\
&= -\int f(y)\sum_{s\in\mathcal{S}} P(s|y)\cdot\left(\log P(s|y)\int f(\upsilon|s,y)\,d\upsilon + \int f(\upsilon|s,y)\log f(\upsilon|s,y)\,d\upsilon\right)dy \\
&= -\int f(y)\sum_{s\in\mathcal{S}} P(s|y)\log P(s|y)\,dy - \int f(y)\sum_{s\in\mathcal{S}} P(s|y)\int f(\upsilon|s,y)\log f(\upsilon|s,y)\,d\upsilon\,dy \\
&= H(S\,|Y_x) + h(U|S, Y_x)
\end{aligned}
$$

□

Because kernel entropy $H(\Theta)$ and conditional entropy $H(\Theta|Y_x)$ decompose into the terms in Propositions 3.2.1 and 3.2.2 respectively, this allows us to intuitively verify that a discrete approximation of the entropy serves as a valid estimate of the true entropy. Concretely, imagine drawing samples of GP kernels θ with structures s according to $P(s)$ and parameters υ according to $f(\upsilon|s)$, such that $P(\theta) = P(s) \cdot f(\upsilon|s)$; the average negative log probability of the sampled θ is a valid estimator of of the entropy $H(\Theta)$. Using this principle, we define sampling-based estimators of $H(\Theta)$ and $H(\Theta|Y_x)$ in the next section.

3.2.2 Kernel Entropy Estimator

$H(\Theta)$ and $H(\Theta|Y_x)$ cannot be calculated exactly since they require integrating over an infinite set of possible covariance kernels θ. $H(\Theta|Y_x)$ also requires integrating over unbounded Y_x. The following section describes how to estimate $H(\Theta)$ and $H(\Theta|Y_x)$ in two parts: (1) integrating over kernels using particle system approximation and (2) integrating over Y_x using Riemann sums.

Integrating over Kernels

The results of Propositions 3.2.1 and 3.2.2 in the last section provide confidence that we can estimate $H(\Theta)$ and $H(\Theta|Y_x)$ using discrete samples from the model distribution $P(\theta)$. In practice, the model distribution is unknown, but particle filter algorithms (section 2.3) allow us to approximately sample from $P(\theta)$ via importance sampling. In this section, we describe how to estimate $H(\Theta)$ and $H(\Theta|Y_x)$ using the following feature of particle systems.

Lemma 3.2.1 (Particle System Consistent Estimator)**.** A particle system with target distribution π provides a consistent (but biased) estimator $\hat{\mu}(h)$ of $\mathbb{E}_\pi[h(\theta)]$.

$$\hat{\mu}(h) = \frac{\sum_{i=1}^{k} w_i h(\theta_j)}{\sum_{i=1}^{k} w_i}$$

Proof. Proven by Doucet et al. in [5]. □

Using this lemma, we can define consistent estimators of $H(\Theta)$ and $H(\Theta|Y_x)$ as presented in Propositions 3.2.3 and 3.2.4. For the following, let $[(\theta_1, w_1), ..., (\theta_k, w_k)]$ be the particle system targeting the model distribution $P(\theta)$, and $\bar{w}_i = \frac{w_i}{\sum_{i=1}^{k} w_i}$ be the normalized weights of the particle system. The quantity $f(Y_x = y|\theta_i)$ is the posterior probability density given the model θ_i, which can be analytically computed.

Proposition 3.2.3 (Kernel Entropy Estimator)**.** Let $\hat{P}(\theta_i) = \bar{w}_i$, which is a consistent estimate of $P(\theta_i)$ according to Lemma 3.2.1. Then the following estimator $\hat{H}_k(\Theta)$ is a consistent estimator of $H(\Theta)$.

$$\hat{H}_k(\Theta) = -\sum_{i=1}^{k} \bar{w}_i \cdot \log\ \hat{P}(\theta_i)$$

Proof. Using Lemma 3.2.1

$$\begin{aligned}
\lim_{k\to\infty} \sum_{i=1}^{k} \bar{w}_i \cdot \log\ \hat{P}(\theta_i) &= -\mathbb{E}_{P(\theta)}[\log\ P(\theta)] \\
&= -\int P(\theta) \log P(\theta)\, d\theta \\
&= H(\Theta)
\end{aligned}$$

□

Proposition 3.2.4 (Kernel Conditional Entropy Estimator)**.** Let $\hat{P}(\theta_i) = \bar{w}_i$, which is a consistent estimate of $P(\theta_i)$ according to Lemma 3.2.1. Let $f(Y_x = y|\theta_i)$ be the posterior probability density calculated using the GP covariance kernel θ_i. Then the following estimator $\hat{H}_k(\Theta|Y_x)$ is a consistent estimator of $H(\Theta|Y_x)$.

$$\hat{H}_k(\Theta|Y_x) = \int \hat{f}_k(Y_x = y) \cdot \hat{H}_k(\Theta|Y_x = y)\, dy \tag{3.1}$$

$$\hat{H}_k(\Theta|Y_x = y) = -\sum_{i=1}^{k} \hat{P}_k(\theta_i|Y_x = y) \cdot \log\ \hat{P}_k(\theta_i|Y_x = y) \tag{3.2}$$

$$\hat{P}_k(\theta_i|Y_x = y) = \frac{f(Y_x = y|\theta_i) \cdot \hat{P}(\theta_i)}{\hat{f}_k(Y_x = y)} \tag{3.3}$$

$$\hat{f}_k(Y_x = y) = \sum_{i=1}^{k} \bar{w}_i \cdot f(Y_x = y|\theta_i) \tag{3.4}$$

Proof. Using Lemma 3.2.1, we first show $\hat{f}_k(Y_x = y)$ (Equation 3.4) is consistent.

$$\begin{aligned}
\lim_{k\to\infty} \sum_{i=1}^{k} \bar{w}_i \cdot f(Y_x = y|\theta_i) &= -\mathbb{E}_{P(\theta)}[f(Y_x = y|\theta_i)] \\
&= -\int P(\theta) f(Y_x = y|\theta_i)\, d\theta \\
&= f(Y_x = y)
\end{aligned}$$

From this, it follows that $\hat{P}_k(\theta_i|Y_x = y)$ (Equation 3.3) is consistent.

$$\begin{aligned}\lim_{k\to\infty} \frac{f(Y_x = y|\theta_i) \cdot \hat{P}(\theta_i)}{\hat{f}_k(Y_x = y)} &= \frac{f(Y_x = y|\theta_i) \cdot P(\theta_i)}{f(Y_x = y)} \\ &= P(\theta_i|Y_x = y)\end{aligned}$$

Using this and Lemma 3.2.1, we show $\hat{H}_k(\Theta|Y_x = y)$ (Equation 3.2) is consistent.

$$\begin{aligned}\lim_{k\to\infty} -\sum_{i=1}^{k} \hat{P}_k(\theta_i|Y_x = y) \cdot \log\, \hat{P}_k(\theta_i|Y_x = y) &= -\mathbb{E}_{P(\theta|Y_x=y)}[\log\, P(\theta|Y_x = y)] \\ &= -\int P(\theta|Y_x = y) \log P(\theta|Y_x = y)\, d\theta \\ &= H(\Theta|Y_x = y)\end{aligned}$$

Since we've shown $\hat{f}_k(Y_x = y)$ and $\hat{H}_k(\Theta|Y_x = y)$ are consistent, it follows that $\hat{H}_k(\Theta|Y_x)$ (Equation 3.1) is consistent.

$$\begin{aligned}\lim_{k\to\infty} \int \hat{f}_k(Y_x = y) \cdot \hat{H}_k(\Theta|Y_x = y)\, dy &= \int f(Y_x = y) \cdot H(\Theta|Y_x = y)\, dy \\ &= H(\Theta|Y_x)\end{aligned}$$

□

Note that Equations 3.3 and 3.4 reweight particles to shift the target distribution of the particle system from $P(\theta)$ to $P(\theta|Y_x = y)$, and Equation 3.2 estimates conditional entropy using the reweighted system. The next section describes how to approximate the indefinite integral over y in Equation 3.1.

Integrating over Y_x

To compute the conditional entropy estimator in in Proposition 3.2.4, we need to approximate the indefinite integrate over y in Equation 3.1. To do so, we can first rewrite the estimator as follows

$$\begin{aligned}\hat{H}_k(\Theta|Y_x) &= \int \hat{f}_k(Y_x = y) \cdot \hat{H}_k(\Theta|Y_x = y)\, dy \\ &= \int \sum_{i=1}^{k} \bar{w}_i \cdot f(Y_x = y|\theta_i) \cdot \hat{H}_k(\Theta|Y_x = y)\, dy \\ &= \sum_{i=1}^{k} \bar{w}_i \cdot \int f(Y_x = y|\theta_i) \cdot \hat{H}_k(\Theta|Y_x = y)\, dy\end{aligned}$$

and approximate the resulting indefinite integral over Y_x using a midpoint Riemann sum with m partitions over an interval $[a_i, b_i]$.

$$\int_{a_i}^{b_i} \hat{f}_k(Y_x = y|\theta_i) \cdot \hat{H}_k(\Theta|Y_x = y)\, dy \approx \frac{b_i - a_i}{m} \sum_{j=1}^{m} \hat{f}_k(Y_x = y_{i,j}|\theta_i) \cdot \hat{H}_k(\Theta|Y_x = y_{i,j})$$

where $y_{i,j}$ are m midpoints on the interval $[a_i, b_i]$.

$$y_{i,j} = (a_i + \frac{b_i - a_i}{2m}) + (j-1) \cdot \frac{b_i - a_i}{m}$$

We can make a reasonable choice for $[a_i, b_i]$ based on where we believe most of the probability mass of Y_x lies. Given the GP kernel θ_i, we can calculate $Y_x^{\theta_i}$, an estimate of the posterior distribution of Y_x, using the equations in section 3.3.1.

$$Y_x^{\theta_i} = \mathcal{N}(\mu_i, \sigma_i)$$

Using $\mathcal{N}(\mu_i, \sigma_i)$, we can define $[a_i, b_i]$ as $[\mu_i - 2\sigma_i, \mu_i + 2\sigma_i]$, since this interval captures $\sim 95\%$ of the probability mass of the predicted distribution of Y_x.

With this approximation method, our final estimator for $H(\Theta|Y_x)$ becomes

$$\hat{\hat{H}}_{m,k}(\Theta|Y_x) = \sum_{i=1}^{k} \bar{w}_i \cdot \frac{4\sigma_i}{m} \sum_{j=1}^{m} f(y_{i,j}|\theta_i) \cdot \hat{H}_k(\Theta|Y_x = y_{i,j})$$

such that

$$\lim_{k,m\to\infty} \hat{\hat{H}}_{m,k}(\Theta|Y_x) = \int_{(\mu-2\sigma)_{Y_x}}^{(\mu+2\sigma)_{Y_x}} f_k(Y_x = y) \cdot H_k(\Theta|Y_x = y)\, dy$$

3.2.3 IG-K Objective Pseudocode

Below is the pseudocode implementation of the kernel information gain (IG-K) objective that can be called on line 10 of Algorithm 2. $K(\theta, x, x')$ denotes the covariance matrix between all possible pairs (x, x') for a given covariance kernel function θ.

Algorithm 3 Kernel Information Gain (IG-K) Objective

1: **procedure** IGK-OBJ($\tilde{x}, [(\theta_i, w_i)]_{1:k}, [(x_j, y_j)]_{1:t-1}$)
2: **parameters**: m, number of partitions in Y_x integral approximation
3: $\bar{w}_i \longleftarrow w_i / \sum_{j=1}^{k} w_j$ for $i \in [1, k]$ ▷ Calculate normalized weights $\bar{w}_i$
4: $H \longleftarrow -\sum_{i=1}^{k} \bar{w}_i \cdot \log \bar{w}_i$ ▷ Calculate $\hat{H}_k(\Theta)$
5:
6: **for** $i \in [1, k]$ **do** ▷ Simulate observations $\tilde{y}_{i,j}$ at $\tilde{x}$
7: $\mu_i \longleftarrow K(\theta_i, \tilde{x}, x_{1:t-1}) K(\theta_i, x_{1:t-1}, x'_{1:t-1}) y_{1:t-1}$
8: $\sigma_i \longleftarrow \sqrt{K(\theta_i, \tilde{x}, \tilde{x}) - K(\theta_i, \tilde{x}, x_{1:t-1}) K(\theta_i, x_{1:t-1}, x'_{1:t-1})^{-1} K(\theta_i, \tilde{x}, x_{1:t-1})^T}$
9: $\tilde{y}_{i,j} \longleftarrow (\mu_i - 2\sigma_i + \frac{2\sigma_i}{m}) + (j-1) \cdot \frac{4\sigma_i}{m}$ for $j \in [1, m]$ ▷ Riemann midpoints $\tilde{y}_{i,j}$
10: **end for**
11: **for** $\tilde{y}_{i,j} \mid i \in [1, k], j \in [1, m]$ **do** ▷ Simulate entropy after observing $\tilde{y}_{i,j}$ at $\tilde{x}$
12: $\tilde{w}_l \longleftarrow w_l \cdot f(\tilde{y}_{i,j} \mid y_{1:t-1}, \theta_l \,;\, \tilde{x}, x_{1:t-1})$ for $l \in [1, k]$ ▷ Simulate reweighting by likelihood
13: $\bar{\tilde{w}}_l \longleftarrow \tilde{w}_l / \sum_{p=1}^{k} \tilde{w}_p$ for $l \in [1, k]$ ▷ Normalize simulated weights $\tilde{w}_l$
14: $\tilde{H}_{i,j} \longleftarrow -\sum_{l=1}^{k} \bar{\tilde{w}}_l \cdot \log \bar{\tilde{w}}_l$ ▷ Calculate $\hat{H}_k(\Theta | Y_x = y_{i,j})$
15: **end for**
16: $\tilde{H} \longleftarrow \sum_{i=1}^{k} \bar{w}_i \cdot \frac{4\sigma_i}{m} \sum_{j=1}^{m} f(\tilde{y}_{i,j} \mid y_{1:t-1}, \theta_i \,;\, \tilde{x}, x_{1:t-1}) \cdot \tilde{H}_{i,j}$ ▷ Calculate $\hat{H}_{m,k}(\Theta | Y_{\tilde{x}})$
17: **return** $H - \tilde{H}$ ▷ Return estimated information gain $I(\Theta; Y_{\tilde{x}})$
18: **end procedure**

Lines 1-2 estimate the kernel entropy $H(\Theta)$ following the procedure presented in Proposition 3.2.3. Lines 5-9 calculate the predictive intervals and Riemann midpoints $y_{i,j}$ for approximating the integral over Y_x as described in section 3.2.2. Lines 10-14 estimate the kernel entropies $H(\Theta | Y_x = y_{i,j})$ conditioned on observing $y_{i,j}$ at x as described in Proposition 3.2.4. Line 15 estimates the conditional entropy $H(\Theta | Y_x)$ as described in Proposition 3.2.4 and section 3.2.2. Finally, line 16 returns the estimated kernel information gain (IG-K) objective.

3.3 Predictive Information Gain (IG-P) Objective

Unlike the IG-K objective, which reduces uncertainty of the covariance kernel model, the predictive information gain (IG-P) objective reduces uncertainty of the model predictions by maximizing information gain on GP predictive posteriors (averaged over a region of interest). The IG-P objective directly extends the Active Learning Cohn (ALC) GP active learning criterion implemented by Seo et al. in [17] which assumes the GP model is known.

Section 3.3.1 formulates the IG-P objective assuming the model is known. Section 3.3.2 describes how to estimate the objective and extend it to our active learning setting where the model is unknown using particle filter approximation. Section 3.3.3 presents the pseudocode for estimating the objective.

3.3.1 IG-P Objective

In this section, assume the GP model θ is known. Let Y_x^θ be a random variable from the GP predictive distribution at x conditioned on θ. We can also calculate the posterior conditioned on $Y_{\tilde{x}}$, observations at other locations $\tilde{x}$

$$\begin{aligned} P(Y_x^\theta) &= \mathcal{N}(\mu(x), \sigma^2(x)) \\ P(Y_x^\theta | Y_{\tilde{x}}^\theta) &= \mathcal{N}(\mu(x, \tilde{x}), \sigma^2(x, \tilde{x})) \end{aligned}$$

using the following equations, letting $K(\theta, x, x')$ denote the covariance matrix between all possible pairs (x, x') for the covariance kernel θ and $\tilde{y}$ denote the observed values at $\tilde{x}$ [13].

$$\mu^2(x) = 0 \tag{3.5}$$

$$\mu^2(x, \tilde{x}) = K(\theta, x, \tilde{x}) K(\theta, \tilde{x}, \tilde{x}')^{-1} \tilde{y} \tag{3.6}$$

$$\sigma^2(x) = K(\theta, x, x) \tag{3.7}$$

$$\sigma^2(x, \tilde{x}) = K(\theta, x, x) - K(\theta, x, \tilde{x}) K(\theta, x, \tilde{x}')^{-1} K(\theta, x, \tilde{x})^T \tag{3.8}$$

Now assume we are interested in predictions over the interval $x \in [a, b]$. Let $\mathcal{Y}_{a,b}^\theta$ be a random variable representing the average value of Y_x^θ over the interval $x \in [a, b]$. Let $\mathcal{Y}_{a,b}^\theta | Y_{\tilde{x}}^\theta$ be the posterior given observations at locations $\tilde{x}$.

$$\mathcal{Y}_{a,b}^\theta = \int_a^b \frac{1}{b-a} \cdot (Y_x^\theta)\, dx$$

$$\mathcal{Y}_{a,b}^\theta | Y_{\tilde{x}}^\theta = \int_a^b \frac{1}{b-a} \cdot (Y_x^\theta | Y_{\tilde{x}}^\theta)\, dx$$

Active learning with IG-P selects $x* \in \mathcal{X}$ to maximize information gain on the entropy of $\mathcal{Y}^{\theta}_{a,b}$, conditioned on the unknown observation Y^{θ}_{x*} at $x*$:

$$\begin{aligned} I(\mathcal{Y}^{\theta}_{a,b}; Y^{\theta}_{\tilde{x}}) &= H(\mathcal{Y}^{\theta}_{a,b}) - H(\mathcal{Y}^{\theta}_{a,b}|Y^{\theta}_{\tilde{s}}) \\ x* &= \underset{\tilde{x}\in\mathcal{X}}{\operatorname{argmax}}\, I(\mathcal{Y}^{\theta}_{a,b}; Y^{\theta}_{\tilde{x}}) \end{aligned}$$

3.3.2 Estimating IG-P Objective

Approximating Predictive Entropy

We define the predictive entropy $H(\mathcal{Y}^{\theta}_{a,b})$ using differential entropy of a normal distribution as follows, where $\sigma^2(x)$ is defined using Equation 3.7 in the previous section.

$$\begin{aligned} h(Y^{\theta}_{x}) &= h(\mathcal{N}(\mu(x), \sigma^2(x))) = \ln(\sqrt{\sigma^2(x) \cdot 2\pi e}) \\ H(\mathcal{Y}_{a,b}) &= \int_a^b \frac{1}{b-a} \cdot h(Y^{\theta}_{x})\, dx \\ &= \int_a^b \frac{1}{b-a} \cdot \ln(\sqrt{\sigma^2(x) \cdot 2\pi e})\, dx \end{aligned}$$

We define the conditional entropy $H(\mathcal{Y}_{a,b}|\tilde{x})$ similarly, where $\sigma^2(x, \tilde{x})$ is defined using Equation 3.8 in the previous section.

$$\begin{aligned} h(Y^{\theta}_{x}|Y_{\tilde{x}}) &= h(\mathcal{N}(\mu(x, \tilde{x}), \sigma^2(x, \tilde{x}))) = \ln(\sqrt{\sigma^2(x, \tilde{x}) \cdot 2\pi e}) \\ H(\mathcal{Y}^{\theta}_{a,b}|Y_{\tilde{x}}) &= \int_a^b \frac{1}{b-a} \cdot h(Y^{\theta}_{x}|Y_{\tilde{x}})\, dx \\ &= \int_a^b \frac{1}{b-a} \cdot \ln(\sqrt{\sigma^2(x, \tilde{x}) \cdot 2\pi e})\, dx \end{aligned}$$

We can approximate the integral over x by a midpoint Riemann sum with m partitions.

IG-P Objective with Unknown Model

So far, we have assumed the model GP kernel θ is known, which is not true for our active learning setting. We can calculate the entropies when the model is unknown, $H(\mathcal{Y}_{a,b})$ and $H(\mathcal{Y}_{a,b}|Y_{\tilde{x}})$, as expectations over the model distribution $P(\theta)$, and estimate this by taking a

weighted average over a particle system as described in section 3.2.2.

$$
\begin{aligned}
H(\mathcal{Y}_{a,b}) &= \mathbb{E}_{P(\theta)}\left[\int_a^b \frac{1}{b-a} \cdot h(Y_x^\theta)\, dx\right] \\
H(\mathcal{Y}_{a,b}|\tilde{x}) &= \mathbb{E}_{P(\theta)}\left[\int_a^b \frac{1}{b-a} \cdot h(Y_x^\theta|Y_{\tilde{x}})\, dx\right]
\end{aligned}
$$

To approximate this expectation, let $[(\theta_1, w_1), ..., (\theta_k, w_k)]$ be the particle system targeting $P(\theta)$, and $\bar{w}_i = \frac{w_i}{\sum_{i=1}^k w_i}$ be the normalized weights of the particle system. The resulting estimators, presented below, are consistent by Lemma 3.2.1.

$$
\begin{aligned}
\hat{H}_k(\mathcal{Y}_{a,b}) &= \sum_{i=1}^k \bar{w}_i \cdot \int_a^b \frac{1}{b-a} \cdot h(Y_x^{\theta_i})\, dx \\
\hat{H}_k(\mathcal{Y}_{a,b}|\tilde{x}) &= \sum_{i=1}^k \bar{w}_i \cdot \int_a^b \frac{1}{b-a} \cdot h(Y_x^{\theta_i})\, dx
\end{aligned}
$$

We can approximate the integral over x by a midpoint Riemann sum with m partitions.

3.3.3 IG-P Objective Pseudocode

Below is the pseudocode implementation of the predictive information gain (IG-P) objective that can be called on line 10 of Algorithm 2. For the implementation of IG-P used in this thesis, we set the region of interest $[a, b]$ to the range of observation locations $\mathcal{X}$.

Algorithm 4 Predictive Information Gain (IG-P) Objective

1: **procedure** IGP-OBJ($\tilde{x}$, $[(\theta_i, w_i)]_{1:k}$, $[(x_j, y_j)]_{1:t-1}$)
2: **parameters**: $\mathcal{X}$, potential observation locations; m, number of partitions in x integral approximation
3: $\bar{w}_i \longleftarrow w_i / \sum_{j=1}^k w_j$ for $i \in [1, k]$ ▷ Calculate normalized weights $\bar{w}_i$
4: $x_t \longleftarrow \tilde{x}$
5: $a, b \longleftarrow min(\mathcal{X}), max(\mathcal{X})$ ▷ Set region of interest to range of $\mathcal{X}$
6: $\hat{x}_j \longleftarrow a + \frac{b-a}{2m} + (j-1) \cdot \frac{b-a}{m}$ for $j \in [1, m]$ ▷ Riemann midpoints
7: **for** $\theta_i, \hat{x}_j \mid i \in [1, k], j \in [1, m]$ **do** ▷ Calculate $\sigma^2(x)$ and $\sigma^2(x, \tilde{x})$
8: $(\sigma^2)_{i,j} \longleftarrow K(\theta_i, \hat{x}_j, \hat{x}_j) - K(\theta_i, \hat{x}_j, x_{1:t-1}) K(\theta_i, x_{1:t-1}, x'_{1:t-1})^{-1} K(\theta_i, \hat{x}_j, x_{1:t-1})^T$
9: $(\tilde{\sigma}^2)_{i,j} \longleftarrow K(\theta_i, \hat{x}_j, \hat{x}_j) - K(\theta_i, \hat{x}_j, x_{1:t}) K(\theta_i, x_{1:t}, x'_{1:t})^{-1} K(\theta_i, \hat{x}_j, x_{1:t})^T$
10: **end for**
11: $H \longleftarrow \sum_{i=1}^k \bar{w}_i \cdot \frac{1}{m} \sum_{j=1}^m \ln(\sqrt{(\sigma^2)_{i,j} \cdot 2\pi e})$ ▷ Estimate $H(\mathcal{Y}_{a,b})$
12: $\tilde{H} \longleftarrow \sum_{i=1}^k \bar{w}_i \cdot \frac{1}{m} \sum_{j=1}^m \ln(\sqrt{(\tilde{\sigma}^2)_{i,j} \cdot 2\pi e})$ ▷ Estimate $H(\mathcal{Y}_{a,b}|Y_{\tilde{x}})$
13: **return** $H - \tilde{H}$ ▷ Return estimated information gain $I(\mathcal{Y}_{a,b}; Y_{\tilde{x}})$
14: **end procedure**

Lines 7-9 calculate the predictive variance for each kernel θ_i and Riemann midpoint $\hat{x}_j$ according to the formulas presented in section 3.3.1, both with and without conditioning on the observation $Y_{\tilde{x}}$ at $\tilde{x}$. Lines 11-12 estimate $H(\mathcal{Y}_{a,b})$ and $H(\mathcal{Y}_{a,b}|\tilde{x})$ according to the

procedure outlined in section 3.3.2. Note that $\frac{1}{m}$ factor is the product of the Riemann partition width $\frac{b-a}{m}$ and $\frac{1}{b-a}$, the density of x. Line 13 returns the estimated predictive information gain (IG-P) objective.

. . .

The Monte Carlo estimators formulated in this chapter allow us to tractably compute approximations of the kernel information gain (IG-K) and predictive information gain (IG-P) objectives for SMC-based Bayesian active learning as presented at the beginning of the chapter. In the next chapter, we test Bayesian active learning with IG-K and IG-P on synthetic datasets to compare how well each objective performs, in practice, at reducing model uncertainty and providing accurate predictions of unknown data.

Chapter 4

Validating Grid Approximated Kernel Information Gain (IG-K) Objective

The previous chapter presents an SMC algorithm for Bayesian active structure learning with GP probabilistic programs, based on Monte Carlo approximations of two objectives: kernel information gain (IG-K) and predictive information gain (IG-P). To evaluate and compare the efficacy of the approximated IG-K and IG-P objectives at model inference and accurately predicting data, we ran a series of experiments testing the active learning algorithm with both objectives on synthetic datasets drawn from one-dimensional GPs.

To avoid the variance and potential inference error introduced by resampling and rejuvenation, we estimate the model distribution $P(\theta)$ using grid approximation: we enumerate discrete structures s and grid over numeric parameters υ in the GP model class to obtain a set of sample θ for the particle filter meant to approximate the entire model class; hence, the particles do not need to be resampled or rejuvenated. Unfortunately it is intractable to grid over the infinite model class of all possible models θ in the DSL (Section 2.1). Thus, for these experiments, we restrict the model class and change the covariance prior (Section 2.1.2) — e.g. restricting the model class to *only* periodic covariance kernels by setting $P(\texttt{PER}) = 1$ — and we grid over the restricted model class.

Section 4.1 presents the metrics used to assess active learning performance (4.1.1), the experimental procedure (4.1.2), and results of experiments (4.1.3) where the model class is constrained to only periodic kernels; only linear kernels; only squared exponential kernels; and periodic, linear, and periodic + linear kernels respectively. Section 4.2 presents a qualitative summary of the experiment results (4.2.1) and discussion and analysis of the results (4.2.2, 4.2.3), supported by visualizations (4.2.4) of how the information gain and estimated posterior evolves over time in one run of active learning on a particular synthesized dataset.

4.1 Experiments

4.1.1 Evaluation Metrics

To measure the efficacy of active learning, we define and record the following metrics.

- **Predictive MSE -** the mean squared error of the GP predictive mean, calculated as a weighted average over gridded models according to their estimated posterior likelihood. This measures how accurately the inferred models are able to predict unknown data, given the data observed at locations chosen by active selection.
- **Parameter MSE -** the mean squared error of the estimated kernel parameters against the true parameters used to generate the test dataset, also calculated as a weighted average. This measures how well active learning infers the correct model parameters for the data. It is not possible to calculate parameter MSE among kernels with different structures, since different structures have different parameters.
- **Noise MSE -** the mean squared error of the estimated noise parameter against the true noise used to generate the test dataset, also calculated as a weighted average. This measures how well active learning infers the correct noise parameter for the data.
- **Ground truth probability -** the estimated posterior probability of the ground truth model used to generate the dataset, which is always explicitly within the grid of model structures and parameters. This measures how well active learning infers the exact model parameters for the data.
- **Structure posterior -** in experiments where the model class is restricted to more than one type of kernel structure, we record the total estimated posterior weight placed on models of each structure type. This measures how well active learning infers the correct kernel structure for the data.

4.1.2 Experiment Procedure

For each experiment, we follow the procedure below.

1. Restrict the model class to a particular set of covariance structures and grid over parameterizations υ to obtain a set of models for our grid approximation. Since kernel parameters υ are drawn from a Uniform(0,1) prior (see section 2.1.2), we grid each parameter over the range $[0, 1]$ at a width of 0.1 or 0.2. Thus, we grid over kernel parameters from the sets $S_{0.1} = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$ and $S_{0.2} =$

$\{0.2, 0.4, 0.6, 0.8, 1.0\}$. We fix the noise parameter η of gridded models to 0.1, the ground truth η used to generate the datasets. For most experiments, we also present a separate set of results where η is unknown, and we grid over 5 values of η from a high probability region of the Gamma(1,1) noise prior: $N = \{0.3, 0.4, 0.5, 0.6, 0.7\}$

2. Run grid-approximated Bayesian active learning with IG-K and IG-P objectives and a budget of 16 observations on a number of synthetic datasets generated from GP models in the grid. Each dataset consists of 100 observations evenly spaced over the range [-1,1]. Similar to the procedure in experiments by Seo et al. [17], for each of our experiments, the first observation is fixed to be the center point of the dataset, and the remaining observations are chosen via active selection from the remaining locations in the dataset.

3. Record appropriate evaluation metrics (defined above) at each timestep (i.e. after each new observation), averaged over all dataset runs.

4.1.3 Experiment Results

This section presents results of experiments where we restrict the model class to the following GP covariance kernel types. Section 4.2.1 provides a qualitative summary of the results presented in this section.

- **Periodic-Only** (Figures 4-1 and 4-2)
- **Linear-Only** (Figures 4-3 and 4-4)
- **Squared Exponential-Only** (Figures 4-5 and 4-6)
- **Periodic, Linear, Periodic + Linear** (Figures 4-7, 4-8, and 4-9)

For each experiment, we describe the set of gridded GP models; we grid over kernel and noise parameters using the sets $S_{0.1}$, $S_{0.2}$, and N (defined in experiment procedure, step 1). We present appropriate evaluation metrics averaged over all dataset runs at all timesteps plotted with standard error ribbons. We also present values of the metrics at timesteps 1, 5, 10, and 15 in a table, where **bolded values** indicate a metric is significantly better (e.g. lower MSE or higher ground truth probability) according to the standard error ($p \leq 0.32$). For the first three experiments above, we also present results of where we grid over the noise η. We avoid this for the fourth experiment (Periodic, Linear, Periodic + Linear) due to computational constraints as the model grid is significantly larger.

Periodic-Only Model Class Experiment Results

Observations	1	5		10		15	
		IG-P	IG-K	IG-P	IG-K	IG-P	IG-K
Predictive MSE	123.96	124.51	**104.25**	98.73	**81.41**	**64.07**	70.24
Scale MSE	0.165	0.1608	**0.1539**	0.1123	**0.085**	0.067	**0.0461**
Period MSE	0.165	0.16	**0.0921**	0.1214	**0.0453**	0.0801	**0.0191**
Ground Truth Prob.	0.01	0.0136	**0.0482**	0.0477	**0.1538**	0.114	**0.256**

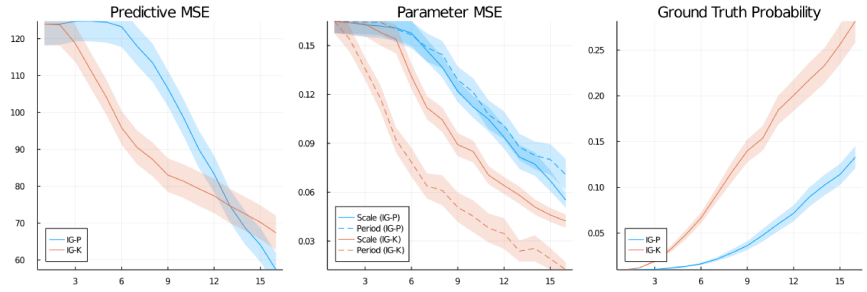


Figure 4-1: **Periodic-Only Model Class with Fixed Noise.** Results of experiments where we restrict the model to only periodic kernels, grid over the scale and period parameters, and fix noise to the correct amount. Scale and period values come from the set $S_{0.1}$, so the grid consists of 100 periodic GP models parameterized by $S_{0.1} \times S_{0.1}$. We draw one dataset from each GP model in the grid and present result metrics averaged over all 100 datasets.

Observations	1	5		10		15	
		IG-P	IG-K	IG-P	IG-K	IG-P	IG-K
Predictive MSE	144.03	142.85	**131.07**	126.64	**107.81**	109.01	90.175
Scale MSE	0.16	0.1581	0.1575	0.1406	0.1268	0.1091	0.109
Period MSE	0.16	0.1578	**0.1186**	0.1385	**0.0684**	0.1005	**0.0408**
Ground Truth Prob.	0.0097	0.0103	**0.0202**	0.0188	**0.0474**	0.0322	**0.0659**
Noise MSE	0.0394	0.0388	0.0384	0.0376	0.0362	0.0355	**0.0333**

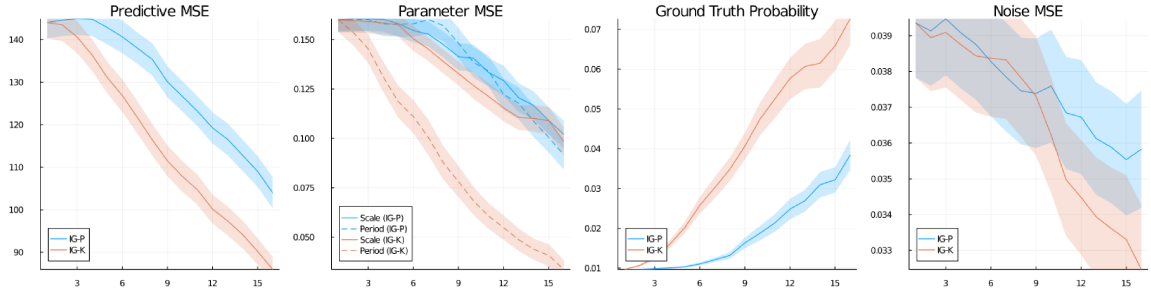


Figure 4-2: **Periodic-Only Model Class with Gridded Noise.** Results of experiments where we restrict the model to only periodic kernels and grid over the scale, period, and noise parameters. Scale and period values come from the set $S_{0.2}$, and noise values come from the set N, so the grid consists of 125 periodic GP models parameterized by $S_{0.2} \times S_{0.2} \times N$. We draw one dataset from each GP model and present result metrics averaged over all 125 datasets.

Linear-Only Model Class Experiment Results

Observations	1	5		10		15	
		IG-P	IG-K	IG-P	IG-K	IG-P	IG-K
Predictive MSE	30.332	**17.364**	21.1848	16.838	16.7456	16.234	**14.3877**
Parameter MSE	0.1553	0.1494	**0.0925**	0.1481	**0.0648**	0.1448	**0.0553**
Ground Truth Prob.	0.1079	0.1107	**0.1664**	0.1115	**0.2341**	0.1127	**0.2635**

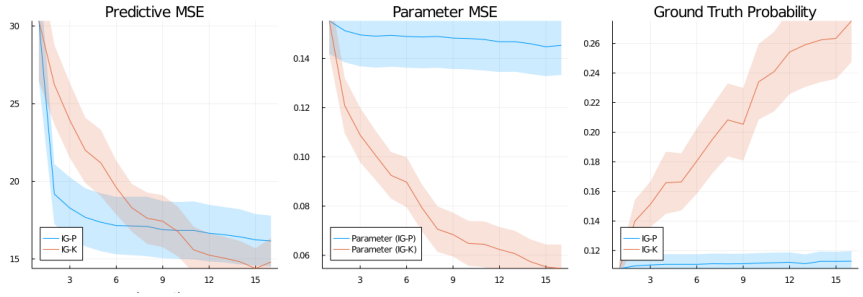


Figure 4-3: **Linear-Only Model Class with Fixed Noise.** Results of experiments where we restrict the model to only linear kernels, grid over the slope variance parameter, and fix noise to the correct amount. Slope variance parameter values in the grid come from the set $S_{0.1}$, so the grid consists of 10 linear GP models. We draw five datasets from each GP model and present result metrics averaged over all 50 datasets.

Observations	1	5		10		15	
		IG-P	IG-K	IG-P	IG-K	IG-P	IG-K
Predictive MSE	94.921	60.8896	63.557	58.455	58.117	57.9755	55.787
Parameter MSE	0.1444	0.1287	**0.0993**	0.127	**0.0877**	0.1248	**0.0702**
Ground Truth Prob.	0.0275	0.0274	**0.0387**	0.0265	**0.0482**	0.0256	**0.0552**
Noise MSE	0.0401	0.0347	0.0346	0.0254	0.0265	0.0212	0.0222

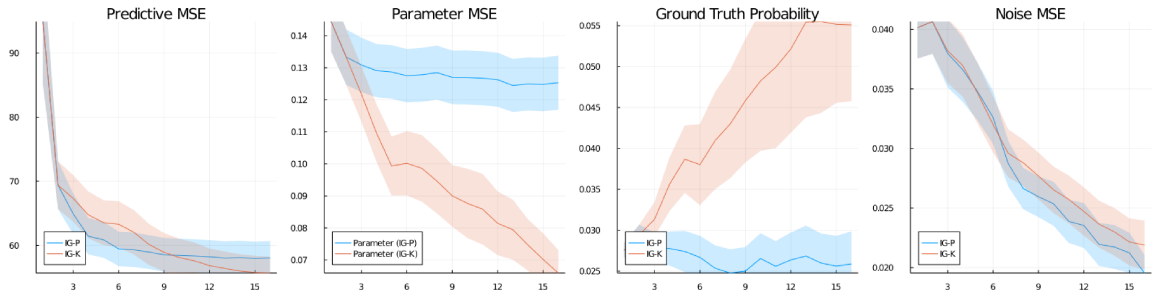


Figure 4-4: **Linear-Only Model Class with Gridded Noise** Results of experiments where we restrict the model to only linear kernels and grid over the slope variance and noise parameters. Slope variance and noise parameter values in the grid come from the sets $S_{0.1}$ and N, respectively, so the grid consists of 50 linear GP models parameterized by $S_{0.1} \times N$. We draw one dataset from each GP model and present result metrics averaged over all 50 datasets.

Squared Exponential-Only Model Class Experiment Results

Observations	**1**	**5**		**10**		**15**	
		IG-P	**IG-K**	**IG-P**	**IG-K**	**IG-P**	**IG-K**
Predictive MSE	88.307	**30.974**	47.224	**17.541**	27.85	**14.141**	21.502
Parameter MSE	0.165	0.0999	0.0888	0.0578	0.0542	0.0529	0.0457
Ground Truth Prob.	0.1	0.1495	0.1924	0.2515	0.2831	0.277	0.2993

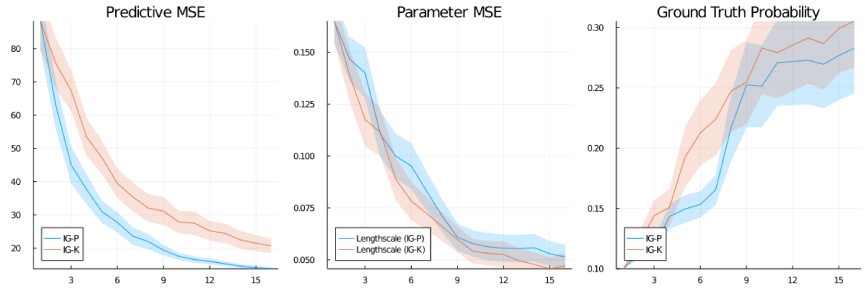


Figure 4-5: **Squared Exponential-Only Model Class with Fixed Noise.** Results of experiments where we restrict the model to only squared exponential kernels, grid over the lengthscale parameter, and fix noise to the correct amount. Lengthscale parameter values in the grid come from the set $S_{0.1}$, so the grid consists of 10 squared exponential GP models. We draw five datasets from each GP model in the grid and present result metrics averaged over all 50 datasets.

Observations	**1**	**5**		**10**		**15**	
		IG-P	**IG-K**	**IG-P**	**IG-K**	**IG-P**	**IG-K**
Predictive MSE	116.09	80.225	79.276	66.502	65.365	60.454	59.070
Lengthscale MSE	0.165	0.1335	0.1314	0.117	0.0971	0.1057	**0.0772**
Ground Truth Prob.	0.0243	0.0293	0.0325	0.0356	**0.0562**	0.0439	**0.0703**
Noise MSE	0.0398	0.0354	0.0393	0.034	0.0369	0.0313	0.0313

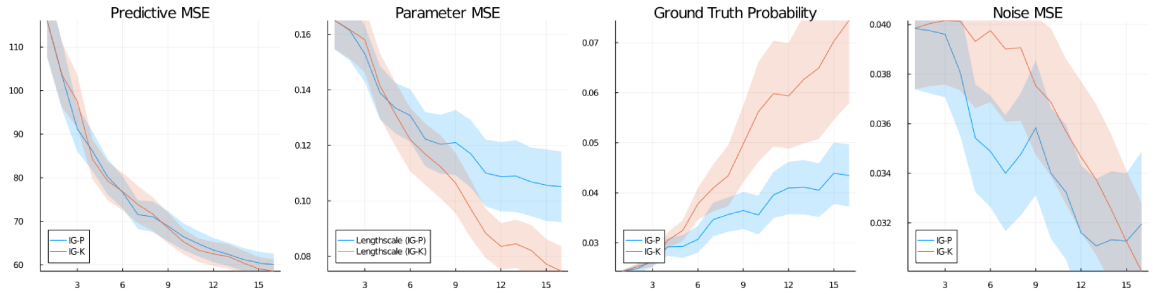


Figure 4-6: **Squared Exponential-Only Model Class with Gridded Noise.** Results of experiments where we restrict the model to only squared exponential kernels and grid over the lengthscale and noise parameters. Lengthscale and noise parameter values in the grid come from the sets $S_{0.1}$ and N, respectively, so the grid consists of 50 squared exponential GP models parameterized by $S_{0.1} \times N$. We draw one dataset from each GP model and present result metrics averaged over all 50 datasets.

Periodic, Linear, Periodic + Linear Model Class Experiment Results

The following figures show the results of experiments where we restrict the model to only periodic, linear, and periodic + linear kernels. We grid over parameterizations of the three possible structures and fix the noise to the correct amount. The kernel parameters are scale and period for the periodic kernel, and slope variance for the linear kernel. Parameter values in the grid come from the set $S_{0.2}$, so the grid consists of 155 GP models as follows:

(a) 25 periodic models with scales and periods in $S_{0.2} \times S_{0.2}$

(b) 5 linear models with slope variances in $S_{0.2}$

(c) 125 periodic + linear models composed of kernel pairs in (a) $\times$ (b)

We draw one dataset from each GP model in the grid and present result metrics averaged over all 155 datasets. In figures 4-7, 4-8, and 4-9 we present the results on datasets drawn from (a), (b), and (c), respectively. For each set of results, the parameter MSE is averaged over models with the correct structure. The structure posterior metric measures the total normalized posterior weight placed on each structure type. The covariance prior for these experiments is $\{0.3, 0.3, 0.4\}$ for periodic, linear, and plus kernels, respectively.

Observations	**1**	**5**		**10**		**15**	
		IG-P	**IG-K**	**IG-P**	**IG-K**	**IG-P**	**IG-K**
Predictive MSE	116.86	104.25	91.982	76.038	72.122	50.683	61.487
Scale MSE	0.1338	0.1486	0.1376	0.1049	0.0944	0.0585	0.0619
Period MSE	0.1328	0.1502	**0.0729**	0.1158	**0.0152**	0.0748	**0.0109**
Ground Truth Prob.	0.0224	0.0376	**0.1016**	0.0978	**0.2174**	0.2134	**0.3234**
Correct Structure Prob.	0.5597	0.7035	**0.7641**	0.7594	**0.8151**	0.8006	**0.8597**

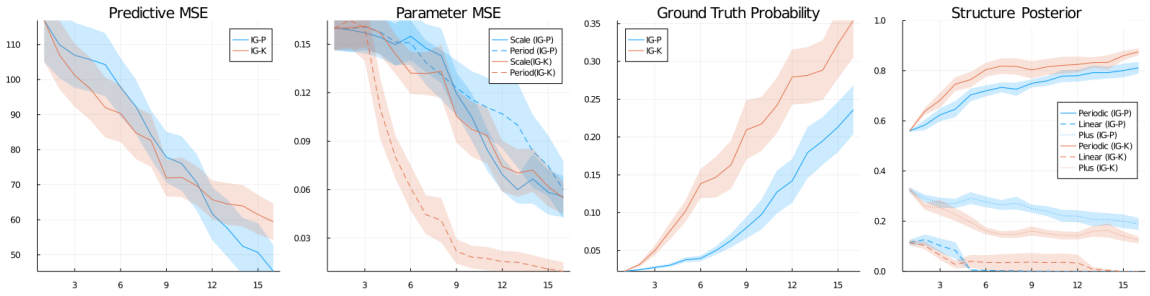


Figure 4-7: **Periodic Datasets Only.** Results averaged over 25 periodic datasets drawn from (a).

Observations	1	5		10		15	
		IG-P	IG-K	IG-P	IG-K	IG-P	IG-K
Predictive MSE	24.445	16.154	16.147	12.925	12.474	12.09	11.337
Parameter MSE	0.1475	0.145	**0.067**	0.0962	**0.0706**	0.0922	**0.0661**
Ground Truth Prob.	0.0315	0.1583	**0.1747**	0.2969	**0.3491**	0.3079	**0.405**
Correct Structure Prob.	0.1501	**0.7828**	0.5689	0.9671	0.9429	0.9896	0.9909

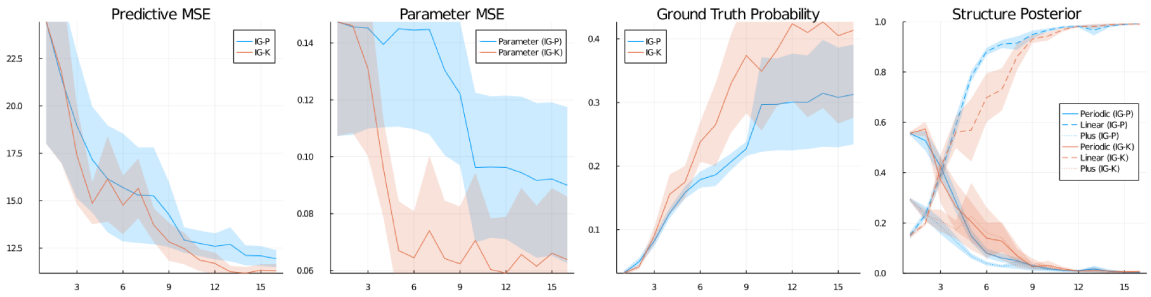


Figure 4-8: **Linear Datasets Only.** Results averaged over 5 linear datasets drawn from (b).

Observations	1	5		10		15	
		IG-P	IG-K	IG-P	IG-K	IG-P	IG-K
Predictive MSE	155.74	114.08	**101.19**	86.991	76.8	**53.218**	62.116
Scale MSE	0.16	0.1567	**0.1478**	0.1237	**0.09**	0.0617	0.0556
Period MSE	0.16	0.153	**0.079**	0.1193	**0.0267**	0.0648	**0.0083**
Parameter MSE	0.1578	0.148	0.1432	0.1394	0.1425	0.1356	0.1383
Ground Truth Prob.	0.0027	0.0047	**0.0203**	0.0135	**0.0575**	0.036	**0.0807**
Correct Structure Prob.	0.3372	0.437	**0.5693**	0.5182	**0.7162**	0.6191	**0.7587**

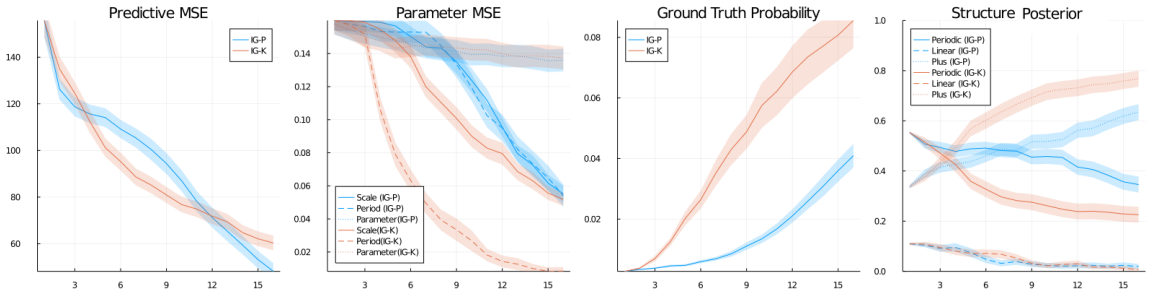


Figure 4-9: **Plus Datasets Only.** Results averaged over 125 periodic + linear datasets drawn from (c).

4.2 Analysis of Results

This section provides a qualitative analysis of the experiment results presented in section 4.1.3. We compare how effectively active learning with the IG-K and IG-P objectives (1) infers the correct model parameters and (2) makes accurate predictions, based on (1) reduction in parameter MSE and increase in ground truth probability metrics, and (2) reduction in predictive MSE. We summarize these comparisons in 4.2.1. In 4.2.2 and 4.2.3, we discuss possible reasons for the observed differences between the IG-K and IG-P objectives at inferring correct model parameters and making accurate predictions. To support our reasoning and further examine the dynamics of the IG-K and IG-P objectives, we present visualizations in 4.2.4 of how GP data predictions, information gain objectives, and the estimated model posterior evolve over time in one run of active learning on a chosen dataset.

4.2.1 IG-K vs. IG-P Summary

The noise MSE results are not compared since they exhibit too much variance to show meaningful differences between IG-K and IG-P. However, results from the gridded noise experiments (where ground truth noise η is higher) allow us to observe the effect of increased noise on model inference and predictive accuracy. Note that the use of the term *significant* here refers to significance with regards to the standard error ($p \leq 0.32$). While this is more lenient than the standard $p \leq 0.05$ significance level, we note that our grid validation experiments are in a different setting than experiments in the real world. Because gridding leads to entirely deterministic results, and our standard error is computed over the set of test datasets, significance at the $p \leq 0.32$ level means that approximately 68% or more of the test datasets perform better on a particular metric when using IG-K vs. IG-P (or vice versa).

Periodic-Only (Figures 4-1 and 4-2)

- **Inferring Model Parameters** IG-K is significantly better than IG-P at inferring the correct period and scale parameters in the presence of lower noise. With more noise, IG-K is unable to infer the correct scale parameter as well, and performs only slightly better than IG-P.
- **Predictive Accuracy** IG-K is significantly better at predicting the data than IG-P, except in the last few timesteps of the lower noise experiments. In figure 4-10, we present an example of active learning on a particular dataset in which we obtain this surprising result.

Linear-Only (Figures 4-3 and 4-4)

- **Inferring Model Parameters** IG-K is significantly better than IG-P at inferring the correct slope variance parameter.
- **Predictive Accuracy** IG-K is significantly worse at predicting the data than IG-P, requiring more observations to achieve the same predictive accuracy as IG-P.

Squared Exponential-Only (Figures 4-5 and 4-6)

- **Inferring Model Parameters** IG-K is slightly better than IG-P at inferring the correct lengthscale parameter in the presence of higher noise. With less noise, both objectives perform similarly.
- **Predictive Accuracy** IG-K is significantly worse at predicting the data than IG-P, in the presence of lower noise. With more noise, both objectives perform similarly.

Periodic, Linear, Periodic + Linear (Figures 4-7, 4-8, and 4-9)

- **Inferring Model Parameters** Consistent with previous results, IG-K is significantly better than IG-P at inferring the correct period and slope variance for periodic and linear datasets, respectively. For periodic + linear datasets, IG-K is significantly better at inferring period and scale of the periodic kernel.
- **Model Structure Inference** Based on the structure posterior metric, IG-K is significantly better than IG-P at inferring the correct structure for the periodic and periodic + linear datasets, while in linear datasets, IG-K does slightly worse at inferring the correct structure.
- **Predictive Accuracy** Both objectives perform about the same at predicting the data for the periodic and linear datasets, and for the periodic + linear datasets, we see a similar pattern as in the Periodic-only experiment, where IG-K predicts the data significantly better until the last few timesteps.

4.2.2 Inferring Model Parameters

Overall, the experiments demonstrate that IG-K is better than IG-P at inferring the correct GP model parameters, reducing the parameter MSE more with fewer observations. **The degree to which IG-K is better able to infer the correct model parameters varies depending on the GP parameter's effect on the generated data.** For example, IG-K is able to consistently infer the period parameter of the periodic kernel far better than IG-P since

the period's effect on the data is predictable: if the period is p, then data at two locations p apart will have the same value, with any disparity attributed only to noise. Thus, it is possible to select observation locations methodically spaced apart to confirm or eliminate hypothesized period values. In figure 4-10 of the next section, we provide evidence that the IG-K objective induces this strategy to reduce period parameter uncertainty.

IG-K is less effective at inferring other GP parameters, such as the scale parameter of the periodic kernel and the lengthscale parameter of the squared exponential kernel. The effect of these parameters on the data is the same: a lengthscale or scale of ℓ means that data at a location x is highly correlated with data at locations within ℓ of x. Interestingly, we notice this effect causes the IG-K objective to favor selecting observations located near each other to discern the precise lengthscale or scale, as shown in figure 4-11. According to Duvenaud, these scale parameters determine "the length of the wiggles" in the function [7]. Since kernel noise causes a similar effect ("wiggles") in the data, IG-K is less effective at inferring the the correct these scale parameters than the period parameter, since it is difficult infer whether observed "wiggles" are caused by noise or the kernel parameter. Consequently, the results of the periodic-only experiments show that IG-K is worse at inferring scale when noise η is higher.

4.2.3 Predictive Accuracy

The summary in 4.2.1 reveals that even though IG-K is better than IG-P at inferring the correct model parameters, knowledge of the correct GP model parameters does not necessarily result in more accurate predictions of the data. **Even with the correct model, sufficient data is necessary in order to make accurate predictions. Additionally, an observation selection strategy suited for inferring the correct model parameters may not favorable for providing accurate predictions.**

Surprisingly, in experiments where IG-K reduces parameter MSE much more than IG-P does, IG-K *does not* reduce predictive MSE more than IG-P does. In some cases, such as the linear-only experiments, IG-K is actually *worse* than IG-P at reducing predictive MSE. For the linear kernel, the GP parameter determines the variance of the slope of the data, so knowing the parameter is less useful than knowing the slope predicting the test data. Since IG-K selects observations to learn this parameter rather than learning the slope itself, IG-K requires more observations than IG-P to reduce predictive MSE. On the other hand, IG-P is able to rapidly reduce the predictive MSE by inferring the slope with two well-spaced observations via simple linear regression.

Similarly, the periodic kernel scale and squared exponential kernel lengthscale parame-

ters (discussed in 4.2.2) are not directly useful for prediction. Since these parameters only tell us the correlation among data points near each other, making accurate predictions using these parameters requires data near where we are trying to predict. Due to this, the IG-K objective's tendency to select nearby points to learn the lengthscale (discussed in 4.2.2) is actually an extremely poor strategy for reducing predictive MSE with the squared exponential kernel, since only observing data located in one small region makes it difficult to predict data beyond that region. This helps explain why IG-K is significantly worse than IG-P at reducing predictive MSE in the squared exponential-only experiments.

The period parameter of the periodic kernel is relatively more informative in making accurate predictions due to the repetitive nature of periodic functions, as discussed in 4.2.2. With the correct period and one datapoint, we can predict data at any number of period lengths from the datapoint. Given this property, and the IG-K objective's efficiency in inferring the correct period (see figure 4-10), it is unsurprising that the periodic-only experiment initially shows IG-K is better than IG-P at reducing predictive MSE. However, by the last few observations, IG-P is able to reduce predictive MSE further than IG-K. To help understand this puzzling result, we present a visualization (figure 4-11) of active learning with IG-K and IG-P on a periodic dataset which exhibits this trend. The visualization shows that the observation selection strategy induced by the IG-K objective to quickly infer the correct period is not favorable for improving predictions, since repeatedly selecting points exactly one period apart does not provide new information about the function shape between those points. By the last few observations, the IG-P objective also learns the correct model parameters and has previously selected points which provide more information for prediction than IG-K.

4.2.4 Visualizing IG-K and IG-P

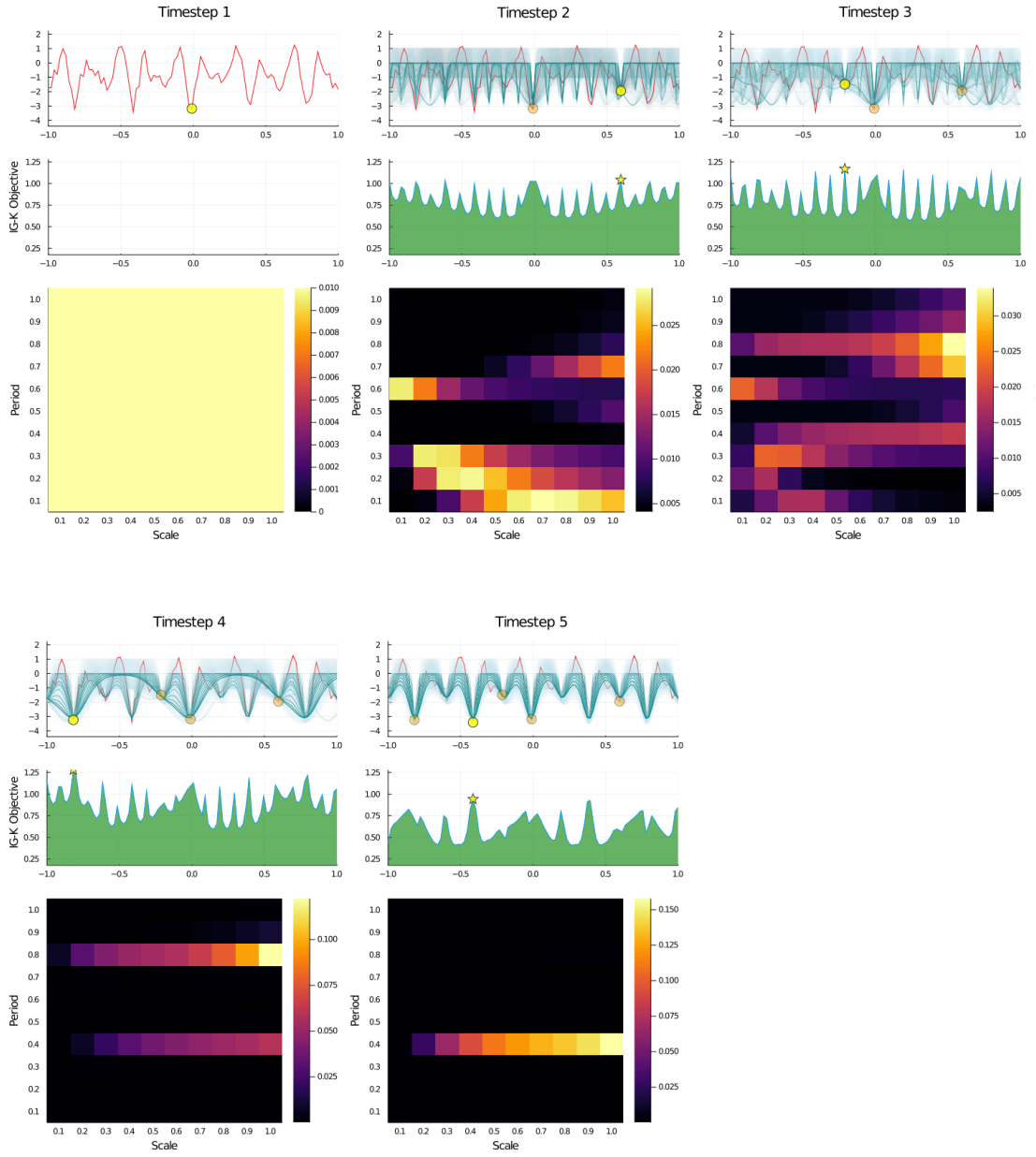
To support claims made in 4.2.2 and 4.2.3, and to further examine the dynamics of the IG-K and IG-P objectives, we present visualizations of active learning with both objectives on dataset drawn from a periodic GP kernel with period, scale, and noise parameters of 0.4, 0.6, and 0.1 respectively. For active learning, the model class is restricted to periodic kernels only, and the grid consists of 100 periodic GP models where the period and scale are chosen from $S_{0.1} \times S_{0.1}$ as defined in 4.1.2. Figure 4-10 presents the first 5 timesteps of active learning with the IG-K objective, and figure 4-11 presents timesteps 5, 10, and 15 of active learning with both IG-K and IG-P objectives.

In the figures, the n-th timestep is presented as three plots stacked vertically. The top plot shows the test dataset in red, the GP predictions in blue (with error ribbons), and the

first n observations as yellow circles. The middle plot shows the value of the information gain objective at the n-th timestep, visualized in green over all x, with a yellow star at the x which maximizes the objective and is selected as the n-th observation location. The bottom plot is a heatmap visualizing the model posterior over the gridded models after the nth observation. Each cell at (x, y) in the heatmap corresponds to a unique gridded model parameterized by scale x and period y, colored according to the normalized posterior probability.

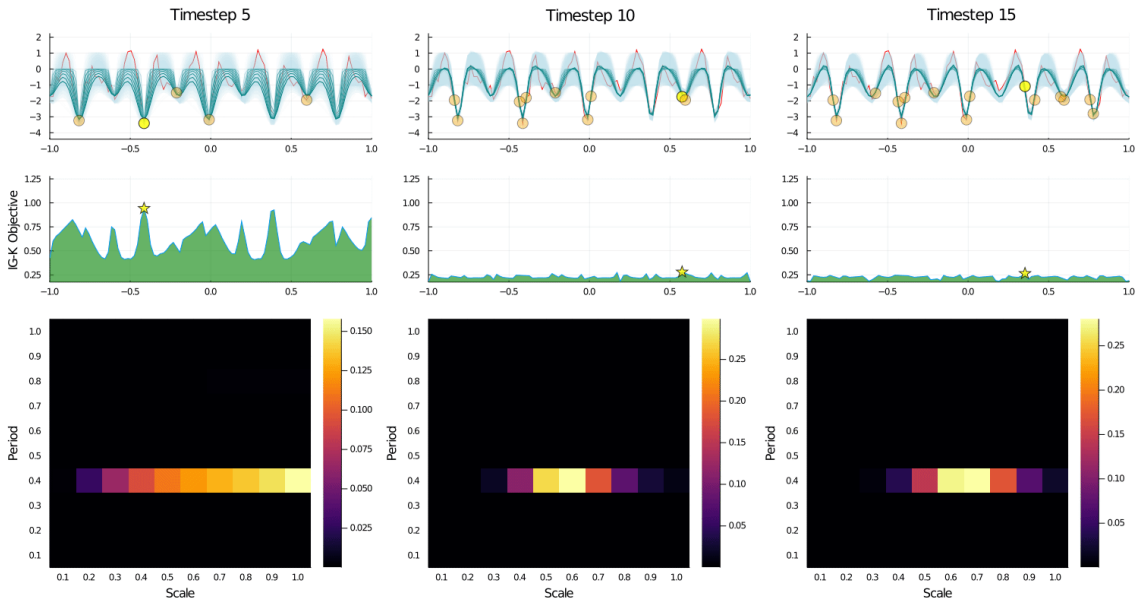
In figure 4-10, convergence of the model posterior (bottom plot, heatmap) to the correct period over timesteps demonstrates that active learning with the IG-K objective is able to learn the correct period parameter with the first 5 observations. Examination of the observations (top plot, yellow circles) selected by maximizing information gain (center plot green) suggests that active learning selects observations to infer the period according to the strategy described earlier in 4.2.2.

In figure 4-11, the model posteriors suggest that IG-K (subfigure 4-11a) is able to learn the correct period by the fifth timestep and the correct scale by the tenth timestep; in contrast, IG-P (subfigure 4-11b) is only able to learn the correct period by the fifteenth timestep. However, by the fifteenth timestep, the GP predictive posteriors (top plot, blue) for active learning with IG-P predict the unknown data more accurately than those of IG-K, as demonstrated by the fit of the blue curves to the red curve. The GP predictions for active learning with IG-K do not appear to improve much between the tenth and fifteenth timestep, since it repeatedly selects observations exactly one period apart, which reveal little information about the shape of the unknown function between observations. This exemplifies the phenomenon discussed earlier in 4.2.3 where a data selection strategy suited for inferring the correct model parameters does not select informative data for providing accurate predictions.

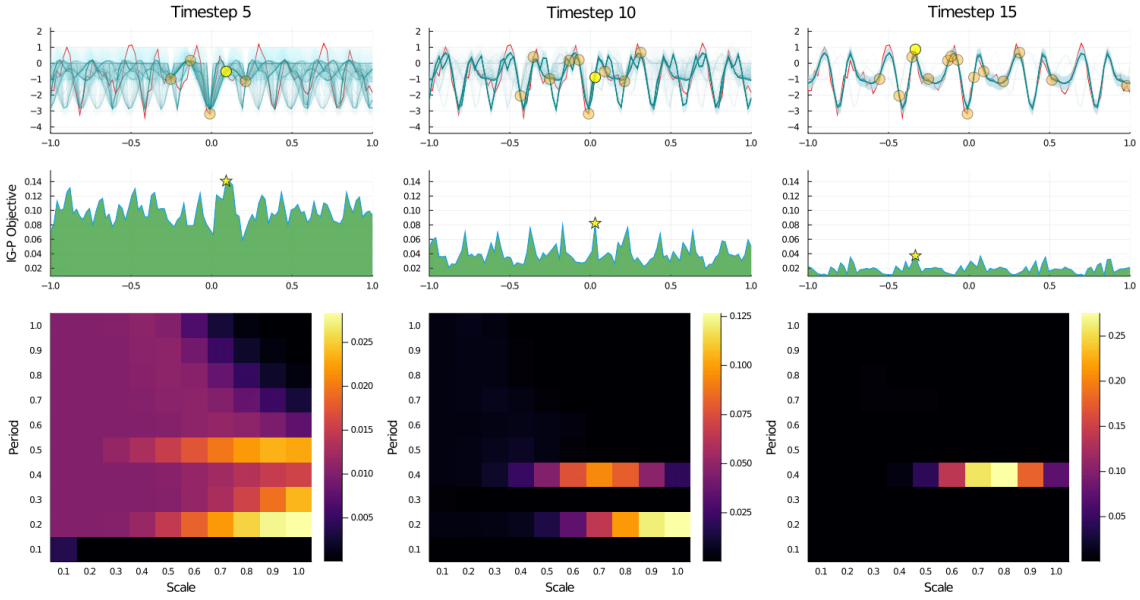


(a) Active learning on periodic data using the IG-K Objective, Timesteps 1-5

Figure 4-10: *Inferring period using IG-K.* **Timestep 1:** observation 1 is fixed at the center of the dataset; **Timestep 2:** observations 1 and 2 are located 0.6 apart and have similar y-values, informing the model posterior (grid) to converge towards periods 0.1, 0.2, 0.3, 0.6 (factors of 0.6) as discussed in 4.2.2; **Timestep 4:** observations 1 and 4 are located 0.8 apart and have nearly identical y-values,informing the model posterior to converge to periods of 0.4, 0.8 (factors of 0.8); **Timestep 5:** observations 1, 4, and 5 are located 0.4 apart and have nearly identical y-values, which is more likely with a period of 0.4, so the model posterior converges to period 0.4, the correct period.



(a) Active learning on periodic data using the IG-K Objective, Timesteps 5,10,15



(b) Active learning on periodic data using the IG-P Objective, Timesteps 5,10,15

Figure 4-11: *Predicting periodic data: IG-K vs. IG-P.* **Timestep 5:** the model posteriors indicate IG-K learns the correct period (see figure 4-10), but IG-P does not; **Timestep 10:** IG-K observation 10 is extremely close to a previous observation, informing the model posterior to converge towards the correct scale (0.6) as discussed in 4.2.2; **Timestep 15:** the model posteriors indicate that both IG-K and IG-P have converged to the correct period, but IG-P's GP predictions (blue on top plot) predict the unknown data (red) better, since IG-K's selected observations are sub-optimal for prediction as discussed in 4.2.3.

. . .

The results of the grid approximation active learning experiments presented in this chapter empirically validate that active learning with the IG-K objective reduces uncertainty over model parameters and learns the parameters better than active learning with the IG-P objective. Further analysis of the experimental results reveal that:

1. The degree to which IG-K is better able to infer the correct model parameters varies depending on the GP parameter's effect on the generated data.

2. Knowledge of the correct model parameters does not necessarily result in more accurate predictions of the data. Even with the correct model, sufficient data is necessary in order to make accurate predictions.

3. An observation selection strategy suited for inferring the correct model parameters may not be favorable for providing accurate predictions.

While grid approximation removes the confound of inference approximation to give us clear intuition about how IG-K and IG-P work on restricted model classes of structures, it does not scale to real-world applications where the structure is unknown. The next chapter presents experiments validating inference of our SMC Bayesian active learning algorithm and testing SMC active learning with IG-K and IG-P objectives on real-world data.

Chapter 5

Validating Sample-Based Approximation of Kernel Information Gain (IG-K) Objective

The experiments presented in the previous chapter test Bayesian active learning by gridding over kernel parameters for restricted model classes of structures. In practice, Bayesian active structure learning may not always perform well because of (i) the quality of the inference approximation; (ii) the quality of the approximation to the information gain objective; or (iii) negative interactions between the dataset, the model class, and the information gain objective. Grid approximation removes the confound of inference quality (i), but does not scale to the general setting where there are too many possible structures to grid over, including many cases of real-world data.

In this general setting, we measure inference quality by using simulation-based calibration (SBC) [19], which allows us to determine whether our inference procedure produces sound posterior estimates of the model distribution $P(\theta)$ over the entire (unrestricted) model class. Section 5.1 presents the SBC procedure (5.1.1), rank-statistic (5.1.2), and results (5.1.3) validating the posterior inference quality of our underlying SMC algorithm. We further evaluate the performance of our inference algorithm on a real-world dataset in Section 5.2, which presents Bayesian active learning experiments with IG-P and IG-K on a dataset of airline passenger volumes.

5.1 SBC for SMC Active Learning

5.1.1 SBC Algorithm

SBC is well-established sampling-based method for validating an inference algorithm that only requires the ability to samples from the model $P(\theta)$ in question, simulated data $\tilde{y}$ given a particular model θ, and posterior samples θ_i given the simulated data $\tilde{y}$ [19]. Below we state the procedure for SBC, specialized for the context of validating our SMC-based inference algorithm (Algorithm 2):

Algorithm 5 SBC for Bayesian Active Structure Learning with GPs

1. Sample a GP $\tilde{\theta} \sim P(\theta)$ from the covariance prior (Section 2.1.2).
2. Sample data $\tilde{y} \sim P(y|\tilde{\theta}; x)$ at the sequence of locations x.
3. Perform SMC inference with $\tilde{y}$ (Algorithm 2 where the observation locations are fixed to x) to obtain a posterior estimate $\hat{P}(\theta|\tilde{y})$
4. Sample L models $\{\theta_1, ..., \theta_L\} \sim \hat{P}(\theta|\tilde{y})$ and calculate the rank statistic as follows, where the procedure to compare two kernels $\theta_1 < \theta_2$ is defined in Section 5.1.2.

$$r(\{\theta_1, ..., \theta_L\}, \tilde{\theta}) = \sum_{l=1}^{L} \mathbb{1}[\theta_l < \tilde{\theta}] \in [0, L]$$

5. Repeat steps 1-4 N times to obtain N rank statistics.
6. Plot the N rank statistics as a histogram and compare it to the discrete uniform distribution that would arise if the inference is correct, as proven by Theorem 1 in [19].

The observation locations x in step 2 are a fixed sequence of locations $x_1, ..., x_n$ for a sampling budget of n, so the inference procedure in step 3 does not involve active selection (i.e. IG-K, IG-P). The proof of the correctness of SBC relies on the intuition that the *data averaged posterior* recovers the prior distribution as shown below [19].

$$P(\theta) = \int P(\theta|\tilde{y}; x) \cdot P(\tilde{y}|\tilde{\theta}; x) \cdot P(\tilde{\theta})\, d\tilde{y} d\tilde{\theta}$$

With SMC active learning, however, the observation locations x are not fixed; we select n observations and their locations $(\tilde{x}_1, \tilde{y}_1), ..., (\tilde{x}_n, \tilde{y}_n)$ sequentially using the IG-K or IG-P objective (Algorithm 2). The first observation location $\tilde{x}_1$ is fixed, and future $\tilde{x}$ locations are selected conditioned on previous $\tilde{y}$ values. We can test SMC active learning with SBC by treating future $\tilde{x}$ locations as random functions of previous observations:

$$\tilde{x}_i \sim P(\tilde{x}_i|\tilde{y}_{1:i-1}, \tilde{x}_{1:i-1}, \theta)$$

While this may seem like a significant departure from the standard setting, SBC in fact remains valid because the averaging the posterior over *both* the data $\tilde{y}$ and the locations $\tilde{x}$ recovers the prior. First we note that using active learning to repeatedly choose locations $\tilde{x}_i$ and observations $\tilde{y}_i$ leads to the following decomposition of the joint distribution:

$$P(\tilde{y}_{1:n}, \tilde{x}_{1:n}|\tilde{\theta}) = P(\tilde{y}_1, \tilde{x}_1|\tilde{\theta}) \prod_{i=2}^{n} P(\tilde{y}_i|\tilde{\theta}, \tilde{y}_{1:i-1}, \tilde{x}_{1:i}) \cdot P(\tilde{x}_i|\tilde{y}_{1:i-1}, \tilde{x}_{1:i-1})$$

This means that averaging the posterior $P(\theta|\tilde{y}_{1:n}; \tilde{x}_{1:n})$ over an active learning process that sequentially chooses locations and observations $(\tilde{x}_1, \tilde{y}_1), ..., (\tilde{x}_n, \tilde{y}_n)$ simply means averaging over the joint distribution $P(\tilde{y}_{1:n}, \tilde{x}_{1:n}|\tilde{\theta})$, which clearly returns the prior $P(\theta)$:

$$P(\theta) = \int P(\theta|\tilde{y}_{1:n}, \tilde{x}_{1:n}) \cdot \left(P(\tilde{y}_1, \tilde{x}_1|\tilde{\theta}) \prod_{i=2}^{n} P(\tilde{y}_i|\tilde{\theta}, \tilde{y}_{1:i-1}, \tilde{x}_{1:i}) \cdot P(\tilde{x}_i|\tilde{y}_{1:i-1}, \tilde{x}_{1:i-1}) \right) \cdot P(\tilde{\theta})\, d\tilde{y} d\tilde{x} d\tilde{\theta}$$

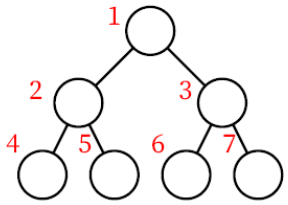
$$P(\theta) = \int P(\theta|\tilde{y}_{1:n}, \tilde{x}_{1:n}) \cdot P(\tilde{y}_{1:n}, \tilde{x}_{1:n}|\tilde{\theta}) \cdot P(\tilde{\theta})\, d\tilde{y} d\tilde{x} d\tilde{\theta}$$

5.1.2 SBC Rank Statistic for GP Kernels

In order to calculate the rank statistic in step 4 of Algorithm 5, we need an ordering on kernels θ_1, θ_2 to be able to evaluate $\theta_1 < \theta_2$. We define a kernel ordering procedure loosely inspired by ideas in [15] which maps the structure of covariance kernels to numeric arrays for comparison. To map kernels to arrays, we first define a linear ordering on the nodes of full binary trees and an ordering on the kernel types, as shown in Figure 5-1.

1	Constant (`C`)
2	Linear (`LIN`)
3	Squared Exponential (`SE`)
4	Periodic (`PER`)
5	Plus (+)
6	Times (×)

(a) Ordering on kernel types



(b) Ordering on nodes of full binary tree

Figure 5-1

We observe that the structure of any compositional covariance kernel can be represented as a binary tree as shown in the left of Figure 5-2. To make kernels easier to compare, we convert the structure tree of each kernel to a numeric array using the previously defined kernel ordering, as shown in the right of Figure 5-2. Nodes of the full binary tree that do

not appear in the covariance tree, e.g. nodes 4 and 5 in the example, are denoted by 0 in the array representation. Since the + and × operators are commutative, there are potentially many functionally equivalent covariance structures with different tree representations, e.g. swapping nodes 6 and 7 in the example produces a different tree, but a functionally equivalent kernel. To ensure both trees reduce to the same array representation, we preprocess trees to make sure left and right children are in ascending kernel order before conversion.

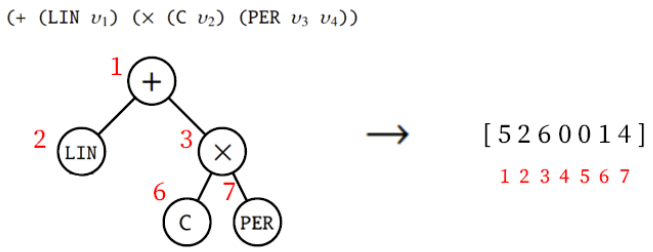


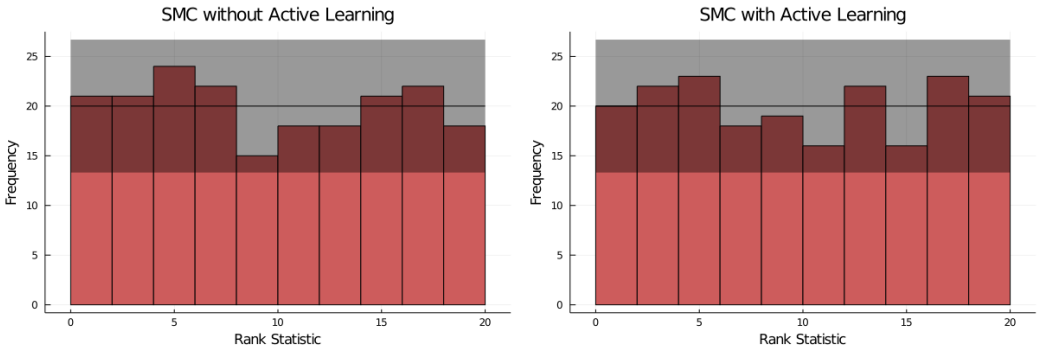
Figure 5-2: We convert compositional covariance kernels into array form by imagining kernel structure as binary trees. **Left**: The structure of the covariance kernel specified in the DSL syntax can be represented as the binary tree shown. **Right**: The covariance tree is represented as an array where the array index indicates the position in the tree, and the value (1-6) represents the kernel type according to the previous ordering.

To compare two kernels, we first convert them to numeric structure arrays as described. Then, we compare array length (tree depth). If the arrays have equal length, we compare the arrays lexicographically. If the arrays are identical (kernels have the same structure), we check the kernel parameters in tree order (e.g. $\upsilon_1, \upsilon_2, \upsilon_3, \upsilon_4$ in the example), which will almost surely be unique as they are continuous uniform random variables.

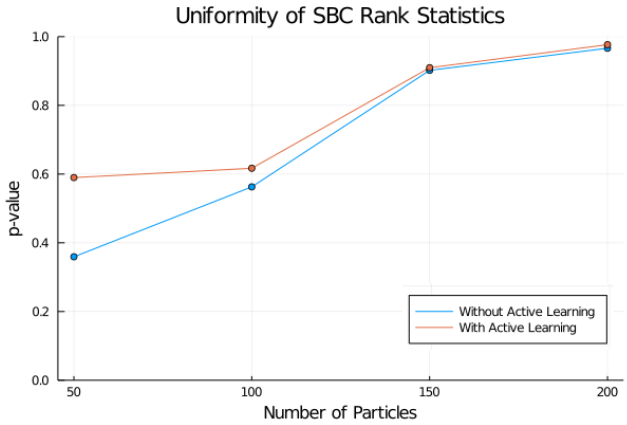
5.1.3 SBC Results

We ran SBC according to Algorithm 5 to validate SMC inference (Algorithm 2) with and without active learning (specifically IG-K), as discussed in 5.1.1. We use the following parameters as defined in Section 5.1.1 and Section 3.1, and we present the SBC rank statistics results in Figure 5-3.

- **SBC Algorithm:** $N = 200$ iterations, $L = 19$ posterior samples per iteration.
- **SMC Algorithm:** Number of particles $k = 50, 100, 150, 200$; resampling threshold $c = 0.5$; sampling budget (i.e. number of observations $\tilde{y}_i$) $T = 15$. We use the covariance structure, parameter, and noise priors given in Section 2.1.2.
- **Rejuvenation MCMC Procedure:** $\alpha = 3$ Gaussian drift kernel repetitions with standard deviation $\sigma = 0.1$, truncated to $[0, \gamma]$ where $\gamma = 1.0, \beta = 10$ full repetitions.



(a) Histograms of SBC rank statistics from SMC inference with $k = 200$ particles, with and without active learning. Both histograms indicate no issues as the empirical rank statistics (red) are consistent with the variation expected of a uniform histogram (gray), as defined in [19].



(b) Plot of uniformity (measured by Pearson chi-squared test p-values) of SBC rank statistics from SMC inference with $k = 50, 100, 150$, and 200 particles. Uniformity increases with the number of inference particles, confirming that using more particles leads to higher inference quality.

Figure 5-3: **SBC Results.** Results of SBC ($N = 200, L = 19$) validating the SMC inference (Algorithm 2) with and without active learning.

The empirical rank statistics of SBC on SMC inference with 200 particles are plotted in the histograms of Figure 5-3a and show that SMC inference is sound, both with and without active learning. As an alternative to visual inspection of the histogram, we can assess uniformity of the rank statistics by using a standard hypothesis test to compute p-values of the rank statistic frequency distribution under the expected discrete uniform distribution that would arise if the inference is correct. Like [15], we use the Pearson chi-squared test to assess uniformity, plotting the p-values for the rank statistics of SBC on inference with 50, 100, 150, and 200 particles in Figure 5-3b. The plot shows that the p-value (i.e. of uniformity) increases as we increase the number of particles of SMC inference both with and without active learning. This demonstrates that using more particles leads to higher quality inference, better approximating the true posterior.

5.2 Active Learning on an Airline Dataset

To evaluate how Bayesian active learning with the IG-K and IG-P objectives might perform on real-world data, we performed experiments on the airline dataset used in Duvenaud et al. [6] and Saad et al. [14] which measures monthly totals of international airline passengers from 1948 to 1960. The dataset, plotted in Figure 5-4, exhibits both periodic and linear trends, indicating that it is best modeled by a periodic + linear GP kernel.

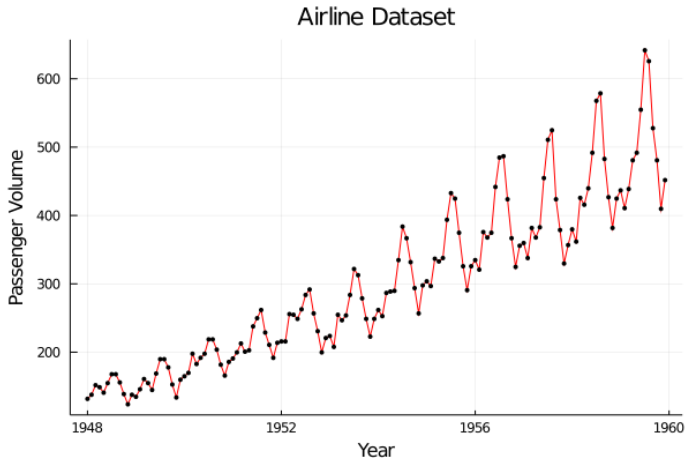


Figure 5-4: **Airline Dataset.** Dataset of monthly totals of international airline passengers; the data exhibits both periodic and linear trends.

For our experiments, we ran Bayesian active learning according to Algorithm 2 with the IG-K and IG-P objectives, using $k = 200$ particles for SMC inference and the values for rest of the parameters as defined in 5.1.3. Like in previous experiments in Chapter 4, the first observation is fixed to be the center point of the dataset, and the remaining observations are chosen via active selection. We standardize the airline dataset so it is suitable to be modeled by our GP model class with a zero-mean assumption and kernel parameter support of [0, 1]. We run active learning on the last 100 points of the dataset.

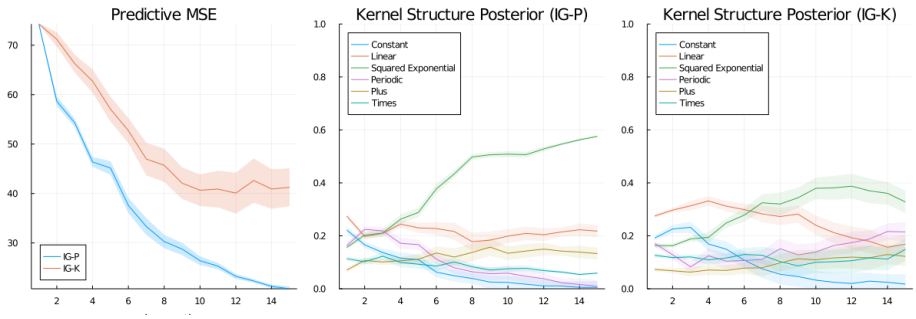
In our experiments, we record the **predictive MSE** and **structure posterior** metrics as defined in Section 4.1.1. Because there is no "ground truth" model or parameters for the airline dataset, we cannot measure the parameter MSE metric from before to assess the correctness of inferred models. Instead, we examine the posterior over kernel structures (structure distribution metric) to assess whether active learning is able to detect qualitative structure in the data (i.e. linear and periodic structure for the airline dataset).

5.2.1 Experiment Results

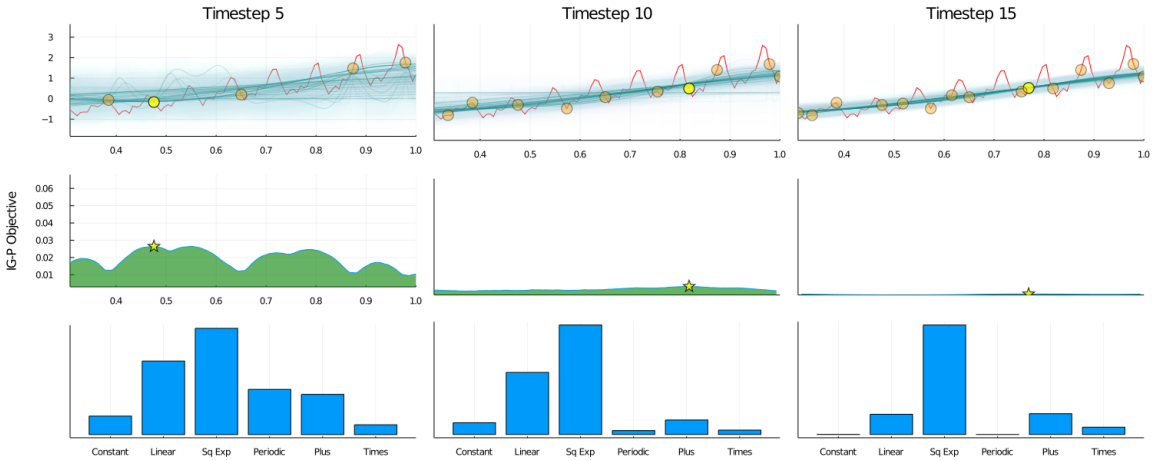
Figure 5-5a presents the predictive MSE and kernel structure posterior metrics averaged over 20 runs of active learning on the airline dataset. To provide a clearer understanding of how active learning behaves on the dataset, Figures 5-5c and 5-5b present one of the active learning runs with the IG-K and IGP objectives respectively. Active learning is visualized in a similar manner to the figures in Section 4.2.4, where each timestep is represented by three plots stacked vertically. The top plot shows the dataset, selected observations, and GP predictive posteriors; the middle plot shows the information gain objective; and the bottom plot shows the structure posterior probabilities plotted as a bar chart over the six kernel structure types.

The metrics in Figure 5-5a demonstrate that IG-P is better than IG-K at reducing predictive MSE, but a closer examination of the GP predictive posteriors in Figures 5-5c and 5-5b show that neither IG-K nor IG-P predict unknown data well. Both IG-K and IG-P fail to infer the linear and periodic trends in the data, and instead converge to a squared exponential kernel. The predictive variance ribbons in Figures 5-5c and 5-5b reveal that this is because both IG-K and IG-P interpret periodicity in the data as noise. This overestimation of noise could be due to mismatch between our noise prior and the noise level of the airline dataset. To verify this, we ran a second set of experiments where we fix the noise parameter to be smaller ($\eta = 0.01$), and present the results in figure 5-6.

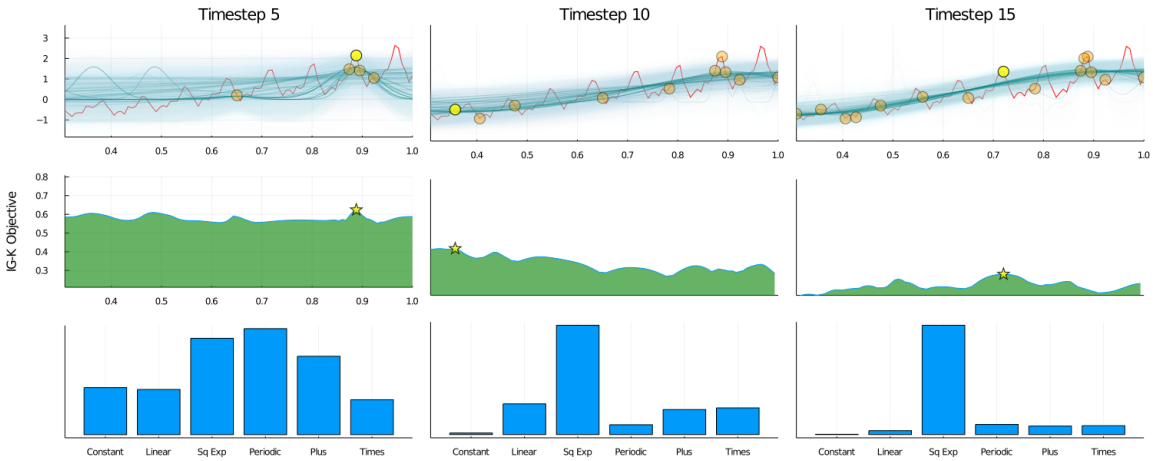
With noise $\eta = 0.01$, both IG-K and IG-P are able to infer periodic structure within the first five observations, as demonstrated by the posterior structure distributions in Figures 5-6c and 5-6b. With additional observations, IG-P is able to infer more than periodicity in the data and predict unknown data quite well. In contrast, IG-K does not predict the data well, since it selects observations located extremely close together which are uninformative for both model structure inference and data prediction. In the gridded experiments, we observed this selection strategy in cases where IG-K is inferring the periodic kernel scale or squared exponential lengthscale parameter (see 4.2.2). Thus, this failure case of IG-K on the airline dataset is likely due to initial overconfidence in the periodic structure combined with a sub-optimal selection strategy to reduce uncertainty of the period kernel scale parameter.



(a) Result metrics of active learning on airline dataset averaged over 20 runs

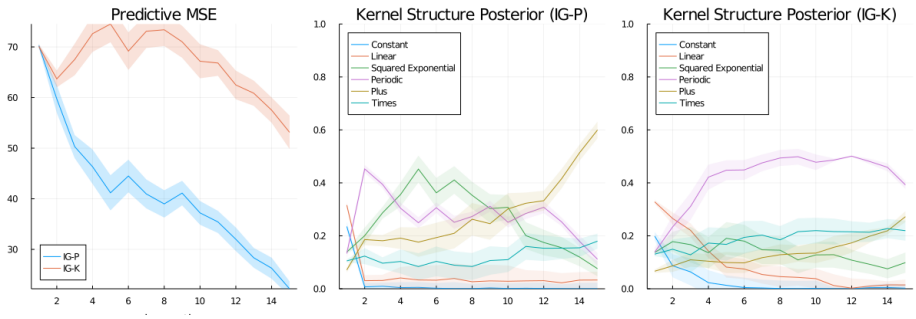


(b) Active learning on airline data using the IG-P Objective, Timesteps 5,10,15

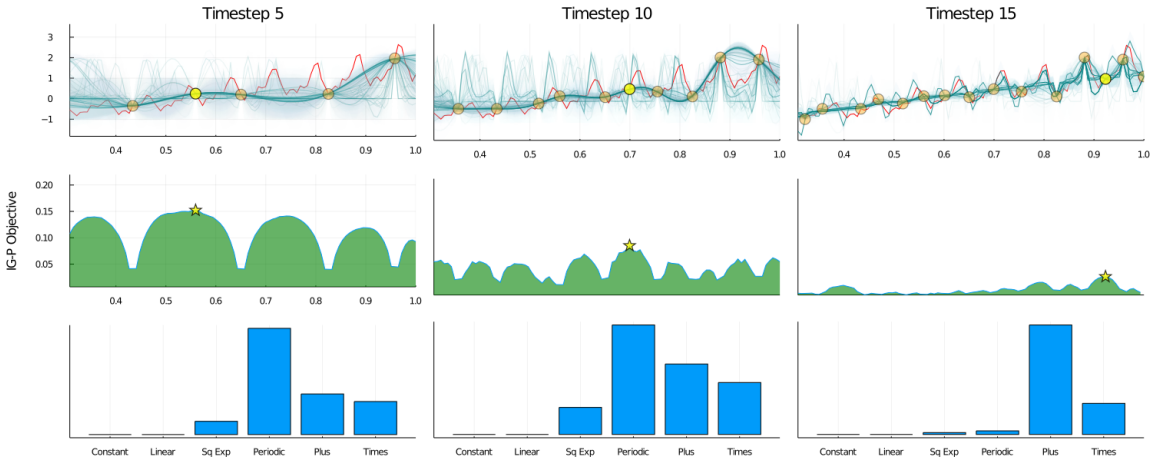


(c) Active learning on airline data using the IG-K Objective, Timesteps 5,10,15

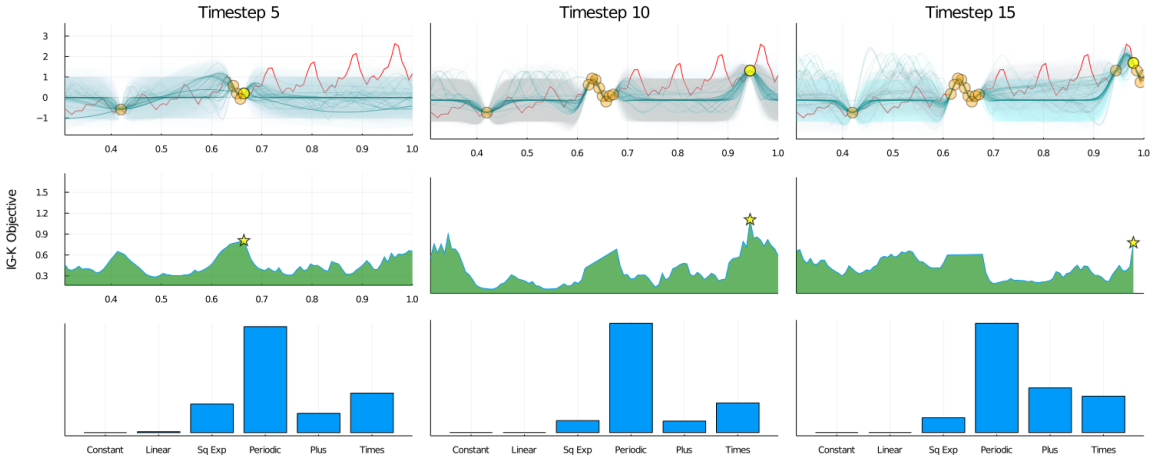
Figure 5-5: *Active learning with noise prior on the airline dataset.* The kernel structure posteriors in **(a)** show that both IG-K and IG-P fail to infer periodic structure in the data; the squared exponential structure has the highest posterior likelihood. Although IG-P reduces predictive MSE better than IG-K, examination of the predictive posteriors in **(b)** and **(c)** reveal that neither active learning algorithm predicts unknown data well; periodic structure in the data falls within the predictive variance (blue error ribbon), suggesting the predicted noise is too high.



(a) Result metrics of active learning with $\eta = 0.01$ on airline dataset averaged over 20 runs



(b) Active learning with $\eta = 0.01$ on airline data using the IG-P Objective, Timesteps 5,10,15



(c) Active learning with $\eta = 0.01$ on airline data using the IG-K Objective, Timesteps 5,10,15

Figure 5-6: *Active learning with* $\eta = 0.01$ *on the airline dataset.* The kernel structure posteriors show that both IG-K and IG-P are able to quickly infer periodic structure in the data; the periodic structure has the highest posterior likelihood at timestep 5. The structure posterior in Figure **(b)** shows that IG-P is able to infer more than periodic structure in the data, and the predictive posteriors model the data very well by the fifteenth timestep (fit of blue curves to red curve). **(c)** shows that IG-K does not predict the data well, even after 15 observations, since it selects observations in a cluster that reveals very little about the model structure or unknown data.

. . .

The results of SBC on Algorithm 2 demonstrate that our SMC inference produces sound posterior estimates of the model distribution, with and without active learning (IG-K). Additionally the SBC results demonstrate that increasing the number of particles used for SMC leads to higher quality inference. However, when run on the airline dataset, active learning with IG-K fails to recover the correct periodic + linear structure. This discrepancy between the SBC results and results on real-world data could be due to the fact that SBC tends to sample mostly simpler kernels (due to the default covariance prior). While the sampling budget of 15 observations is enough to infer simpler structures well, it may not be sufficient for accurate inference on more complex datasets such as the airline dataset with good performance.

In such cases, it may be important to ensure that the model priors are well-calibrated to the type of data being modeled. For example, after calibrating the noise level in the airline dataset experiments, active learning with IG-P is able to infer the periodic + linear within 15 observations, suggesting that the initial noise prior was miscalibrated for datasets with minimal noise. Such issues could be addressed by calibrating model priors against a wide variety of real-world datasets before applying it to new data of a similar type. Other visualization methods in Bayesian data analysis (e.g. prior and posterior predictive checks) can also serve as aids in model calibration [8].

Besides model calibration issues, the limited performance of active learning with IG-K in reducing predictive uncertainty on real world data may be ultimately due to how the IG-K objective incentivizes data selection. We discuss this tradeoff between inferring structure and predicting data, as well as potential solutions, in our final chapter.

Chapter 6

Discussion and Future Work

In this thesis, we presented a sequential Monte Carlo algorithm for Bayesian active learning with Gaussian processes to infer structure in data. We defined two objective functions: (1) kernel information gain (IG-K) to reduce uncertainty over model structure and parameters, and (2) predictive information gain (IG-P) to reduce uncertainty over predictive posteriors. We derived Monte Carlo estimators for both information gain objectives to make them tractably computable, and empirically validated that active learning with our novel IG-K objective is able to more accurately infer the structure of synthetic datasets using fewer datapoints than active learning with IG-P. We also validated the underlying SMC inference algorithm using simulation-based calibration and tested our active learning algorithm on a real-world dataset with complex structure.

Collectively, the results provide a deeper understanding of the benefits and limitations of active learning using Gaussian processes, as well as the trade-offs between minimizing structural vs. predictive uncertainty, which we summarize below:

- **Active learning performance is structure-dependent.** Optimally learning about model structure and parameters with the IG-K objective exhibits very structure-dependent behavior. This behavior can be non-intuitive and sub-optimal for prediction, such as selecting points close to each other to infer the scale of the periodic kernel (e.g. Figure 5-6).

- **Structural accuracy does not imply predictive accuracy.** Even after inferring the correct model structure, sufficient and appropriate data is necessary in order to make accurate predictions with the inferred GP model. In multiple experiments (e.g. Figure 4-11), we find that IG-K active learning is more efficient at inferring the correct model, but IG-P selects better points to improve prediction, especially once the correct model has been inferred.

- **Importance of model calibration.** In real-world applications, it is important to ensure that our model priors are reasonably calibrated to the data to avoid problems like the noise parameter overestimation in the airline dataset (Section 5.2.1).

The trade-off between improving structural accuracy and predictive accuracy suggests that in future work, a hybrid strategy of IG-K and IG-P may be more suitable than either objective alone for inferring structure in data efficiently and predicting the data well. In the hybrid strategy, we would initially use the IG-K objective to prioritize learning the correct model. Once we have enough confidence in our estimate of the model (i.e. kernel entropy is below some threshold) we would switch to using the IG-P objective to prioritize increasing predictive accuracy, assuming that the inferred model is approximately correct. However, if selected observations cause the kernel entropy to rise above the threshold, we switch back IG-K until entropy is once again below the threshold. This strategy would be helpful whenever the posterior initially converges towards an incorrect structure, as is the case in the airline dataset experiments when both IG-P and IG-K active learning initially converge to the periodic kernel instead of the periodic + linear kernel (Figure 5-6).

In addition to better navigating the trade-off between structural inference and data prediction, the dependence on structure and importance of model calibration suggest that, in any future work using GPs for active learning, we must carefully consider interactions between the data and the model class. The results of our experiments suggest that we can adjust the covariance structure, parameter, and noise priors to express expert information about or detect specific structure in the data we are trying to model. Results of experiments with particular kernel structures also help us understand for which types of structures the information gain objectives offer a more definitive advantage.

Beyond the findings presented above, the development of this thesis led us to many other important questions about GP active learning that we were not able to explore:

- **Efficient optimization procedures for maximizing information gain.** Our active learning approach enumerates over possible observation locations, which does not generalize to the case where may select observations on a continuous range. As such, enumeration could become quite expensive and slow as the number of possible locations increases. For better performance, we could use gradient descent, Bayesian optimization (as suggested in [20]), or other black-box optimization techniques to efficiently maximize the information gain objective over a continuous range of observation locations.

- **Extending to multiple input dimensions.** In this thesis, we only consider x as one-dimensional, but GPs are often used to model multidimensional phenomena (e.g.

spatio-temporal data). A key challenge for extending active learning to the multidimensonal case will be solving the efficient optimization issue above, since enumeration over multiple dimensions will become too expensive.

- **More general and realistic decision contexts.** Our experiments only consider one decision context, i.e. maximizing information gain under a fixed sample budget where samples can be chosen from anywhere within the dataset. In real-world applications, this context may not always apply. For example, in the temporal setting, we can only choose future samples, not past ones. Alternatively, sample budgets may not be fixed, but each sample may instead have some cost that we want to trade off against gaining more information, e.g. deciding whether to pay to buy more data.

We believe that these and many related research directions are ripe for future work. We hope that the methods and experiments presented in this thesis will serve as a useful resource for such work in GP active learning, and in the active learning of probabilistic programs more broadly, enabling the efficient discovery of hidden structure from rich and complex data.

Bibliography

[1] Nicolas Chopin. A sequential particle filter method for static models. *Biometrika*, 89, 01 2001.

[2] David Cohn. Neural network exploration using optimal experiment design. In J. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6. Morgan-Kaufmann, 1994.

[3] Marco Cusumano-Towner, Alexander K Lew, and Vikash K Mansinghka. Automating involutive mcmc using probabilistic and differentiable programming. *arXiv preprint arXiv:2007.09871*, 2020.

[4] Marco F. Cusumano-Towner, Feras A. Saad, Alexander K. Lew, and Vikash K. Mansinghka. Gen: A general-purpose probabilistic programming system with programmable inference. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI 2019, pages 221–236, New York, NY, USA, 2019. ACM.

[5] Arnaud Doucet, Nando De Freitas, and Neil Gordon. *Sequential Monte Carlo Methods in practice*. Springer, 2011.

[6] David Duvenaud, James Lloyd, Roger Grosse, Joshua Tenenbaum, and Ghahramani Zoubin. Structure discovery in nonparametric regression through compositional kernel search. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1166–1174, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[7] David Kristjanson Duvenaud. *Automatic Model Construction with Gaussian Processes*. PhD thesis, University of Cambridge, 2014.

[8] Jonah Gabry, Daniel Simpson, Aki Vehtari, Michael Betancourt, and Andrew Gelman. Visualization in bayesian workflow. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 182(2):389–402, 2019.

[9] Robert B. Gramacy. *Surrogates: Gaussian Process Modeling, Design and Optimization for the Applied Sciences*. Chapman Hall/CRC, Boca Raton, Florida, 2020. `http://bobby.gramacy.com/surrogates/`.

[10] Trong Nghia Hoang, Bryan Kian Hsiang Low, Patrick Jaillet, and Mohan Kankanhalli. Nonmyopic ε-bayes-optimal active learning of gaussian processes. In *International Conference on Machine Learning*, pages 739–747. PMLR, 2014.

[11] James Robert Lloyd, David Duvenaud, Roger Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. Automatic construction and natural-language description of nonparametric regression models. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, page 1242–1250. AAAI Press, 2014.

[12] David J. C. MacKay. Information-Based Objective Functions for Active Data Selection. *Neural Computation*, 4(4):590–604, 07 1992.

[13] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[14] Feras A. Saad, Marco F. Cusumano-Towner, Ulrich Schaechtle, Martin C. Rinard, and Vikash K. Mansinghka. Bayesian synthesis of probabilistic programs for automatic data modeling. *Proc. ACM Program. Lang.*, 3(POPL), January 2019.

[15] Feras A. Saad, Cameron E. Freer, Nathanael L. Ackerman, and Vikash K. Mansinghka. A family of exact goodness-of-fit tests for high-dimensional discrete distributions. In *AISTATS 2019: Proc. 22nd International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 1640–1649. PMLR, 2019.

[16] Ulrich Schaechtle, Ben Zinberg, Alexey Radul, Kostas Stathis, and Vikash K. Mansinghka. Probabilistic programming with gaussian process memoization. *CoRR*, abs/1512.05665, 2015.

[17] Sambu Seo, M. Wallat, T. Graepel, and K. Obermayer. Gaussian process regression: active data selection and test point rejection. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, volume 3, pages 241–246 vol.3, 2000.

[18] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'12, page 2951–2959, Red Hook, NY, USA, 2012. Curran Associates Inc.

[19] Sean Talts, Michael Betancourt, Daniel Simpson, Aki Vehtari, and Andrew Gelman. Validating bayesian inference algorithms with simulation-based calibration, 2020.

[20] Julius von Kügelgen, Paul K Rubenstein, Bernhard Schölkopf, and Adrian Weller. Optimal experimental design via bayesian optimization: active causal structure learning for gaussian process networks, 2019.

[21] Xiaowei Yue, Yuchen Wen, Jeffrey H. Hunt, and Jianjun Shi. Active learning for gaussian process considering uncertainties with application to shape control of composite fuselage. *IEEE Transactions on Automation Science and Engineering*, 18(1):36–46, 2021.

[22] Benjamin Zinberg. Bayesian optimization as a probabilistic meta-program. Master's thesis, Massachusetts Institute of Technology, 2015.