

Bayesian Quadrature for Gaussian Process Kernel Learning, Neural Ensemble Search, and High Dimensional Integrands



Saad Hamid

Kellogg College and Department of Engineering Science

University of Oxford, Oxford, UK

Supervised by

Prof. Michael Osborne

and

Prof. Stephen Roberts

A thesis presented for the degree of Doctor of Philosophy

Hilary Term 2023

Abstract

The central challenge of performing inference in a model is the computation of marginalisation integrals over the model’s parameters. In most cases of interest, these integrals are intractable, and evaluation of the integrand is expensive. A probabilistic approach to numerical integration offers a principled framework for allocating computation in such a setting. This is achieved by using a probabilistic surrogate to model the integrand, and selecting evaluations Bayesian Decision Theoretically. We offer Bayesian Quadrature (BQ) schemes that incorporate special structure in the model parameters for two widely-used model classes: Gaussian Processes (for which we marginalise over a broad class of stationary kernels), and Neural Networks (for which we marginalise over a large space of architectures). We further investigate the use of scalable approximations of Gaussian Processes for scaling BQ to higher dimensional (Euclidean) spaces for non-negative integrands.

For GP kernel learning, our BQ framework makes use of the maximum mean discrepancies between distributions to define a kernel over kernels that captures invariances between Spectral Mixture (SM) Kernels. Kernel samples are then selected by generalising an information-theoretic acquisition function for warped BQ.

By viewing ensembling as approximately marginalising over architectures, we bring the tools of BQ to bear upon Neural Ensemble Search. Additionally, the resulting ensembles consist of architectures weighted commensurately with their performance, unlike previous approaches that use equally weighted ensembles.

The core challenge of scaling BQ to higher dimensions is the cubic complex-

ity of GP regression. We explore the use of scalable approximations to GPs for BQ, particularly the recently proposed VISH model – a Variational GP for which the inter-domain inducing variables are projections of the modelled function onto the spherical harmonics – and Bézier GP model – defined by placing a Gaussian distribution over the control points of a Bézier curve.

Acknowledgements

I am grateful to the EPSRC (Engineering and Physical Sciences Research Council) and Kellogg College for providing the funding that allowed me to undertake this DPhil.

I would like to thank my primary supervisor Mike Osborne for his constant support throughout my DPhil. His encouragement fostered in me an interest in Probabilistic Numerics, and the academic freedom he provided allowed me to develop my confidence as a researcher. I thank also my co-supervisor, Steve Roberts, for his erudite guidance and his unwavering patience.

I owe thanks to my collaborators, Martin Jørgensen, Sebastian Schulze, Xingchen Wan, Binxin Ru, and Vincent Dutordoir, for insightful discussions and for their infectious enthusiasm. The wider BXL and MLRG also deserve my thanks for the many interesting conversations.

I am deeply grateful to my parents and my brothers for their love and support over these last few years.

Finally, I would like to thank the many friends who have been a constant source of warmth for me, and of whom Yee He and Ada Hermelink deserve special mention.

Contents

List of Figures	vii
List of Tables	ix
1 Introduction	1
2 Background	5
2.1 Hierarchical Bayesian Modelling	5
2.2 Gaussian Processes	7
2.2.1 Scalable Gaussian Process Approximations	10
2.2.2 Covariance Functions on non-Euclidean Spaces	12
2.2.3 Kernel Learning	14
2.3 Bayesian Quadrature	16
2.3.1 Warped Bayesian Quadrature	17
2.3.2 Acquisition Functions	19
2.3.3 Recombination	20
2.4 Neural Architecture Search	21
2.4.1 Neural Ensemble Search	23
3 Marginalising over Stationary Kernels for Gaussian Process Re- gression with Probabilistic Integration	24
3.1 Abstract	26
3.2 Introduction	27
3.3 Background	29
3.3.1 Gaussian Processes	29
3.3.2 Bayesian Quadrature	31
3.4 Related Work	31
3.4.1 Kernel learning	31
3.4.2 Bayesian Quadrature	33
3.4.3 Gaussian Processes on Spaces of Measures	34
3.5 Our Method: MASKERADE	34
3.5.1 The generative model	34
3.5.2 Posterior Inference	37
3.5.3 Bayesian Quadrature	37
3.5.4 Computational Complexity	42
3.6 Results	42
3.6.1 Experiment setup	43

3.6.2	Qualitative Analysis	44
3.6.3	Medium scale data sets	45
3.6.4	Large scale data sets	46
3.6.5	Ablation Study	47
3.7	Discussion	49
3.8	Supplement	50
3.8.1	Summary of WSABI	50
3.8.2	A Remark on Posterior Inference	52
3.8.3	Full Algorithm Description	54
4	Bayesian Quadrature for Neural Ensemble Search	57
4.1	Abstract	58
4.2	Introduction	59
4.3	Background	61
4.3.1	Neural Architecture Search	61
4.3.2	Bayesian Optimisation for Neural Architecture Search	64
4.3.3	Neural Ensemble Search	65
4.3.4	Bayesian Quadrature	66
4.3.5	Recombination	68
4.4	Bayesian Quadrature for Neural Ensemble Search	68
4.4.1	Building the Candidate Set	69
4.4.2	Selecting the Ensemble	71
4.5	Experiments	73
4.5.1	Ablation Study	75
4.6	Discussion and Future Work	82
4.7	Supplement	83
4.7.1	Verification of Surrogate Quality	83
4.7.2	Additional Experiments	84
5	Scalable Bayesian Quadrature with Gaussian Process Approximations	89
5.1	Abstract	90
5.2	Introduction	91
5.3	Background	92
5.3.1	Probabilistic Integration	92
5.3.2	Variational Gaussian Processes and the VISH Model	94
5.3.3	Bézier Gaussian Process	96
5.4	Methods	97
5.4.1	Using VISH for Active Learning of the Integrand	97
5.4.2	Bayesian Quadrature with the Bézier Log-Normal Process	99
5.5	Related Work	101
5.6	Experiments	102
5.7	Discussion	106
5.8	Supplement	106
5.8.1	Mixed Inducing Variables for VISH	106

6 Conclusion	111
6.1 Future Directions	112
Bibliography	114

List of Figures

3.1	Bayesian Network of our model. The number of mixture components n is drawn from a uniform prior; α parameterises a Dirichlet distribution over mixture weights w ; μ and Σ parameterise Gaussians over mixture means; ν and τ parameterise Log-Normal distributions over mixture scales. Not shown are hyperparameters of the hyper-kernel, λ and l	36
3.2	A comparison of a 5 component SM kernel with optimised hyperparameters, and two variants of MASKERADE – one that places a prior over up to 3 mixture components, and the other up to 8 mixture components – on a toy dataset drawn from a 5 component SM kernel. Each column plots attributes of the labelled model. The first row shows the posterior conditioned on the training data (for the MASKERADE models we show the moment-matched posterior). The second row shows the spectra (for positive frequencies) of the data generating kernel, and (for the 5 component SM kernel model) the optimised or (for the MASKERADE models) the sampled kernels. For the MASKERADE model, the opacity of a sampled kernel is proportional to the quadrature weight for all GP products of which that kernel is a part. (Recall that the posterior is a weighted sum of products of GP posteriors.) MASKERADE 1–3 is able to select samples near the data generating kernel, and therefore produce a posterior that generalises better than the other two models. Despite having the same number of mixture components as the data generating kernel, the optimised SM kernel sets the weights of 3 components to be very small. MASKERADE 1–8 struggles to explore its larger hyperparameter space with the same budget (500 likelihood evaluations) as MASKERADE 1–3. This can be seen by the fact that it spreads its posterior mass more evenly over a larger number of samples.	44
3.3	Posteriors for several methods on the Mauna Loa dataset (For MASKERADE and FKL we show moment matched posteriors). All methods struggle to model the linear trend, but MASKERADE is best able to extrapolate the periodic structure.	45

3.4	MASKERADE and VSSGP on the Solar Irradiance Data Set. “MAS” in the legend refers to MASKERADE, and “Conf.” refers to the (2 standard deviation) confidence interval. For clarity, we show only the posterior mean for VSSGP. Note that VSSGP is a sparse spectrum method, which is why it sometimes fails to achieve low error at the training points. Whilst VSSGP is able to approximate well the underlying periodic structure of the function, MASKERADE is able to generalise better.	46
3.5	A schematic representation of the inference procedure under our model. The left column shows the GP posterior for different Spectral Mixture Kernels on the same dataset. The centre column shows the SM kernels in their spectral domain. The right column illustrates a GP posterior on a space indexed by the spectral densities of the SM kernels. BQ can then be used to marginalise over SM kernels which may have different numbers of mixture components.	55
4.1	A diagram showing the process of building the WL Kernel’s feature vectors for NAS. At the top is the macro-skeleton which is varied by changing the cell. Defining the search space in this way allows us to model the objective function on the space of cells. Cells are represented as labelled DAGs. The $h = 0$ level features are simply histograms of the labels for each node in the graph. To compute the features at the next level, each node has its label appended with the aggregated labels of all nodes in its 1-out-neighbourhood. The $h = 1$ level features are histograms of these modified labels. Higher level features are computed by aggregating the labels for larger out-neighbourhoods.	63
4.2	A schematic representation of our proposal. The plot on the left shows a Gaussian Process modelling the likelihood over the space of architectures. The architectures to train and evaluate the likelihood for are selected by maximising a Bayesian Quadrature acquisition function, as described in Section 4.4.1. One of the algorithms described in Section 4.4.2 is then used to select the subset of architectures to include in the ensemble, along with their weights. The final prediction is then a linear combination of the predictions of each ensemble member.	69
4.3	Visualisation of the (WL) covariance matrix for the 500 unique architectures with the highest likelihoods in the search space for each dataset, sorted by (a smoothed estimate, using a GP, of the) likelihood. The colourscale varies from 1 (yellow) to 0 (blue). We observe larger blocks of architectures within the top 500 that covary strongly for CIFAR-100 than for ImageNet16-120, which implies that the modes of the architecture likelihood surface are wider for CIFAR-100. This suggests that a more exploratory strategy will do better on ImageNet16-120, and a more exploitative strategy for CIFAR-100. . . .	75

List of Tables

3.1	Test set RMSE for the Solar Irradiance Data Set.	45
3.2	Posterior log likelihood of and RMSE on the test set for UCI MLR datasets. The P-values are based on paired Student-t tests.	47
3.3	MSE on the (normalised) test set for two UCI data sets. Results for BaNK taken from (Oliva et al. 2016).	47
3.4	Posterior log likelihood of and RMSE on the test set for FKL and MASKERADE fit to two large datasets. Both methods were given a time budget of 20 minutes for training.	48
3.5	Test set RMSE for the Airline Passenger and Mauna Loa datasets. MASKERADE-I indicates the use of our proposed information theoretic acquisition function, MASKERADE-U indicates uncertainty sampling (Gunter et al. 2014), and MASKERADE-R indicates random sampling under the prior.	48
3.6	The effect of number of Monte Carlo samples used to estimate the kernel integrals (i.e. the those described in Section 3.8.1). For a MASKERADE model that marginalises over up to 5 mixture components we randomly sample sets of 100, 500 and 1000 hyperparameters from the prior, and compute their corresponding likelihoods on the UCI Airfoil Self-Noise dataset. We then infer the model evidence with WSABI using the MASKERADE hyper-kernel. For a given number of MC samples, we repeat this five times and report the mean and SEM for the posterior mean of the model evidence. We observe that the estimate for the model evidence is not highly sensitive to the number of monte carlo samples used to compute the kernel integrals.	53
4.1	Test accuracy, expected calibration error (ECE), and log likelihood (LL) on CIFAR-100 for our proposals (NES-BQ and NES-USS) and baselines. For reference we also include the performance of the best architecture (measured by validation loss) on the test set (labelled Best Single). The numbers shown are means and standard error of the mean over 10 repeats. Where applicable, the candidate set selection method is initialised with 10 random architectures, and used to build a set of 150 architectures. We find that NES-RE performs best in terms of accuracy and LL. Particularly for larger ensembles, NES-BQ performs best in terms of ECE.	76

4.2	<p>Test accuracy, expected calibration error (ECE), and log likelihood (LL) on ImageNet16-120 for our proposals (NES-BQ and NES-USS) and baselines. For reference we also include the performance of the best architecture (measured by validation loss) on the test set (labelled Best Single). The numbers shown are means and standard error of the mean over 10 repeats. Where applicable, the candidate set selection method is initialised with 10 random architectures, and used to build a set of 150 architectures. We see that NES-USS performs best across ensemble sizes in terms of LL, and joint best with NES-RE in terms of accuracy. Particularly for larger ensembles, NES-BQ performs best in terms of ECE.</p>	77
4.3	<p>Test accuracy, expected calibration error, and log likelihood on CIFAR-100 for our candidate set selection method (US) and baselines. The numbers shown are means and standard error of the mean over 10 repeats. Each candidate set selection method is initialised with 10 random architectures, and used to build a set of 150 architectures. The ensemble is chosen and weighted using our variant of weighted stacking. We see that the RE candidate set performs best for CIFAR-100. We speculate that this is because the wide peaks of the likelihood surface for CIFAR-100 favour a more exploitative strategy.</p>	78
4.4	<p>Test accuracy, expected calibration error, and log likelihood on ImageNet16-120 for our candidate set selection method (US) and baselines. The numbers shown are means and standard error of the mean over 10 repeats. Each candidate set selection method is initialised with 10 random architectures, and used to build a set of 150 architectures. The ensemble is chosen and weighted using our variant of weighted stacking. We see that the US candidate set performs best in terms of accuracy and LL across the ensemble sizes.</p>	79
4.5	<p>Test accuracy, expected calibration error, and log likelihood on CIFAR-100 for Beam Search (BS), Weighted Stacking (WS), Posterior Recombination (PR), and Re-weighted Stacking (RS). The numbers shown are means and standard error of the mean over 10 repeats. The candidate set selection method is our method – Uncertainty Sampling with a WSABI-L surrogate – initialised with 10 random architectures, and used to build a set of 150 architectures. We see that the stacking variants consistently perform best for accuracy and LL, with RS slightly improving upon WS. For ECE, RS and WS perform well for small ensembles, but PR works best for larger ensembles.</p>	80

4.6	Test accuracy, expected calibration error, and log likelihood on ImageNet16-120 for Beam Search (BS), Weighted Stacking (WS), Posterior Recombination (PR), and Re-weighted Stacking (RS). The numbers shows are means and standard error of the mean over 10 repeats. The candidate set selection method is our method – Uncertainty Sampling with a WSABI-L surrogate – initialised with 10 random architectures, and used to build a set of 150 architectures. Again, we see that the stacking variants consistently perform best for accuracy and LL, but PR for ECE.	81
4.7	The (normalised) RMSE and NLPD of a WSABI-L surrogate and a GP surrogate on the test sets.	83
4.8	Test accuracy, expected calibration error (ECE), and log likelihood (LL) on CIFAR-10 and CIFAR-100 for NES-USS (our proposal) and NES-RE (the strongest baseline).	85
4.9	Test accuracy, expected calibration error (ECE), and log likelihood (LL) on CIFAR-10 for NES-RE (the strongest baseline), and NES-USS (our strongest proposal).	86
4.10	Test accuracy, expected calibration error (ECE), and log likelihood (LL) on CIFAR-100 for NES-RE (the strongest baseline), and NES-USS (our strongest proposal).	87
5.1	The final fractional integration error after the evaluation budget is exhausted for our proposals, VISH-PI and BLNBQ, compared against several baselines. All values shown are means and standard error of the mean over 3 repeats. We see that our proposals often compare favourably to existing BQ methods, but do not perform well compared to other baselines when given the same wall-clock time budget.	105
5.2	The runtime in seconds required for each algorithm to build up the design set for each problem. The values shown are means and standard deviations over 3 repeats.	105
5.3	The modelling performance of VISH and VISH-M, measured by the Negative Log Predictive Density (NLPD) and Root Mean Squared Error (RMSE) of a held-out test set sampled from the prior. The dimensionality of the integrand (i.e. the function being modelled) is shown under the heading “D”, the number of training points under “N”, the number of inter-domain inducing variables under “ M_s ”, and the number of pseudo-input inducing variables under “ M_i ”. The size of the test set is always a quarter of the size of the training set. The values shown are means and standard deviations over 5 repeats. Surprisingly, introducing pseudo-input inducing variables degrades the performance of the model in most cases.	109
5.4	A comparison of integration performance between VISH-PI and VISH-PI-M, which has additional pseudo-input inducing variables. The fractional error in the posterior mean for the integral is reported (in log space for BLR and GPR). The values shown are means and standard deviations over 5 repeats.	109

Chapter 1

Introduction

Recent years have seen significant improvements in the performance of machine learning models on a range of tasks from signal processing (Lim et al. 2021) to semantic segmentation of images (Ronneberger, Fischer and Brox 2015) and speech recognition (Ott et al. 2019). These impressive results have been achieved by leveraging larger datasets and more expressive models. It is reasonable, therefore, to assume that further growth in dataset sizes and model expressivity will lead to further improvements in modelling performance. However, one of the key challenges in this is performing inference, which becomes computationally more demanding commensurately with dataset size and often with model complexity. It is this challenge that is the focus of the present work.

To infer means to characterise a distribution over, and this process requires the computation of marginalisation integrals (see Bishop (2006), Chapter 1, Section 2 for an introduction). In particular, the machine learning practitioner is often interested in computing posterior distributions over the parameters θ of a model M . This can be achieved using Bayes' rule,

$$p(\theta | \mathcal{D}, M) = \frac{p(\mathcal{D} | \theta, M)p(\theta | M)}{p(\mathcal{D} | M)} \quad (1.1)$$

$$= \frac{p(\mathcal{D} | \theta, M)p(\theta | M)}{\int p(\mathcal{D} | \theta, M)p(\theta | M)d\theta}, \quad (1.2)$$

where \mathcal{D} is a dataset. The marginalisation integral required here is the one in

the denominator, referred to as the marginal likelihood or the model evidence. This quantity is often intractable, and has to be numerically approximated. The difficulty is in the fact that, for expressive models of the kinds in which modern practitioners are interested, the dimensionality of the parameters θ can be high, and the likelihood $p(\mathcal{D} \mid \theta, M)$ can be expensive to evaluate.

Inferring a distribution over a target quantity (i.e. making a prediction) using a machine learning model also requires the computation of an integral. Once the posterior over the parameters (i.e. Equation 1.1) has been characterised, the distribution over a target variable y at a predictand x is given by a marginalisation over θ .

$$p(y \mid x, \mathcal{D}, M) = \int p(y \mid x, \mathcal{D}, \theta, M)p(\theta \mid \mathcal{D}, M)d\theta. \quad (1.3)$$

This distribution is referred to as the posterior predictive. For non-probabilistic models $p(y \mid x, \mathcal{D}, \theta, M)$ can be considered a delta distribution. The computation of this integral can be challenging for the same reasons outlined previously – that θ can be high dimensional, and either term in Equation 1.3 can be expensive to evaluate.

Almost ubiquitously, the purpose of making a prediction is to make a decision. The appropriate framework for this, from a Bayesian perspective, is provided by Bayesian Decision Theory (see Murphy (2012), Chapter 5, Section 7 for an introduction). An agent requires, in addition to a probabilistic model for making predictions, a loss function over possible decisions $x \in \mathcal{X}$ and outcomes $y \in \mathcal{Y}$. A rational agent then takes the decision x_* that minimises its loss, in expectation,

$$x_* = \min_{x \in \mathcal{X}} \int_{\mathcal{Y}} \mathcal{L}(x, y)p(y \mid x, \mathcal{D}, M)dy. \quad (1.4)$$

This integral is often intractable due to the expense of characterising $p(y \mid x, \mathcal{D}, M)$, so point estimates are frequently used. Such overconfidence can lead to poor decision making, especially far from the training data where the model may not be accurate.

Bayesian Quadrature (BQ), or probabilistic integration (PI), is a probabilistic

approach to integration, and is well suited to approximating integrals for computing model evidences, characterising posterior predictive distributions, and calculating expectations of loss functions (See Chapter 2, Hennig and Osborne (2021) for an introduction). Let $Z \in \mathbb{R}$ denote the integral of a function f of some input $x \in \mathcal{X}$ with respect to π , some density over \mathcal{X} ,

$$Z = \int_{\mathcal{X}} f(x)\pi(x)dx. \quad (1.5)$$

BQ maintains a probabilistic surrogate, typically a Gaussian Process (see Rasmussen and Williams (2006) for an introduction), over the integrand f (conditioned on potentially noisy observations of f) which induces a distribution over Z . The surrogate can be used to decision theoretically select new points to evaluate the integrand (Osborne, Duvenaud et al. 2012). This process has been shown to be sample efficient, and therefore appropriate for expensive black box integrands such as those required for Bayesian machine learning. The key contributions of this thesis concern the application of BQ for model selection in two widely used model classes – Gaussian Processes and Neural Networks (see Goodfellow, Bengio and Courville (2016) for an introduction) – and extending BQ to higher dimensional spaces. These follow after the background in Chapter 2.

Chapter 3 develops a BQ scheme for marginalising over a broad class of stationary kernels for Gaussian Processes. Using Bochner’s theorem (Bochner 1959), we work with the spectral representation of Spectral Mixture (SM) kernels (Wilson and Adams 2013). By using the maximum mean discrepancy between distributions (Gretton et al. 2012), we define a kernel over kernels that captures the invariances between SM kernels. We also extend an information-theoretic acquisition function for warped BQ, and use this to select new SM kernels to evaluate. We show empirically that the proposed method achieves superior regression performance compared to state-of-the-art baselines.

Chapter 4 addresses the problem of model selection for Neural Networks. By taking the view of ensembling as approximately performing marginalisation over NN architectures we can use the tools of Bayesian Quadrature to select the set of architectures (a subset of the support of the prior over architectures) to train. Additionally, BQ can be used to set the relative weights of these architectures to form the ensemble. We empirically demonstrate that our method outperforms state-of-the-art methods from the recent literature.

Chapter 5 investigates how to extend Bayesian Quadrature to higher dimensional (Euclidean) spaces. The key challenge in this context is that the volume of the integration domain increases exponentially in the number of dimensions. The surrogate model, typically a GP with a stationary kernel, therefore requires a very large number of data points to effectively “cover” the space. As the computational complexity of GP regression scales cubically in the number of observations, this becomes infeasible. We therefore investigate the use of scalable approximations for GP regression, in particular the recently proposed VISH (Dutordoir, Durrande and Hensman 2020) and Bézier GP (Jørgensen and Osborne 2022) models, in the context of BQ. We empirically compare our proposals to existing approaches for computing high dimensional model evidence integrals.

Finally, Chapter 6 concludes and suggests directions for future work.

Chapter 2

Background

Contents

2.1	Hierarchical Bayesian Modelling	5
2.2	Gaussian Processes	7
2.2.1	Scalable Gaussian Process Approximations	10
2.2.2	Covariance Functions on non-Euclidean Spaces	12
2.2.3	Kernel Learning	14
2.3	Bayesian Quadrature	16
2.3.1	Warped Bayesian Quadrature	17
2.3.2	Acquisition Functions	19
2.3.3	Recombination	20
2.4	Neural Architecture Search	21
2.4.1	Neural Ensemble Search	23

2.1 Hierarchical Bayesian Modelling

The task of specifying a model is referred to as the model selection problem (see Murphy (2022), Chapter 2, Section 2 for a discussion). Almost always, models are specified hierarchically so that one must select a model from a model class $M \in \mathcal{M}$, and the parameters of the model $\theta \in \Theta \mid M$. The Bayesian approach to model selection performs inference at each “level” in the hierarchy.

At the parameter-level, we begin with Bayes rule, as in Equation (1.1), repeated here for clarity:

$$p(\theta \mid \mathcal{D}, M) = \frac{p(\mathcal{D} \mid \theta, M)p(\theta \mid M)}{p(\mathcal{D} \mid M)}.$$

Then, at the model-level

$$p(M | \mathcal{D}) = \frac{p(\mathcal{D} | M)p(M)}{p(\mathcal{D})}, \quad (2.1)$$

where

$$p(\mathcal{D}) = \sum_{m \in \mathcal{M}} p(\mathcal{D} | M)p(M). \quad (2.2)$$

Note that the likelihood at the model-level $p(\mathcal{D} | M)$ is the marginal likelihood at the parameter-level. As $p(\mathcal{D} | M)$ can be expensive to compute, characterising $p(M | \mathcal{D})$ is also often challenging.

It is common to replace the prior $p(\theta | M)$ with a delta distribution $\delta(\theta_*)$ at the maximiser of the likelihood (the maximum likelihood estimate or MLE)

$$\theta_* | M = \operatorname{argmax}_{\theta \in \Theta | \mathcal{M}} p(\mathcal{D} | \theta, M) \quad (2.3)$$

or the maximiser of the numerator in Equation (1.1) (the maximum a posteriori or MAP estimate)

$$\theta_* | M = \operatorname{argmax}_{\theta \in \Theta | \mathcal{M}} p(\mathcal{D} | \theta, M)p(\theta | M). \quad (2.4)$$

A similar approach can be taken at the model-level.

Note that, if using MLE or MAP estimates at multiple levels, models are not appropriately penalised for excessive modelling capacity. This creates a higher risk of overfitting – the modelling of statistical artefacts that are present in the training data, but not in the underlying function that is to be modelled. Therefore, it is common to partition the dataset into a training set $\mathcal{D}_{\text{train}}$ and a validation set \mathcal{D}_{val} . $p(\theta | M)$ is then replaced by $\delta(\theta_*)$ where

$$\theta_* | M = \operatorname{argmax}_{\theta \in \Theta | \mathcal{M}} p(\mathcal{D}_{\text{train}} | \theta, M)p(\theta | M), \quad (2.5)$$

and $p(M)$ by $\delta(M_*)$ where

$$\begin{aligned} M_* &= \operatorname{argmax}_M p(\mathcal{D}_{\text{val}} | M)p(M) \\ &= \operatorname{argmax}_M p(\mathcal{D}_{\text{val}} | \theta_*, M)p(M). \end{aligned} \quad (2.6)$$

By using a separate validation set at the model-level the likelihood is approximated by $p(\mathcal{D}_{\text{val}} | \theta_*, M)$ which is a better proxy for generalisation error than $p(\mathcal{D}_{\text{train}} | \theta_*, M)$ (Rasmussen and Williams 2006).

Whilst MLE/MAP estimation is cheaper computationally, it increases the risk of overfitting by ignoring the uncertainty in the parameters/model class. It is possible that the likelihood (or product of likelihood and prior) may not have a unique maximiser, in which case this risk is exacerbated. Additionally, the estimate will perform poorly if the true posterior distribution is skewed (MacKay 1999). Therefore, the preferred approach is to marginalise over both the parameters and the model class

$$p(y | x, \mathcal{D}) = \frac{1}{p(\mathcal{D})} \sum_{m \in \mathcal{M}} p(M) \int_{\Theta} p(y | x, \mathcal{D}, \theta, M) p(\theta | \mathcal{D}, M) p(M | \mathcal{D}) d\theta. \quad (2.7)$$

Chapters 3 and 4 will focus on developing methods to achieve this for two popular model classes – Gaussian Processes and Neural Networks.

2.2 Gaussian Processes

A stochastic process is a collection of random variables, and a Gaussian Process (GP) is such a process for which all subsets of the variables are jointly normally distributed (Rasmussen and Williams 2006). They are non-parametric models, meaning that the complexity of the model grows commensurately with the size of the data. This property is desirable in machine learning models as it provides robustness against overfitting – the tendency for models with excessive capacity compared to the dataset size to model statistical artefacts that are present in the sample but not characteristic

of the underlying relationship.

There are two common perspectives of GP regression: a “weight-space” view and a “function-space” view. The function-space view regards a GP as a distribution over functions, defined by a mean function m that specifies the mean at each point in the index set $x \in \mathcal{X}$, and a covariance function (also referred to as a kernel) k that defines the covariance between two variables in the process in terms of their index set locations

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')), \quad (2.8a)$$

$$m(x) = \mathbb{E}[f(x)], \quad (2.8b)$$

$$k(x, x') = \text{Cov}(f(x), f(x')). \quad (2.8c)$$

A covariance function must produce a valid covariance matrix (also referred to as a Gram Matrix) – one that is positive semi-definite. The most frequently used kernels are stationary kernels, which specify the covariance between two points in terms of the Euclidean distance between them. A commonly used covariance function is the Radial Basis Function or Squared Exponential,

$$k(x, x') = \sigma_f \exp\left(-\frac{(x - x')^2}{2l^2}\right). \quad (2.9)$$

The signal variance σ_f and the lengthscale l are hyperparameters which need to be learnt. Other frequently used covariance functions include the Rational Quadratic, Matérn, and Periodic Covariance Functions (see e.g. Rasmussen and Williams (2006) for details). The choice of covariance function specifies the class of functions that the GP can model, and this choice significantly affects the resulting posteriors (Duvenaud et al. 2013). Therefore, much attention has been devoted to learning kernels from data for GPs (Benton et al. 2019; Duvenaud et al. 2013; Remes, Heinonen and Kaski 2017; Samo and Roberts 2015).

A GP prior can be conditioned on observations by exploiting the joint normality

of all the variables in the process

$$\begin{bmatrix} \mathbf{f} \\ f \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} m_{\mathbf{X}} \\ m_x \end{bmatrix}, \begin{bmatrix} k_{\mathbf{X}\mathbf{X}} & k_{\mathbf{X}x} \\ k_{x\mathbf{X}} & k_{xx} \end{bmatrix} \right), \quad (2.10)$$

where $\mathbf{f} \in \mathbb{R}^n$ are the observations of the function at the set $\mathbf{X} \subset \mathcal{X}$, and n is the dataset size. $m_{\mathbf{X}} \in \mathbb{R}^n$ is the mean function evaluated at the data locations, and $k_{\mathbf{X}\mathbf{X}}$ indicates the $\mathbb{R}^{n \times n}$ covariance matrix formed by evaluating the kernel between all pairs of points in \mathbf{X} . $k_{\mathbf{X}x}$ is the real-valued $n \times 1$ vector of covariances between the data points and the query point x , and $k_{x\mathbf{X}}$ is its transpose. Gaussian identities then provide the posterior distribution

$$f \mid \mathbf{f}, \mathbf{X} \sim \mathcal{GP}(\hat{m}_{\mathbf{f}}(x), \hat{k}_{\mathbf{f}}(x, x')), \quad (2.11a)$$

$$\hat{m}_{\mathbf{f}}(x) = k_{x\mathbf{X}} k_{\mathbf{X}\mathbf{X}}^{-1} (\mathbf{f} - m_{\mathbf{X}}) + m(x), \quad (2.11b)$$

$$\hat{k}_{\mathbf{f}}(x, x') = k(x, x') - k_{x\mathbf{X}} k_{\mathbf{X}\mathbf{X}}^{-1} k_{\mathbf{X}x}. \quad (2.11c)$$

It is often the case that only noisy observations of the function are available. Under an I.I.D. noise model $p(y \mid f) = \mathcal{N}(f, \sigma_n^2)$, the posterior over f becomes

$$f \mid \mathbf{y}, \mathbf{X} \sim \mathcal{GP}(\hat{m}_{\mathbf{y}}(x), \hat{k}_{\mathbf{y}}(x, x')), \quad (2.12a)$$

$$\hat{m}_{\mathbf{y}}(x) = k_{x,\mathbf{X}} (k_{\mathbf{X}\mathbf{X}} + \sigma_n^2 \mathbf{I})^{-1} (\mathbf{y} - m_{\mathbf{X}}) + m(x), \quad (2.12b)$$

$$\hat{k}_{\mathbf{y}}(x, x') = k(x, x') - k_{x,\mathbf{X}} (k_{\mathbf{X}\mathbf{X}} + \sigma_n^2 \mathbf{I})^{-1} k_{\mathbf{X}x}. \quad (2.12c)$$

where \mathbf{I} is an $n \times n$ identity matrix.

The prior mean and covariance functions are usually parameterised by a set of hyperparameters $\theta = \{\theta_m, \theta_k\}$. These hyperparameters can be marginalised out (Svensson, Dahlin and Schön 2015), or set to their MAP estimates. The key quantity required for either approach is the marginal likelihood – the probability of the

observations having marginalised out the function f – the log of which is given by

$$\begin{aligned} \log p(\mathbf{y}|\mathbf{X}, \theta) &= -\frac{1}{2}(\mathbf{y} - m_{\mathbf{X}}(\theta_m))^T (k_{\mathbf{X}\mathbf{X}}(\theta_k) + \sigma_n^2 \mathbf{I})^{-1} (\mathbf{y} - m_{\mathbf{X}}(\theta_m)) \\ &\quad - \frac{1}{2} \log |k_{\mathbf{X}\mathbf{X}}(\theta_k) + \sigma_n^2 \mathbf{I}| - \frac{n}{2} \log 2\pi. \end{aligned} \quad (2.13)$$

2.2.1 Scalable Gaussian Process Approximations

The computational cost of exact Gaussian Process regression with standard kernels is dominated by the cost of inverting the kernel matrix $k_{\mathbf{X}\mathbf{X}}$, which is cubic in the size of the training set. This presents a challenge for scaling Gaussian Processes to high number of data points. Typically, a stationary covariance function, which depends only on the distance between data points is used. In high dimensional spaces, the volume increases exponentially in the number of dimensions, so stationary kernels require more data to “cover” the space. As a result, the intractability of inference in a GP with a large number of data points limits their use to relatively low-dimensional problems.

One approach to the scaling of Gaussian Processes, that we will encounter in Chapter 5, is to perform Variational Inference in these models (Hensman, Fusi and Lawrence 2013; Matthews 2016; Titsias 2009). Our exposition follows Leibfried et al. (2021) and we denote by u a set of “inducing variables” that are joint normally distributed with the latent function f

$$\begin{bmatrix} f \\ u \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} m_f \\ m_u \end{bmatrix}, \begin{bmatrix} k_{ff} & k_{fu} \\ k_{uf} & k_{uu} \end{bmatrix} \right). \quad (2.14)$$

The conditional distribution $p(f | u)$ takes the same form as Equation (2.11a). We now place a variational distribution over u which takes the form of a Gaussian whose moments are referred to as the variational parameters,

$$q(u) = \mathcal{N}(\mu_u, \Sigma_u). \quad (2.15)$$

Using standard Gaussian identities, we can then marginalise out u

$$\begin{aligned} q(f) &= \int p(f | u)q(u)du \\ &= \mathcal{N}(k_{\cdot u}k_{uu}^{-1}(\mu_u - m_u) + m_{\cdot}, k_{\cdot\cdot} - k_{\cdot u}k_{uu}^{-1}(k_{uu} - \Sigma_{uu})k_{uu}^{-1}k_{u\cdot}). \end{aligned} \quad (2.16)$$

Importantly, the moments of the variational posterior $q(f)$ only require the inversion of an $m \times m$ matrix, k_{uu} , where m is the number of inducing variables, rather than the inversion of an $n \times n$ matrix. As the cost of matrix inversion is cubic in the size of the matrix, this is a significant computational saving. It remains to tune the variational posterior so that it matches the true posterior $p(f | \mathcal{D})$ as closely as possible. To achieve this, we perform Variational Inference (Blei, Kucukelbir and McAuliffe 2018), which seeks to minimise the Kullback-Leibler (KL) divergence between the two distributions,

$$\begin{aligned} \text{KL}(q(f) || p(f | \mathcal{D})) &= \mathbb{E}_{q(f)}[\log q(f)] - \mathbb{E}_{q(f)}[\log p(f | \mathcal{D})] \\ &= \mathbb{E}_{q(f)}[\log q(f)] - \mathbb{E}_{q(f)}[\log p(f, \mathcal{D})] + \log p(\mathcal{D}) \\ &= \log p(\mathcal{D}) - \text{ELBO}(q). \end{aligned} \quad (2.17)$$

$$(2.18)$$

where ELBO is an acronym for Evidence Lower Bound, so called as it lower bounds the log model evidence $\log p(\mathcal{D})$ since the KL divergence must be non-negative. As $\log p(\mathcal{D})$ is unavailable (and expensive to compute), instead of minimising the KL divergence directly, the ELBO is maximised.

$$\begin{aligned} \text{ELBO}(q) &= \mathbb{E}_{q(f)}[\log p(f, \mathcal{D})] - \mathbb{E}_{q(f)}[\log q(f)] \\ &= \mathbb{E}_{q(f)}[\log p(\mathcal{D} | f)] + \mathbb{E}_{q(f)}[\log p(f)] - \mathbb{E}_{q(f)}[\log q(f)] \\ &= \mathbb{E}_{q(f)}[\log p(\mathcal{D} | f)] - \text{KL}(q(f) || p(f)) \\ &= \mathbb{E}_{q(f(X))} \left[\log p(y | f(X)) \right] - \text{KL}(q(u) || p(u)). \end{aligned} \quad (2.19)$$

When the likelihood $p(y | f(X))$ is Gaussian, the ELBO can be computed analytically.

2.2.2 Covariance Functions on non-Euclidean Spaces

Often, the most natural representation of members of the set over which regression is to be performed, \mathcal{X} , is not simply as vectors in Euclidean space. Recall that GP regression on an index set requires the definition of a kernel $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ that gives rise to positive semi-definite matrices. This has motivated much work in the definition of kernels over esoteric spaces.

Riemannian manifolds are surfaces for which, at each point, there is a tangent space which is endowed with an inner product, and so a norm. The manifold must be smooth, meaning that the inner product varies smoothly across the manifold. This allows for the definition of certain geometric notions, including the length of curves. Euclidean space is a specific case of a Riemannian manifold. The shortest curve between two points on a Riemannian manifold is called the geodesic. Feragen, Lauze and Hauberg (2015) consider kernels based on geodesic distances, $d(x, y)$, of the form

$$k(x, y) = \exp(-\lambda d(x, y)^q), \quad \lambda, q > 0. \quad (2.20)$$

For $q = 2$, the kernel is positive definite only when the metric space is flat, meaning that it must be possible to map all points into a Euclidean domain whilst preserving the distance between them (Jayasumana et al. 2015). For $q = 1$, the kernel is positive definite if and only if the metric is conditionally negative definite, meaning that it must give rise to matrices, \mathbf{D} , such that $\mathbf{c}^T \mathbf{D} \mathbf{c} < 0, \forall \mathbf{c} : \sum_i c_i = 0$.

Of interest, particularly in Chapter 3, is the definition of kernels over a space of distributions. To use kernels of the form in Equation 2.20 we require a distance between distributions that either produces conditionally negative definite matrices, or isometrically maps all distributions into a Euclidean domain. Recent work has

proposed the use of Sinkhorn divergences with respect to a reference measure (Bachoc, Suvorikova et al. 2019). Another possibility is the maximum mean discrepancy between distributions (Muandet et al. 2017). These are an instance of an integral probability metric (Sriperumbudur et al. 2012), which calculates distances between distributions p and q supported on some domain \mathcal{X} as

$$d(p, q) = \max_{f \in \mathcal{F}} \left(\int_{\mathcal{X}} f(x) dp(x) - \int_{\mathcal{X}} f(x) dq(x) \right) \quad (2.21)$$

where \mathcal{F} is a class of functions on \mathcal{X} . The choice of \mathcal{F} gives rise to different metrics between distributions, and when it is the unit ball of a Reproducing Kernel Hilbert Space (RKHS) (Berlinet and Thomas-Agnan 2004) the resulting metric is the maximum mean discrepancy.

In Chapter 4 we will be interested in defining kernels over a space of graphs, where a graph is defined as a tuple (V, E, l) where V is the set of nodes, E is the set of (directed) edges, and l is a collection of labels from a fixed alphabet for each node. Typically, such kernels are defined by creating feature representations ϕ , and taking an inner product in an RKHS \mathcal{H} , $k(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}$ (Ghosh et al. 2018; Kriege, Johansson and Morris 2020; Nikolentzos, Siglidis and Vazirgiannis 2021). In particular, we will be making use of the Weisfeiler-Lehman (WL) kernel (Shervashidze 2011), which generates feature vectors from a graph in the following way:

1. Compute a histogram of the labels. These are the $h = 0$ level features.
2. Relabel each node with a new label generated by aggregating the labels of the nodes in its 1-out-neighbourhood (as we are now computing the $h = 1$ level features). The h -out-neighbourhood of a node is the set of nodes that are reachable by traversing h edges starting from that node. Then compute a histogram of these aggregated labels. These are the $h = 1$ level features.
3. Repeat (2) up to a pre-specified depth $h = H$.

4. Generate the feature representation as the aggregate of the histograms for each level $h = 0$ to $h = H$.

The WL kernel has been shown to work especially well for modelling the likelihood surface over cell-based search spaces for Neural Architecture Search (Ru, Wan et al. 2021).

2.2.3 Kernel Learning

Learning covariance functions from data by composing sums and products of a standard set of kernels (Duvenaud et al. 2013) has been attempted, but more recent work focusses on learning kernels by representing them in their spectral domain (Benton et al. 2019; Gal and Turner 2015; Jang et al. 2017; Lázaro-Gredilla et al. 2010; Oliva et al. 2016; Remes, Heinonen and Kaski 2017; Samo 2017; Simpson, Lalchand and Rasmussen 2021; Wilson and Adams 2013).

Bochner’s theorem (Bochner 1959) states that all stationary covariance functions, k , are the inverse fourier transform of a spectral density, μ , so that

$$k(\mathbf{x}, \mathbf{x}') = \int_{\mathbb{R}^d} \exp(2\pi i(\mathbf{s}^T \boldsymbol{\tau})) \mu(d\mathbf{s}). \quad (2.22)$$

where τ is the Euclidean distance between the kernel’s arguments. Bochner’s theorem can be extended to include non-stationary kernels (Genton 2001; Kakihara 1985),

$$k(\mathbf{x}, \mathbf{x}') = \int_{\mathbb{R}^d \times \mathbb{R}^d} \exp(2\pi i(\mathbf{s}^T \mathbf{x} - \mathbf{s}'^T \mathbf{x}')) \mu(d\mathbf{s}, d\mathbf{s}'). \quad (2.23)$$

When the spectral density has mass only along the diagonals $\mathbf{s}_i = \mathbf{s}'_i$, Equation (2.23) simplifies to Equation (2.22) (Samo and Roberts 2015).

The constraint of positive semidefiniteness is difficult to enforce when searching over covariance functions defined over the data domain. One is often restricted to specifying a covariance function (such as an RBF or Rational Quadratic) and learning it’s hyperparameters. Arbitrary distributions, and therefore arbitrary cov-

ariance functions, can be constructed in the spectral domain, however. This enables a broader class of covariance functions to be explored during learning.

Using the spectral density representation is also advantageous when dealing with large datasets, as it enables efficient approximation of the kernel matrix. Recall that kernels compute inner products between feature vectors

$$k(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle \approx \mathbf{z}(\mathbf{x})^T \mathbf{z}(\mathbf{x}'), \quad (2.24)$$

where we choose \mathbf{z} to be feature vectors of sines and cosines. This means that the kernel can be approximated with a Monte Carlo sum by sampling from the spectral density (Rahimi and Recht 2008). If the number of samples, m , is less than the number of data points, n , we can define $\mathbf{A} = \mathbf{Z}\mathbf{Z}^T$ where \mathbf{Z} is the $m \times n$ matrix of feature vectors for each data point. The posterior process can then be computed without using the kernel trick, so that the cost is $\mathcal{O}(m^3)$ rather than $\mathcal{O}(n^3)$. Its moments are given by

$$\mathbb{E}[y_*] = \phi(\mathbf{x}_*)^T \mathbf{A}^{-1} \mathbf{Z} \mathbf{y} \quad \text{and} \quad (2.25a)$$

$$\text{Var}[y_*] = \phi(\mathbf{x}_*)^T \mathbf{A}^{-1} \phi(\mathbf{x}_*). \quad (2.25b)$$

The locations of the samples can be optimised using the log marginal likelihood (Lázaro-Gredilla et al. 2010). This approach often overfits; an issue that can be alleviated by using a variational approximation (Gal and Turner 2015) to account for the uncertainty in the spectral locations.

Of particular interest in Chapter 3 of this work is the Spectral Mixture Kernel (Wilson and Adams 2013) which is characterised by a Gaussian Mixture Model in the spectral domain. The inverse fourier transform can be analytically computed as

$$k(\boldsymbol{\tau}) = \sum_{q=1}^Q w_q \cos(2\pi \boldsymbol{\tau}^T \boldsymbol{\mu}_q) \prod_{p=1}^P \exp(-2\pi^2 \tau_p^2 v_p^{(q)}) \quad (2.26)$$

for a symmetric mixture of Gaussians,

$$\mu(\mathbf{s}) = \sum_{q=1}^Q w_q (\mathcal{N}(\mathbf{s}; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q) + \mathcal{N}(-\mathbf{s}; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)), \quad (2.27)$$

where $\boldsymbol{\Sigma}_q$ is a diagonal covariance, $\text{diag}(v_1^{(q)}, \dots, v_P^{(q)})$. Extensions to mixtures of Laplacians (Samo and Roberts 2015) have also been proposed. Oliva et al. (2016) and Samo (2017) use MCMC sampling to approximately marginalise over such covariance functions.

Recently attempts have been made to learn spectral densities as Gaussian Processes (Benton et al. 2019; Remes, Heinonen and Kaski 2017) by designing appropriate sampling schemes, and enforcing positivism by placing the spectral GP over the log of the spectrum of the covariance function of the data GP.

2.3 Bayesian Quadrature

Bayesian Quadrature (BQ) (Minka 2000; O’Hagan 1991, 1992; Rasmussen and Ghahramani 2003), or Probabilistic Integration (PI), is a probabilistic approach to the numerical approximation of integrals. A probabilistic surrogate, often a GP, is used to model the integrand. This induces a belief over the value of the integral. This approach is advantageous in particular for black-box integrands that are expensive to evaluate because the design set – the set of observation of the integrand – can be selected in a decision-theoretic way.

Defining the quantity of interest,

$$Z = \int_{\mathcal{X}} f(x) d\pi(x), \quad (2.28)$$

the expectation of a function f with respect to some distribution π over some domain $x \in \mathcal{X}$, we can condition a GP on observations of f . Since integration is an affine

transform, the joint distribution between Z and f is a multivariate Gaussian

$$\begin{bmatrix} \mathbf{y} \\ Z \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} m_{\mathbf{X}} \\ \int m(x) d\pi(x) \end{bmatrix}, \begin{bmatrix} k_{\mathbf{X}\mathbf{X}} + \sigma_n I & \int k(\mathbf{X}, x) d\pi(x) \\ \int k(x, \mathbf{X}) d\pi(x) & \int \int k(x, x') d\pi(x) d\pi(x') \end{bmatrix} \right), \quad (2.29)$$

The moments of the marginal distribution over Z are then given by

$$\mathbb{E}_{p(Z|\mathcal{D})}[Z] = \int k(x, \mathbf{X}) d\pi(x) (k_{\mathbf{X}\mathbf{X}} + \sigma_n I)^{-1} (\mathbf{y} - m_{\mathbf{X}}) + \int m(x) d\pi(x) \quad (2.30)$$

and

$$\begin{aligned} \text{Var}_{p(Z|\mathcal{D})}[Z] &= \int \int k(\mathbf{x}, \mathbf{x}') \pi(\mathbf{x}) \pi(\mathbf{x}') d\mathbf{x} d\mathbf{x}' \\ &\quad - \int \mathbf{K}(\mathbf{x}, \mathbf{X}) \pi(\mathbf{x}) d\mathbf{x} \mathbf{K}_{\mathbf{X}}^{-1} \int \mathbf{K}(\mathbf{X}, \mathbf{x}') \pi(\mathbf{x}') d\mathbf{x}'. \end{aligned}$$

From Equations (2.30) and (2.31) we see that it is necessary to compute

$$\int k(x, x') d\pi(x) dx; \quad \text{and} \quad (2.31)$$

$$\int \int k(x, x') d\pi(x) d\pi(x'). \quad (2.32)$$

Briol, Oates, Girolami, Osborne and Sejdinovic (2015) provide analytical expressions for several combinations of kernel and prior. We note that, if the combination of kernel and prior does not admit a closed-form integral, approximating via Monte Carlo can still be a better strategy than MC approximation of Z due to the cost of evaluating f .

2.3.1 Warped Bayesian Quadrature

Frequently, it is known a-priori that the integrand is non-negative. This is the case when f is a likelihood function, as in Equation (1.1), or a predictive distribution, as in Equation (1.3). Recent work has focussed on incorporating this prior knowledge.

Chai and Garnett (2019) outline the general framework which has emerged:

1. Define $g(x) = \gamma^{-1}(f(x))$, where γ maps from \mathbb{R} to \mathbb{R}^+ .
2. Place a GP prior on g .
3. Use the posterior over g to approximate, with a GP, a belief over f .
4. Query the integrand at additional points using the belief over f to build up a set of observations, \mathcal{D} .
5. Compute $p(Z | \mathcal{D})$.

The square-root $g(x) = \sqrt{2(f(x) - \beta)}$ (where β is a small positive scalar) (Gunter et al. 2014) and $\log g(x) = \log(f(x))$ (Chai and Garnett 2019; Osborne, Duvenaud et al. 2012) transforms have received attention in the literature.

For the square-root transform, the true posterior over f is a non-central χ^2 distribution. Gunter et al. (2014) investigate both a moment-matched and a linearisation approximation. Empirically, they show that a linearisation approximation performs better. Denoting the posterior over $g \sim \mathcal{GP}(m_g(x), k_g(x, x'))$ we take a first order Taylor expansion around the posterior mean $m_g(x)$. This allows us to approximate f as a linear transform of g ,

$$f(x) \simeq \beta - \frac{1}{2}m_g(x)^2 + g(x)m_g(x). \quad (2.33)$$

The GP approximation over f is then given by

$$p(f(x) | \mathcal{D}) \approx \mathcal{GP}(\mu(x), \Sigma(x, x')), \quad (2.34a)$$

$$\mu(x) = \beta + \frac{1}{2}m_g(x)^2, \quad (2.34b)$$

$$\Sigma(x, x') = m_g(x)k_g(x, x')m_g(x'). \quad (2.34c)$$

This model is referred to in the literature as the WSABI model.

For the log transform, (Chai and Garnett 2019) showed that moment matching outperforms linearisation. In this case, the true posterior over f is a Log-Normal distribution. Moment matching yields a GP

$$p(f(x) \mid \mathcal{D}) \approx \mathcal{GP}(\mu(x), \Sigma(x, x')), \quad (2.35a)$$

$$\mu(x) = \exp(m_{\mathbf{g}}(x) + 1/2k_{\mathbf{g}}(x, x)), \quad (2.35b)$$

$$\Sigma(x, x') = m_{\mathbf{g}}(x) \left(\exp(k_{\mathbf{g}}(x, x')) - 1 \right) m_{\mathbf{g}}(x'). \quad (2.35c)$$

This model is referred to in the literature at the MMLT model.

2.3.2 Acquisition Functions

Maintaining a probabilistic surrogate over the integrand allows for a decision-theoretic approach to the selection of new evaluations. Such an approach requires the maximisation of an expected utility (or, equivalently, the minimisation of an expected loss)

$$\begin{aligned} x_* &= \arg \max_{x \in \mathcal{X}} \int u(x, f) p(f \mid x, \mathcal{D}) df \\ &= \arg \max_{x \in \mathcal{X}} \alpha(x) \end{aligned} \quad (2.36)$$

where α is referred to as an acquisition function.

The most commonly used acquisition function for Bayesian Quadrature is uncertainty sampling, which corresponds to the variance of the integrand

$$\text{Var}[f(x)\pi(x)] = \pi(x)^2 \text{Var}[f(x)]. \quad (2.37)$$

Note that this acquisition function does not depend directly on the observed integrand values – only their locations – in the case when the surrogate is an ordinary GP. (There is an indirect dependence, however, due to the optimisation of the sur-

rogate’s hyperparameters.) For the warped Bayesian Quadrature variants, however, the marginal variance of the approximate posterior depends on both the observed integrand values and locations. This encodes an explicit exploration-exploitation trade off that is justifiable for non-negative integrands. As the majority of the mass is under the peaks, exploitation corresponds to sampling where the integrand is high. Exploration corresponds to sampling where the posterior variance is high.

Of interest in Chapter 3 will also be the mutual information acquisition function suggested by Gessner, Gonzalez and Mahsereci (2019) for ordinary GP surrogates

$$I[Z; y \mid x, \mathcal{D}] = H[Z \mid \mathcal{D}] + H[y \mid x, \mathcal{D}] - H[Z, y \mid x, \mathcal{D}]. \quad (2.38)$$

2.3.3 Recombination

Bayesian Quadrature with an ordinary GP surrogate (with zero mean) takes the form of a classical quadrature rule – the mean estimate for the integral is a weighted sum of integrand evaluations

$$\mathbb{E}_{p(Z|\mathcal{D})}[Z] = \sum_{i=1}^N w_i y_i, \quad (2.39)$$

where w_i are the elements of the vector

$$w = \int k(x, X) d\pi(x) (k_{XX} + \sigma_n I)^{-1}, \quad (2.40)$$

and y_i the elements of \mathbf{y} , the observations at the set $\{x_n\}_{n=i}^N$. If the number of evaluations N is large, we may be interested in reducing it, as will be the case in Chapter 4.

Carathéodory’s theorem states that if a point x lies within the convex hull of a set $P \subset \mathbb{R}^d$, then x can be written as a convex combination of at most $d+1$ extremal points in P (Carathéodory 1911). When the weights are convex – meaning that $w_i \geq 0$ for all i and $\sum_{i=1}^N w_i = 1$ (which is not the case for BQ) – Carathéodory’s theorem

can be used to reduce the support of a measure, and achieve the objective of reducing the number of evaluations. In the literature, this is referred to as recombination (Tchernychova 2015). Provided $M - 1$ “test” functions $\{\phi_t(\cdot)\}_{t=1}^{M-1}$, it is possible to find a subset of $M < N$ points $\{x_i\}_{i=1}^M \subset \{x_i\}_{i=1}^N$ for which

$$\sum_{m=1}^M w_m \phi_t(x_m) = \sum_{n=1}^N w_n \phi_t(x_n) \quad (2.41)$$

for all ϕ_t , with $w_m \geq 0$ and $\sum_{m=1}^M w_m = 1$.

The Nyström approximation of the kernel matrix can be used to define a set of test functions (Hayakawa, Oberhauser and Lyons 2022). Let S be a subset of $M - 1$ locations at which integrand observations are available. The kernel can be approximated $\tilde{k}(x, x') = k(x, S)k(S, S)^{-1}k(S, x')$. Using an eigendecomposition $k(S, S) = U\Lambda U^T$,

$$\tilde{k}(x, x') = \sum_t^{M-1} \frac{1}{\lambda_t} (u_t^T k(S, x)) (u_t^T k(S, x')) \quad (2.42)$$

where u_i are the columns of U , and λ_i the diagonal elements of Λ . $\phi_t(\cdot) = u_t^T k(S, \cdot)$ can be used as test functions. In this way one can perform kernel quadrature whilst constraining the weights to be convex.

2.4 Neural Architecture Search

Neural Networks (NNs) have been shown to be excellent function approximators, achieving state-of-the-art performance across a range of application domains including object detection (Wang, Bochkovskiy and Liao 2022), speech recognition (Ott et al. 2019), and even modelling dynamic systems (Chen, Rubanova et al. 2018). It is well known that the architecture of a NN significantly impacts its performance, as the architecture implicitly defines the space of possible functions representable by the NN. Such a view is supported by a body of literature that seeks to incorporate

salient invariances into NN architectures (Cohen and Welling 2016; Worrall et al. 2017; Zaheer et al. 2017).

Neural Architecture Search (NAS) refers to the problem of automatically selecting the architecture of an NN (Ren et al. 2020). This problem has received significant attention in the AutoML community, as the manual design of NN architectures is time-consuming and requires specialist expertise. The key challenges are that the space of possible architectures is very large, and that training each architecture is computationally expensive.

Elsken, Metzen and Hutter (2019) outline three sub-components of a NAS pipeline: the definition of a search space; a strategy for efficiently exploring the space; and a performance estimation for evaluating architectures with the search space.

Broadly, there are two approaches to defining a search space. The first is a cell-based search space, which consists of architectures made by swapping out “cells” in a fixed macro-skeleton (Dong, Liu et al. 2021; Dong and Yang 2020). These cells are represented as directed acyclic graphs where each edge corresponds to an operation from a pre-defined set of operations. Typically, the macro-skeleton will be structured so that repeating copies of the same cell are stacked within the macro-skeleton. This structure allows for the representation of the architecture by the corresponding cell. The second is a search space defined by varying structural parameters such as kernel size, number of layers, and layer widths. Such a search space can be defined using a “slimmable” network (Yu et al. 2019) – the largest possible network is trained and all other networks in the search space are given as sub-networks or “slices”.

A number of strategies have been proposed for selecting an architecture from a NAS search space, including Differentiable Architecture Search (DARTS), Bayesian Optimisation, and evolutionary strategies. DARTS (Liu, Simonyan and Yang 2019) and its derivatives (Chen, Xie et al. 2019; Xu et al. 2020) are a search strategy for cell-based search spaces. The core idea is to take a softmax over all possible operations between nodes to define a continuous relaxation of this categorical choice.

This enables searching over the space of cells using backpropagation. Evolutionary strategies (Liu, Sun et al. 2021; Real, Aggarwal et al. 2019; Real, Moore et al. 2017) maintain a population of architectures and iteratively apply random perturbations to the cells, retaining architectures that perform better and culling those that do not. Bayesian Optimisation (Kandasamy et al. 2019; Ma, Cui and Yang 2019; Ru, Wan et al. 2021; White, Neiswanger and Savani 2020) maintains a probabilistic surrogate over the architecture likelihood (or accuracy) function, and uses this to decision-theoretically select new architectures to train.

Performance estimation is typically done by training the architecture for a fixed number of epochs on a training set, and then evaluating the loss on a separate validation set.

2.4.1 Neural Ensemble Search

Neural Ensemble Search refers to the problem of selecting an ensemble of architectures from a NAS search space, which we will consider in Chapter 4. This involves two steps – the selection of a set of candidate architectures to train, and the selection of a subset of these to include in the ensemble along with their relative weights. Zaidi et al. (2022) suggest using a regularised evolutionary strategy to build up the candidate set of architectures. The ensemble members are then chosen by Beam Search, which starts off with the most performant architecture and greedily adds the architecture (from the candidate set) that most improves the validation loss. Shu et al. (2022) instead propose approximating the posterior over architectures with a Variational distribution. The ensemble members are selected by sampling from (a continuous relaxation of) the Variational distribution, using Stein Variational Gradient Descent to select a diverse set.

Chapter 3

Marginalising over Stationary Kernels for Gaussian Process Regression with Probabilistic Integration

Contents

3.1	Abstract	26
3.2	Introduction	27
3.3	Background	29
3.3.1	Gaussian Processes	29
3.3.2	Bayesian Quadrature	31
3.4	Related Work	31
3.4.1	Kernel learning	31
3.4.2	Bayesian Quadrature	33
3.4.3	Gaussian Processes on Spaces of Measures	34
3.5	Our Method: MASKERADE	34
3.5.1	The generative model	34
3.5.2	Posterior Inference	37
3.5.3	Bayesian Quadrature	37
3.5.4	Computational Complexity	42
3.6	Results	42
3.6.1	Experiment setup	43
3.6.2	Qualitative Analysis	44
3.6.3	Medium scale data sets	45
3.6.4	Large scale data sets	46
3.6.5	Ablation Study	47
3.7	Discussion	49

3.8	Supplement	50
3.8.1	Summary of WSABI	50
3.8.2	A Remark on Posterior Inference	52
3.8.3	Full Algorithm Description	54

Preface

Gaussian Processes are an expressive class of models whose probabilistic foundations allow for data-efficient inference, and whose nonparametric nature provides some degree of robustness to overfitting. However, the choice of covariance function significantly affects how well the model generalises, as it encodes a strong inductive bias. Additionally, inference in GPs scales cubically in the number of data points, limiting their use in practice to relatively small datasets. As volume increases exponentially in the number of dimensions, larger datasets are required for regression in this setting. Limitation to small datasets therefore limits GPs also to relatively low dimensional regression problems.

Recent years have seen attention devoted to the scaling of GPs to enable inference for larger datasets. The core challenge is the inversion of the kernel matrix, which is required not just for computing posterior moments but also for calculating the marginal likelihood. Broadly, the approaches seen in the literature are:

- Approximating the inversion of the kernel matrix, often by taking an approach based on conjugate gradient decent for solving a linear system.
- Using an approximation to the kernel, such as the Random Fourier Feature approximation or the Nyström approximation.
- Performing Variational Inference by minimising the distance between the functional posterior and a parameterised approximation, the variational posterior.

With the exception of the last, these approaches are orthogonal to our proposal, and can be used in conjunction with it.

This chapter concerns the application of Bayesian Quadrature to the kernel learning problem for Gaussian Process regression. The core benefit is to allow for the inference of, and marginalisation against, a posterior over a broad class of stationary kernels. Additionally, the sample-efficient framework provided by active Bayesian Quadrature reduces the number of marginal likelihood evaluations required, which improves the scalability of Gaussian Process regression.

Bochner’s theorem states that all stationary covariance functions are the inverse Fourier Transform of a positive spectral density. As a consequence, we can define a prior over a large class of stationary covariance functions by defining a prior over positive spectral densities. This is convenient as the constraint of point-wise positivity is significantly easier to enforce than that of positive definiteness. In particular, we will use Gaussian Mixture Models in the spectral domain to define stationary covariance function, following Wilson and Adams (2013). By using the maximum mean discrepancy between distributions, we have a distance d between distributions that isometrically maps distributions into a Euclidean space, and therefore gives rise to valid covariance function of form $k(x, y) = \exp(-d(x, y)^2)$ (where x and y represent distributions). We use such a covariance function to perform Bayesian Quadrature. Additionally, we extend an information-theoretic acquisition function for use with warped Bayesian Quadrature.

3.1 Abstract

Marginalising over families of Gaussian Process kernels produces flexible model classes with well-calibrated uncertainty estimates. Existing approaches require likelihood evaluations of many kernels, rendering them prohibitively expensive for larger datasets. We propose a Bayesian Quadrature scheme to make this marginalisation more efficient and thereby more practical. Through use of the maximum mean discrepancies between distributions, we define a kernel over kernels that captures

invariances between Spectral Mixture (SM) Kernels. Kernel samples are selected by generalising an information-theoretic acquisition function for warped Bayesian Quadrature. We show that our framework achieves more accurate predictions with better calibrated uncertainty than state-of-the-art baselines, especially when given limited (wall-clock) time budgets.

3.2 Introduction

Gaussian Processes (GPs) (Rasmussen and Williams 2006) are a rich class of models, which place probability distributions directly on classes of functions. Crucially, the success of these models is tied to the choice of their kernels. Analogously to architecture design in deep learning, kernels control the expressiveness and complexity of a GP model. If the correct kernel is chosen, GPs have shown the ability to make accurate predictions based on comparatively small training data sets. Beyond that, they natively provide predictive uncertainties with little additional computation. These confidence measurements can be vital for various down-stream tasks such as decision-making in the real world.

If, however, the chosen kernel is misspecified, little probability mass will be assigned to the neighbourhood of the true function, resulting in a poor fit. Many commonly used kernels, such as the Radial Basis Function and Matérn kernels, have been shown to be universal kernels (Lugosi 2006), meaning they can approximate arbitrary functions given sufficient data. Despite this they, in practice, encode strong and frequently task-inappropriate inductive biases, significantly delaying learning progress.

To increase model flexibility and avoid such pathologies practitioners typically define parameterised *kernel families*. The selection of a particular kernel is delayed until training and made data-dependent, e.g. via a maximum likelihood estimate (MLE). A fully Bayesian approach gains further expressivity and robustness by

marginalising across the chosen family instead of settling on a single kernel instance for test-time predictions. This is particularly valuable when working with large, complex data sets that exhibit complicated interactions between data points.

Our approach aims to create a flexible regression model capable of representing arbitrary stationary kernels and efficiently learning complex functions from large data sets. We draw on several previously separate strands of research to achieve this.

The analysis of kernels in the spectral domain has been shown to be an effective tool in learning expressive kernels (Wilson and Adams 2013). Due to Bochner (1959) a link between stationary kernel functions and frequency measures can be established. The former is a general class of kernels encompassing many popular families (e.g. RBF, Matérn, periodic). The link to the latter allows the construction of posterior distributions in a principled and computationally efficient way with little user intervention and opens up clear avenues for their interpretation.

Rather than settling on a specific kernel via MLE, we conduct inference in the spectral-kernel framework by marginalising over the entire kernel family. The resulting integrals are computationally intractable and solutions can only be approximated. Previous approaches (Benton et al. 2019; Oliva et al. 2016; Simpson, Lalchand and Rasmussen 2021) based on Monte Carlo variants effectively average across likelihood evaluations of kernels randomly sampled from the posterior. Each likelihood evaluation requires an inversion of the kernel Gram matrix over all data points, making such approaches increasingly expensive as we scale to larger data sets.

To stay efficient even in large data, expensive likelihood settings, we adopt a model-based approach to integral approximation. Bayesian Quadrature (BQ) (O’Hagan 1991; Rasmussen and Ghahramani 2003) methods model the function to be integrated directly, to incorporate and exploit prior knowledge of regularities (e.g. smoothness and non-negativity of likelihood surfaces (Osborne, Duvenaud et al. 2012)). This enables more careful sample acquisition, yielding performance

superior to MCMC methods in wall-clock time for moderate-dimensional problems (Gunter et al. 2014).

Throughout this paper, we make the following contributions. Firstly, the construction of a custom *hyper*-kernel based on the maximum mean discrepancy between distributions. This kernel is cheap to evaluate, whilst producing meaningful covariances between spectral densities, regardless of their parameterisation. Secondly, a BQ framework based on this kernel to conduct computationally efficient GP inference on large data sets. Thirdly, the derivation of an information-theoretic acquisition function for the selection of new parameter evaluations. Finally, we empirically demonstrate the viability of the resulting approach on several synthetic examples and real world data sets. Our algorithm chooses informative likelihood evaluations and achieves high performance (log-likelihoods) on limited computational budgets measured in wall-clock time.

3.3 Background

3.3.1 Gaussian Processes

A Gaussian Process (GP) defines a probability distribution over the space of functions $f : \mathcal{X} \rightarrow \mathbb{R}$, such that for any finite subset $\mathcal{X}' \subset \mathcal{X}$, the vector $\{f(x)\}_{x \in \mathcal{X}'}$ is normally distributed. We denote such a distribution $f \sim \mathcal{GP}(m, k)$, where the mean $m(x) = E[f(x)]$ and kernel functions $k(x_1, x_2) = E[(f(x_1) - m(x_1))(f(x_2) - m(x_2))]$ encode our prior beliefs about the function. Analytic expressions for posterior mean and kernel of a process conditioned on a set of observations D under a Normal likelihood are readily available.¹

¹A detailed discussion of the GP framework can be found in (Rasmussen and Williams 2006).

Spectral Covariance Functions

In this work the prior kernel functions $k(x_1, x_2)$ are assumed to be stationary. With slight abuse of notation, they take the form:

$$k(x_1, x_2) = k(|x_1 - x_2|) = k(\rho), \quad (3.1)$$

such that two function values covary only depending on their separation ρ and not their location.

Bochner's Theorem (Bochner 1959) states that $k(\rho)$ is a positive-definite function on \mathbb{R} (and valid kernel) if and only if its Fourier transform, $S(\omega)$, is a positive spectral density, i.e.:

$$k(\rho) = \int e^{2\pi i \rho \omega} dS(\omega). \quad (3.2)$$

As a consequence, any parameterisation of a set of spectral measures induces a corresponding parameterisation of stationary kernels².

Kernels based on distance metrics

Distance-based kernels which often take the form $k(x_1, x_2) = \lambda^2 \exp(-\frac{d(x_1, x_2)^q}{l^2})$ (λ , and l being hyper-parameters), are valid, i.e. positive definite, for $q = 1$ if and only if the underlying metric $d(x_1, x_2)$ is conditionally negative definite (Feragen, Lauze and Hauberg 2015; Jayasumana et al. 2015), meaning that it must give rise to matrices, D , such that $c^T D c < 0, \forall c : \sum_i c_i = 0$. For $q = 2$ the underlying metric must be Hilbertian, meaning that there must be an isometry between the metric space defining d and a Hilbert space (Feragen, Lauze and Hauberg 2015).

²For $k(\rho)$ to be real, we only consider symmetric $S(\omega)$

3.3.2 Bayesian Quadrature

Bayesian inference in machine learning frequently involves computation of intractable integrals of the form:

$$Z = \int f(x)p(x)dx, \quad (3.3)$$

where $p(x)$ is a known prior density and $f(x)$ a (likelihood-)function. Bayesian quadrature is a model-based approach to approximately evaluating such integrals by modelling f as a GP. Since GPs are closed under affine transforms the posterior over Z is then Gaussian. For suitable kernel-prior combinations the quadrature weights can be evaluated analytically.

While the maintenance of the GP model requires some computational effort, it enables the generalisation of sample information across the integration domain. Consequently, samples can be chosen in a targeted fashion and BQ has been shown to be a competitive, more evaluation-efficient alternative to Monte Carlo methods, particularly in moderate-dimensional domains.

Recent work (Chai and Garnett 2019; Gunter et al. 2014; Osborne, Duvenaud et al. 2012) further increases sample efficiency by encoding model-constraints such as the positivity of likelihood functions using warped GPs. This work builds upon WSABI (Gunter et al. 2014) in particular, which places a GP prior over the square-root of f and is described in more detail in Appendix 3.8.1.

3.4 Related Work

3.4.1 Kernel learning

Early work in kernel learning focussed on searching over compositions (sums and products) of a set of basic covariance functions (Duvenaud et al. 2013). More recent work has proposed learning kernels using their spectral representations (Ambrogioni and Maris 2018; Benton et al. 2019; Gal and Turner 2015; Jang et al. 2017; Lázaro-

Gredilla et al. 2010; Oliva et al. 2016; Remes, Heinonen and Kaski 2017; Samo 2017; Samo and Roberts 2015; Teng et al. 2019; Tobar 2018; Wilson and Adams 2013). Beyond the ability to capture more complex structures, this approach also lends itself to approximate inference, reducing learning time complexity (Gal and Turner 2015; Hensman, Durrande and Solin 2018; Rahimi and Recht 2008). A model similar to ours uses a Dirichlet Process prior over the number of components in a Gaussian Mixture Model (GMM) residing in the spectral domain (Oliva et al. 2016). Simpson, Lalchand and Rasmussen (2021) use Nested Sampling to marginalise over the parameters of a Spectral Mixture kernel with a fixed number of mixture components.

Inference in spectral models reduces to the marginalisation of the likelihood function against the posterior over kernels (or, equivalently, their spectral decompositions). The resulting integrals are intractable and have to be approximated – usually through the use of an appropriate MCMC scheme, such as Gibbs sampling, RJ-MCMC (Green 1995) or elliptical slice sampling (Murray, Adams and MacKay 2010). Scaling MCMC marginalisation over the spectral kernel parameters up to large datasets is problematic. The computational cost of likelihood evaluations scales poorly in the size of the training data set and the collection of sufficiently many samples for an accurate Monte-Carlo estimate becomes prohibitively expensive. As above, BQ may be a more appropriate choice in these settings.

The recently proposed baselines we will compare against in our experiments include:

Variational Sparse Spectrum Gaussian Process Gal and Turner (2015) which performs Variational Inference over a sparse spectrum approximation.

Bayesian Nonparametric Kernel Learning Oliva et al. (2016) which learns the posterior distribution over Spectral Mixture kernels via a parameterisation of spectral densities based on GMMs and a Gibbs sampling scheme.

Functional Kernel Learning Benton et al. (2019) which places a GP prior over the kernel’s spectral density and infers a posterior over kernels using an Elliptical Slice Sampling scheme.

Various approximations to reduce the cost of likelihood evaluations have been proposed in the literature, such as PCG (Cutajar et al. 2016) and BBMM (Gardner et al. 2018) – modified conjugate gradient methods – and Random Fourier Features (Rahimi and Recht 2008). This is orthogonal to our investigation and our framework is optionally capable of leveraging both of these for scalability.

3.4.2 Bayesian Quadrature

Recent work in BQ (Chai and Garnett 2019; Gunter et al. 2014; Osborne, Duvenaud et al. 2012) has proposed the use of warped GPs (Snelson, Ghahramani and Rasmussen 2004) to incorporate a priori known model-constraints into the surrogate. This work builds upon WSABI (Gunter et al. 2014) in particular, which places a GP prior over the square-root of the integrand. The acquisition function developed in Section 3.5.3 also applies to this setting and is an extension of the information-theoretic scheme discussed in Gessner, Gonzalez and Mahsereci (2019). Osborne, Garnett et al. (2012) also proposed a BQ framework to infer ratios of integrals, which is relevant to the computation of marginalised posteriors. However, the combination with WSABI and our framework introduces further intractable integral terms, which must be approximated with Monte Carlo sampling and would prohibitively raise the cost of inference. Additionally, the work of Xi, Briol and Girolami (2018) and Chai, Ton et al. (2019) is related as it concerns the use of BQ for computing correlated integrals, and for automatic model selection.

3.4.3 Gaussian Processes on Spaces of Measures

The BQ integrand model requires the specification of a positive-definite kernel across the integration domain – the space of spectral densities. While numerous metrics and divergences have been proposed to measure differences between probability distributions, many do not give rise to valid kernels.

Recent work has used Wasserstein distances (Bachoc, Gamboa et al. 2018) to define Gaussian Processes on a space of measures. However, Wasserstein distances are only Hilbertian in 1D (Peyré and Cuturi 2019) so extensions to multidimensional distributions rely on Hilbert space embeddings of optimal transport maps to a reference distribution (Bachoc, Suvorikova et al. 2019). Furthermore, evaluating Wasserstein distances can be expensive as it involves finding the solution to an optimal transport problem. The Independence kernel based on the Sinkhorn distance (Cuturi 2013) has been shown to be a valid positive definite kernel, but the distance between identical distributions may not be zero. An elegant alternative that satisfies our desiderata are maximum mean discrepancies (MMDs), which are defined in terms of kernel mean embeddings. MMDs are Hilbertian, so are a valid metric for kernels of the form (3.6) (Muandet et al. 2017).

3.5 Our Method: MASKERADE

In the following we present a flexible, data efficient Gaussian Process framework and show how to conduct Bayesian inference within it.

3.5.1 The generative model

Without loss of generality, we assume input dimensions of the training data to be rescaled to the interval $[0, 1]$ and outputs to be normalised to have zero mean and unit variance.

By this construction, a zero mean prior for our GP model is the most logical

choice. As mentioned above, kernels are only assumed to be stationary. Following (Wilson and Adams 2013) and (Oliva et al. 2016), we parameterise spectral densities through Gaussian Mixture Models (GMMs) to induce a corresponding parameterisation of stationary kernels. In the limit of infinitely many components GMMs can approximate any spectral density to arbitrary precision. For multi-dimensional datasets we use GMMs with a diagonal covariance structure (note that this assumption is not a limitation of our framework and can be relaxed). The number of parameters of a given density is then $n(1 + 2d)$, where n is the number of mixture components, and d is the dimensionality of the dataset.

The kernel, matching a spectral density parameterised by θ , can be recovered via (3.2). To create only real-valued kernels, we reflect all mixtures at the origin. We also note, that an additional output-scale $k(0)$ is required to cover the space of (unnormalised) measures and recover arbitrary stationary kernels. Since data has unit variance, however, we omit this in the following.

Kernels defined in this way are expressive, but inferring their parameters has been shown to be challenging. This is because the likelihood surface is multi-modal and difficult to explore, and the number of parameters grows quickly with the dimensionality (Simpson, Lalchand and Rasmussen 2021).

To perform Bayesian inference and marginalise across stationary kernels, we define a hyperprior $p(\theta)$ (see next section). A graphical representation of the resulting generative model is shown in Figure 3.1.

The Hyperprior $p(\theta)$

The definition of our Bayesian model necessitates the choice of a suitable hyperprior. We see this is an advantage to the practitioner, as they are able to include existing knowledge to reduce the space of kernels needing to be explored. In the absence of such information, we motivate a set of heuristics based on general properties of the training data.

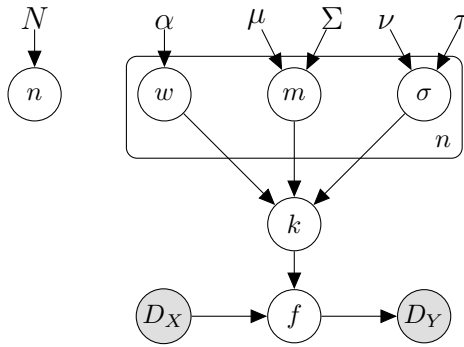


Figure 3.1: Bayesian Network of our model. The number of mixture components n is drawn from a uniform prior; α parameterises a Dirichlet distribution over mixture weights w ; μ and Σ parameterise Gaussians over mixture means; ν and τ parameterise Log-Normal distributions over mixture scales. Not shown are hyperparameters of the hyper-kernel, λ and l .

The prior over the number of mixture components n is uniform up to some maximum N , limiting the size of the overall parameter space. The n components are weighted against one another through a Dirichlet prior.

Mixture means are drawn from a normal distribution with zero mean and standard deviation $F_s/5$ so that negligible mass lies above the Nyquist frequency F_s of the data. Recall that the Nyquist frequency is the highest frequency identifiable by a data set.³

Mixture scales can be seen as approximate inverse lengthscales of the kernel. Accordingly their log-normal prior is set such that most of its mass lies between the inverse of the maximum distance between data points and the Nyquist frequency – so that $\log(1/|D_x|)$ (where $|D_x|$ is the size of the data window) is 5 standard deviations below the mean (of the normal distribution in log-space), and $\log(F_s)$ is 5 standard deviations above.

We stress that the prior outlined above is, in our view, the widest reasonable prior, and that performance is likely to be improved considerably by specifying narrower priors, if possible, especially for high dimensional datasets.

³For unevenly-spaced data we choose F_s as the Nyquist frequency of a fictitious dataset with sampling frequency equal to the inverted average distance between datapoints.

3.5.2 Posterior Inference

As we are epistemically uncertain about the kernel parameters, we seek to marginalise them out. Accordingly, the predictive distribution at some location x_* under the above model, after observing data $D = \{x_D, y_D\}$, is given by:

$$\begin{aligned}
 p(y_* | x_*, D) &= \int p(y_* | x_*, \theta, D) p(\theta | D) d\theta \\
 &= \int p(y_* | x_*, \theta, D) \frac{p(D | \theta) p(\theta)}{p(D)} d\theta \\
 &= \frac{\int p(y_* | x_*, \theta, D) p(D | \theta) p(\theta) d\theta}{\int p(D | \theta) p(\theta) d\theta},
 \end{aligned} \tag{3.4}$$

where $\theta = (n, \{w_i, m_i, \sigma_i\}_{i=1}^n)$ is the parameterisation of a spectral GMM with n components, each described by a weight w_i , mean m_i and scale σ_i . The predictive distributions $p(y_* | x_*, \theta, D)$ and likelihoods $p(D | \theta)$ for the spectral kernel corresponding to a particular θ are given by the GP framework.

Unfortunately, both integrals in (3.4) are intractable and require approximation. We note that a major cost here is the evaluation of Gaussian Process likelihoods. These involve an inversion of the kernel Gram matrix of the entire data set whose computational cost increases quickly with data set size (typically cubically). To reduce the number of likelihood evaluations required, we propose to adopt a Bayesian Quadrature approach.

3.5.3 Bayesian Quadrature

We begin by constructing GP models of the functions $p(y_* | x_*, \theta, D) p(D | \theta)$ and $p(D | \theta)$ that are to be integrated against the prior. Since the locations x^* of test points are unknown at training time, inference separates into two steps.

During training, a “hyper-dataset” \mathfrak{D} of decompositions and matching likelihood evaluations (and model evidences respectively) is compiled through active sampling (see Section 3.5.3) to shape our surrogate models and improve approximation qual-

ity. Hereby, we follow (Gunter et al. 2014) and enforce positivity of the likelihood surrogate by placing a GP prior over the square-root of the likelihood function $L(\theta) = P(D | \theta)$:

$$z = \sqrt{2(L(\theta) - \epsilon)} \sim \mathcal{GP}(0, \kappa), \quad (3.5)$$

where $\kappa(\theta_1, \theta_2)$ is a kernel between spectral decompositions (see section 3.5.3) and ϵ is a small non-negative constant.

Under this model $L(\theta)$ is not a GP, but can be approximated with one using a linear transformation of the surrogate. The posterior mean and covariance functions $\mathbf{m}_{\mathfrak{D}}(\theta)$ and $\mathfrak{K}_{\mathfrak{D}}(\theta_1, \theta_2)$ then remain functions of the kernel κ and the likelihood observations. To ensure a good fit to the actual likelihood function, hyperparameters are re-optimised after each evaluation. Since GPs are closed under continuous bounded linear transformations (Bogachev 1961), we obtain a Gaussian posterior over the model evidence – the integral in the denominator of (3.4). The mean of this Gaussian takes the form $z^T Q z$, where Q are the BQ weights – a full derivation is given in Appendix 3.8.1.

To make a prediction, we compute the integral in the numerator of (3.4) in the same fashion. $p(y_* | x_*, \theta, D)$ is a GP posterior. Since we employ the same kernel κ to measure similarity between spectral densities, and we reuse the same likelihood evaluations from the training stage, the quadrature weights are identical and only z differs, allowing for efficient computation of the desired quantities.

The Hyper-Kernel

Noting that the densities under consideration are GMMs with a limited number of components, it becomes apparent that a suitable *hyper*-kernel, κ , has to overcome two obstacles. Firstly, it needs to cope with varying numbers of parameters, or otherwise assign a predefined covariance between GMMs with differing numbers of components. Secondly, it should capture a number of invariances in the parameterisation of GMMs (e.g. reordering of components, subdivision or recombination of

components etc.). A naïve approach for the construction of such a kernel would base distances between densities on the Euclidean distance between their parameters and fails to achieve either desiderata.

Instead we construct a kernel based on the distance between the spectral densities represented by those parameters. Accordingly, we seek a conditionally-negative definite or Hilbertian metric between distributions. The maximum mean discrepancy meets our criteria, and is cheap to compute.

We obtain the following kernel:

$$\kappa(\theta_1, \theta_2) = \lambda^2 \exp\left(-\frac{d(\theta_1, \theta_2)^q}{l^2}\right), \quad (3.6)$$

where $\lambda > 0$ is an output-scale, $l > 0$ is a length-scale parameter, $q = \{1, 2\}$ and d is a maximum mean discrepancy between the spectral densities (GMMs) parameterised by θ_1 and θ_2 . The exact form of the MMD depends on the underlying kernel. Our experiments use the Energy Distance MMD (Feydy 2020), which arises from using the underlying kernel $\varkappa(x, y) = -\|x - y\|$. The resulting metric is given by

$$d(\theta_1, \theta_2) = w_1^T M_{12} w_2 - \frac{1}{2} w_1^T M_{11} w_1 - \frac{1}{2} w_2^T M_{22} w_2 \quad (3.7)$$

where w_1 and w_2 are vectors of component weights and M_{12} is a matrix of euclidean distances between all pairs of component parameters $\{(m_i, \sigma_i)\}_{i=1}^{n_1}$ and $\{(m_i, \sigma_i)\}_{i=1}^{n_2}$, but our framework allows for alternatives (e.g. the Gaussian MMD, which can also be computed analytically for GMMs). The expression is similar for (diagonal) multi-dimensional GMMs – M_{12} remains a matrix of euclidean distances between all pairs of component parameters, which are now means and standard deviations along each dimension. Note that, since the MMD is analytically computed (rather than approximated through sampling as is usual in the MMD literature), it is a valid distance between unnormalised densities. The hyper-kernel therefore remains valid when kernels with arbitrary, non-unit output-scales are considered.

The only drawback is that integration of this kernel against the hyperprior distribution is no longer possible analytically, so Monte Carlo integration is necessary. (This is because the parameters appear in the squared exponential of Equation (3.6), which needs to be integrated against Dirichlet, Gaussian and Log-Normal distributions.) Arguably, however, the integration of the kernel against the prior lends itself better to such an approximation than that of the data likelihood. Intuitively, the likelihood function will have many sharp peaks for large datasets requiring samples to be drawn at very specific (unknown) locations for an accurate estimate. The kernel function by comparison is smoother, cheaper to evaluate and the resulting integral easier to approximate via random sampling from the prior. Our empirical results justify this approach, showing that we outperform competing methods despite the requirement for sampling.

Information-theoretic point acquisition

The GP surrogate is trained based on a set of sampled spectral distributions, $\mathfrak{D} = \{(\theta_i, L_i)\}$, and the likelihood values of their corresponding kernels. Since adding a new point θ^{new} to this set requires a computationally expensive likelihood evaluation, it should be chosen to maximise its informativeness w.r.t. the quantity we care about – the predictive distribution at test locations x_* .

As we do not expect the model to have access to the test locations at training time, we choose a different acquisition criterion for the new point. Instead, we aim to improve the approximation of the denominator of (3.4) – and thereby the fit of the posterior distribution over spectral mixtures – at training time. In (Gunter et al. 2014) an acquisition function based on the uncertainty of the integrand is proposed. We extend the work of (Gessner, Gonzalez and Mahsereci 2019) and (Ru, McLeod et al. 2018) to propose an information-theoretic criterion instead.

We observe that the expected reduction in the entropy of our integral estimate

after making an observation at sample location θ^s is given by

$$\alpha(\theta_*) = H(L_* | \mathfrak{D}, \theta_*) - E_{p(Z|\mathfrak{D}, \theta_*)}[H(L_* | \mathfrak{D}, \theta_*, Z)]. \quad (3.8)$$

Here L_* is the predicted observation at location θ_* and Z is the integral in the denominator of (3.4). Within the GP framework, the entropy of the predictive distribution $H(L_* | \mathfrak{D}, \theta_*)$ is the entropy of a Normal distribution and can be calculated in closed form. The second term turns out to be the expectation of a constant and the predictive distribution conditioned on Z can once more be computed in closed form. Gessner, Gonzalez and Mahsereci (2019) discuss this acquisition function in the context of ordinary and multi-source Bayesian Quadrature. We note that it is available for warped Bayesian Quadrature. In particular, the acquisition function can be computed analytically when the linearisation approximation is used, provided that the kernel integrals are analytic. This is because the composition of the approximation and integration remains an affine transformation. Therefore, the integral of the approximate warped surrogate is jointly Gaussian distributed with the unwarped surrogate.

The acquisition function (3.8) is also valid for selecting batches of a fixed size. Importantly, our combination of hyper-kernel and acquisition function allows for the selection of (batches of) GMMs that are maximally informative about the overall evidence, $\int p(D | \theta)p(\theta)d\theta$, rather than selecting GMMs for each parameter domain (corresponding to a fixed number of components) separately. This is because the kernel is able to define covariances between GMMs with different numbers of mixture components. Incorporating such information represents an improvement over (Chai, Ton et al. 2019) whose framework, in this instance, would be naïve as it assumes that the GPs over each domain are independent.

3.5.4 Computational Complexity

During learning, each iteration requires optimising the hyperparameters of the surrogate GP and the acquisition function. The time complexity of hyperparameter optimisation is dominated by the cost of making likelihood evaluations, $\mathcal{O}(h^3)$ where h is the number of Spectral Mixture kernels evaluated thus far. Acquisition function optimisation incurs an initial cost of $\mathcal{O}(h^3 + mh^2 + m^2)$, where m is the number of Monte Carlo samples used to approximate the kernel integrals. Subsequent evaluations require $\mathcal{O}(bh^2)$ operations, where b is the batch size. The memory complexity during hyperparameter optimisation is dominated by the cost of storing the hyper-kernel evaluated between observed SM kernels $\mathcal{O}(h^2)$. During the initialisation of the acquisition function, $\mathcal{O}(m^2)$ memory is additionally required to store the hyper-kernel evaluated between the MC samples.

Once the evaluation budget is exhausted the quadrature weights can be computed in $\mathcal{O}(h^3 + mh^2 + m^2)$ time and $\mathcal{O}(h^2 + m^2)$ memory. Inference takes $\mathcal{O}(h|D|^3)$ operations, though this can be reduced significantly by disregarding terms that have low quadrature weight. The memory complexity of inference is $\mathcal{O}(h^2|D|^2)$ for a naïve implementation, and can be reduced to $\mathcal{O}(h|D|^2)$ by looping appropriately (note that this leaves the asymptotic time complexity unchanged).

3.6 Results

In the following we empirically assess the performance of our model on a variety of tasks.

3.6.1 Experiment setup

The experiments were conducted on Nvidia Titan V⁴ and Nvidia GeForce GTX 1080⁵ GPUs. We report the Root Mean Squared Errors (RMSE) of the mean function of the predictive posterior as well as the Log-Likelihoods (LL) of the predictive posterior on held out test data after training each model for a fixed training time budget. Since the cost of likelihood evaluations depends on the dataset size, so does the training budget. The numbers given indicate either mean performance or mean performance and standard deviation.

We compare our approach of MARGinalising Spectral KERnels As DENsities (MASKERADE)⁶ to models previously proposed in the literature including VSSGP (Gal and Turner 2015), BaNK (Oliva et al. 2016), and FKL (Benton et al. 2019). For VSSGP we use the VSSGP⁷ toolkit with parameters tuned to obtain the best performance on the relevant dataset. Results for BaNK are directly taken from the paper. FKL hyperparameters were set to the defaults for the `spectralgp`⁸ Python package, with the ω_{max} (maximum frequency) argument modified to match the Nyquist frequency of the dataset.

We limit the parameter space MASKERADE considers to GMMs with up to 5 components (N=5). The concentration of the Dirichlet prior over weights is set to $\alpha = 1$. The remaining priors are chosen as described in section 3.5.1. We initialise the BQ surrogate GPs with likelihood evaluations of parameters randomly chosen from the prior and thereafter acquire additional evaluations using the acquisition function. LBFGS is used to optimise both the acquisition function as well as the parameters of the hyper-kernel.

⁴Comparisons to BaNK and VSSGP, and ablation studies.

⁵Comparison to FKL.

⁶<https://github.com/saadhamidml/maskerade>

⁷<https://github.com/yaringal/VSSGP>

⁸<https://github.com/wjmaddox/spectralgp>

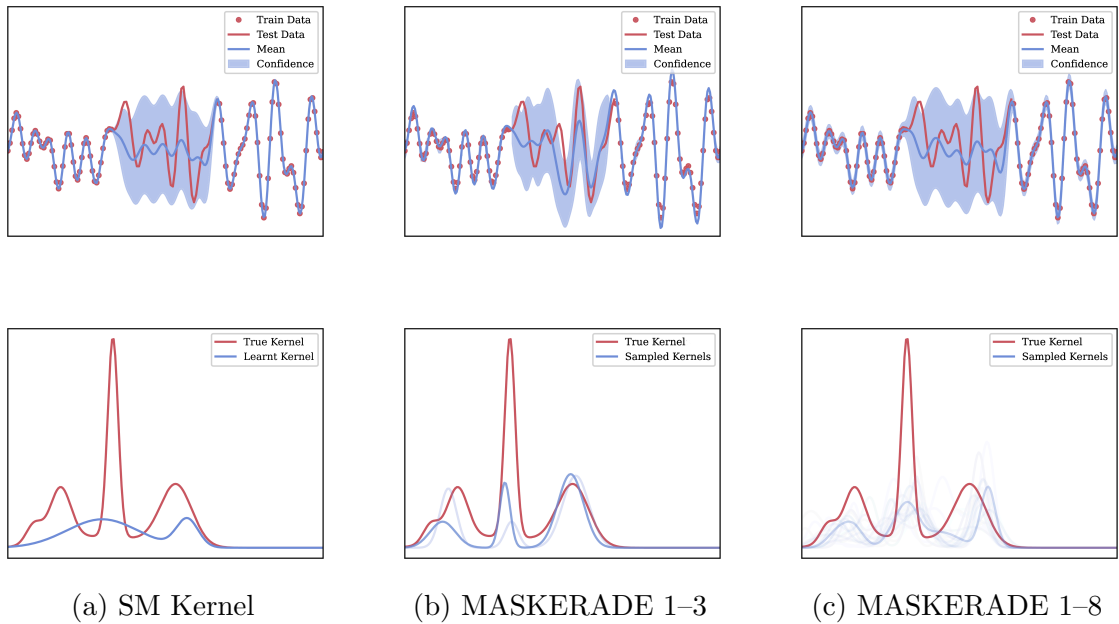


Figure 3.2: A comparison of a 5 component SM kernel with optimised hyperparameters, and two variants of MASKERADE – one that places a prior over up to 3 mixture components, and the other up to 8 mixture components – on a toy dataset drawn from a 5 component SM kernel. Each column plots attributes of the labelled model. The first row shows the posterior conditioned on the training data (for the MASKERADE models we show the moment-matched posterior). The second row shows the spectra (for positive frequencies) of the data generating kernel, and (for the 5 component SM kernel model) the optimised or (for the MASKERADE models) the sampled kernels. For the MASKERADE model, the opacity of a sampled kernel is proportional to the quadrature weight for all GP products of which that kernel is a part. (Recall that the posterior is a weighted sum of products of GP posteriors.) MASKERADE 1–3 is able to select samples near the data generating kernel, and therefore produce a posterior that generalises better than the other two models. Despite having the same number of mixture components as the data generating kernel, the optimised SM kernel sets the weights of 3 components to be very small. MASKERADE 1–8 struggles to explore its larger hyperparameter space with the same budget (500 likelihood evaluations) as MASKERADE 1–3. This can be seen by the fact that it spreads its posterior mass more evenly over a larger number of samples.

3.6.2 Qualitative Analysis

We qualitatively inspect the behaviour of MASKERADE in Figure 3.2 by examining the spectra of kernels assigned the highest weights in the posterior, and the effect of varying the number of mixture components that are marginalised over.

Next, we visually verify that our method improves upon baselines by plotting

	VSSGP	MASKERADE
RMSE	0.328	0.224

Table 3.1: Test set RMSE for the Solar Irradiance Data Set.

fits on the Mauna Loa dataset. These are shown in Figure 3.3. MASKERADE is parameterised to marginalise over 5 mixture components, with priors set as in Section 3.1.1. FKL uses the defaults in the `spectralgp` Python package, with the maximum frequency set to the Nyquist frequency of the Mauna Loa dataset. The Spectral Mixture kernel uses 5 mixtures, initialised using the empirical spectrum of the dataset.

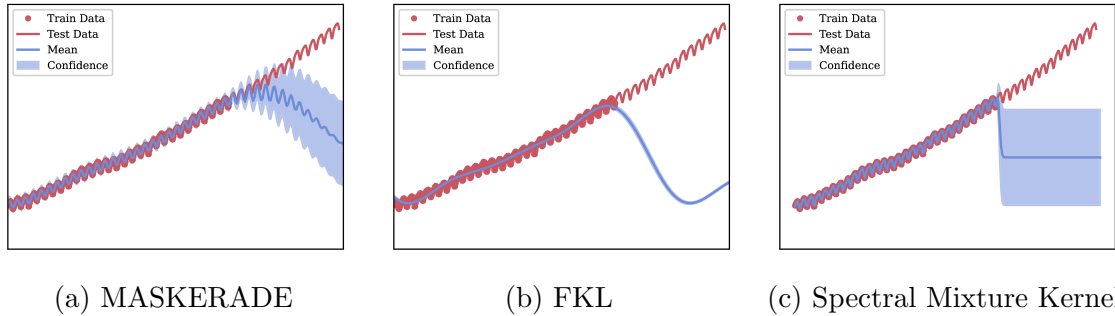


Figure 3.3: Posteriors for several methods on the Mauna Loa dataset (For MASKERADE and FKL we show moment matched posteriors). All methods struggle to model the linear trend, but MASKERADE is best able to extrapolate the periodic structure.

3.6.3 Medium scale data sets

We next fit MASKERADE to the Solar dataset (Lean 2004) to compare against VSSGP. We replicate the setup from Gal and Turner (2015), withholding 5 sets of length 20 as the test set. For VSSGP we follow Gal and Turner (2015) in choosing 50 inducing inputs, and allow LBFGS 5000 iterations to optimise the model’s parameters. Table 3.1 and Figure 3.4 show the results.

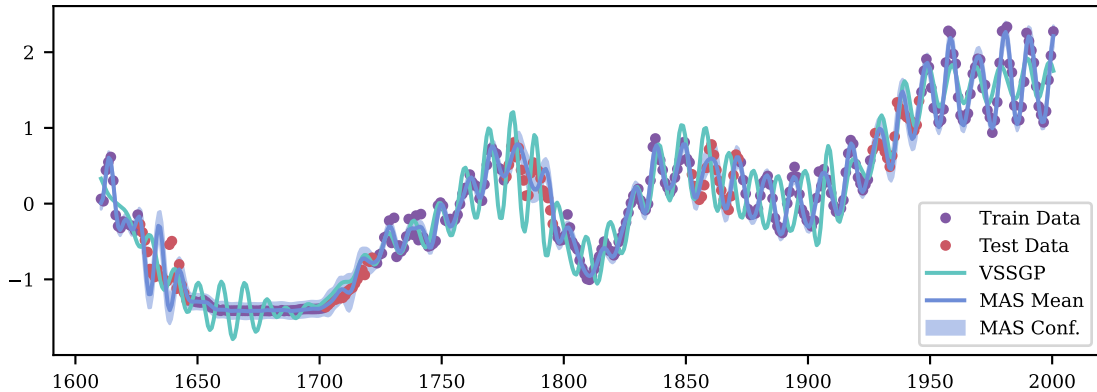


Figure 3.4: MASKERADE and VSSGP on the Solar Irradiance Data Set. “MAS” in the legend refers to MASKERADE, and “Conf.” refers to the (2 standard deviation) confidence interval. For clarity, we show only the posterior mean for VSSGP. Note that VSSGP is a sparse spectrum method, which is why it sometimes fails to achieve low error at the training points. Whilst VSSGP is able to approximate well the underlying periodic structure of the function, MASKERADE is able to generalise better.

We further examine the algorithms’ performance on four datasets from the UCI Machine Learning Repository (Dua and Graff 2017): Yacht Hydrodynamics (308 instances, 6 input dimensions) (Lopez 2013), Auto MPG (398 instances, 8 input dimensions) (Ross 1993), Concrete Compressive Strength (1030 instances, 8 input dimensions) (Yeh 2007), and Airfoil Self-Noise (1503 instances, 5 input dimensions) (Lopez 2014).

To compare against FKL we follow (Benton et al. 2019) and average over 10 runs, randomly partitioning the data into 90/10 train/test sets every time. These results are presented in Table 3.2.

Similarly, to compare against BaNK we follow (Oliva et al. 2016) and perform 3 repeats of 5-fold cross-validation on the Concrete Compressive Strength and Airfoil Self-Noise datasets, for which we present results in Table 3.3.

3.6.4 Large scale data sets

Finally we compare MASKERADE against FKL on two large datasets, with a limited time budget of 20 minutes for learning. We average over 10 runs with a 90/10

Dataset	LL		P-value
	FKL	MASKERADE	
yacht	-32.071 ± 8.45	47.163 ± 5.757	2.97×10^{-9}
autompg	-103.57 ± 5.638	-57.265 ± 3.95	3.56×10^{-9}
concrete	-300.953 ± 34.136	-122.696 ± 6.485	9.72×10^{-8}
airfoil	-863.545 ± 35.959	-151.54 ± 13.263	1.65×10^{-13}

Dataset	RMSE		P-value
	FKL	MASKERADE	
yacht	0.631 ± 0.358	2.013 ± 0.576	1.08×10^{-6}
autompg	3.089 ± 0.358	8.137 ± 0.655	6.77×10^{-9}
concrete	5.808 ± 2.651	13.148 ± 1.314	2.78×10^{-5}
airfoil	83.854 ± 12.335	4.592 ± 0.623	1.06×10^{-8}

Table 3.2: Posterior log likelihood of and RMSE on the test set for UCI MLR datasets. The P-values are based on paired Student-t tests.

Dataset	BaNK	MASKERADE
concrete	0.1195 ± 0.0108	0.6889 ± 0.0168
airfoil	0.3359 ± 0.0354	0.2578 ± 0.0866

Table 3.3: MSE on the (normalised) test set for two UCI data sets. Results for BaNK taken from (Oliva et al. 2016).

train/test split and report results in Table 3.4. The P-values are based on paired t-tests. MASKERADE once more achieves superior performance to FKL on the same time budget in terms of log likelihood. The datasets are the Sterling Broad Based Exchange Rate (England 2020) (data from 1990-01-02–2020-10-08, totalling 7781 data points) and the UCI 3D Road Network (Kaul 2013) (10,000 data point subset of 250,000 data points. We choose the first 10,000 instances with a unique OSM-ID) for which the task is to predict altitude from latitude and longitude.

3.6.5 Ablation Study

We conduct an ablation study to verify the effectiveness of two components of our model in Table 3.5. Firstly, we verify that marginalising across the kernel family does indeed lead to better predictive performance. As a baseline, we compare against

Dataset	LL		P-value
	FKL	MASKERADE	
sterling	-1053.566 ± 30.697	335.37 ± 53.582	7.81×10^{-8}
road alt	-4355.296 ± 219.847	-1070.571 ± 21.548	7.19×10^{-12}

Dataset	RMSE		
	FKL	MASKERADE	
sterling	1.042 ± 0.028	0.523 ± 0.042	1.77×10^{-10}
road alt	10.17 ± 0.62	12.798 ± 0.427	3.68×10^{-6}

Table 3.4: Posterior log likelihood of and RMSE on the test set for FKL and MASKERADE fit to two large datasets. Both methods were given a time budget of 20 minutes for training.

a GP using Spectral Mixture kernel (SM) whose hyperparameters were optimised via gradient descent to maximise the likelihood of the training data. Secondly, we compare our proposed acquisition function against uncertainty sampling (Gunter et al. 2014) and random point acquisition. We set an evaluation budget of 1000 and evaluate performance on the Airline Passenger (Makridakis, Wheelwright and Hyndman 1998) and Mauna Loa Atmospheric CO₂ Concentration (NOAA 2020) datasets. We average over 10 runs with a 90/10 train/test split. In both cases, our chosen approach compares favourably to alternative designs.

Dataset	SM	MASKERADE-I
airpass	0.283 ± 0.000	0.162 ± 0.020
mauna loa	0.802 ± 0.685	0.026 ± 0.002

	MASKERADE-U	MASKERADE-R
	0.164 ± 0.027	0.163 ± 0.022
	0.029 ± 0.001	0.030 ± 0.002

Table 3.5: Test set RMSE for the Airline Passenger and Mauna Loa datasets. MASKERADE-I indicates the use of our proposed information theoretic acquisition function, MASKERADE-U indicates uncertainty sampling (Gunter et al. 2014), and MASKERADE-R indicates random sampling under the prior.

3.7 Discussion

We have introduced a novel framework for kernel learning for Gaussian Processes that seeks to marginalise over Spectral Mixture Kernels using Bayesian Quadrature. Specifically we use a maximum mean discrepancy as a metric underlying an exponential kernel to define a Gaussian Process on a space of GMMs, and show how this can be efficiently computed. This elegantly incorporates invariances between Spectral Mixture Kernels into our model. Additionally we show that an information-theoretic acquisition function is applicable for warped Bayesian Quadrature, and how to use it within our framework. We empirically evaluate our method on several datasets and find that it is competitive with state-of-the-art baselines.

The key limitation of our method is due to the curse of dimensionality. The most scalable instantiation of our framework handles multi-dimensionality by using GMMs in which each Gaussian has a diagonal covariance structure. This means that the number of parameters for an SM kernel with M mixtures is $M(1 + 2D)$, which grows faster than the number of data dimensions, D . This limits the effectiveness of any Bayesian Quadrature regime. Note that Monte Carlo methods also suffer in high dimensional spaces, especially if likelihood evaluations are expensive. (Our method relies on MC integration of kernel integrals, but these are considerably easier to compute than the marginalisation integrals because our method only requires samples from the priors over hyperparameters rather than the posteriors, and the kernel is much cheaper to evaluate than the likelihood.) Our approach is most beneficial for inference based on large, low dimensional datasets.

The societal impacts of this work will depend on the problems that practitioners apply it to, as it is a general-purpose framework.

3.8 Supplement

3.8.1 Summary of WSABI

For completeness we include a summary of the Warped Sequential Active Bayesian Integration (WSABI) algorithm proposed by (Gunter et al. 2014), upon which our work builds heavily.

Recall that we are interested in integrals of the form $\int f(\theta)d\pi(\theta)$, where $f(\theta)$ is non-negative. We enforce this constraint by modelling $g(\theta) = \sqrt{2(f(\theta) - \epsilon)} \sim \mathcal{GP}(0, \kappa)$. This induces a non-central Chi-squared distribution over $f(\theta)$.

Now, denote by Θ the set of observation locations, the matching transformed observations by $z = \sqrt{2(f(\Theta) - \epsilon)}$, the covariance between all pairs of observations as $\kappa_{\Theta\Theta}$, the covariance between an arbitrary point θ and the set of observations as $\kappa_{\theta\Theta}$ and the kernel between two arbitrary points as $\kappa_{\theta_1\theta_2}$. Additionally, let $\mathfrak{D} = (\Theta, z)$.

Further, recall that the posterior over $g(\theta)$ conditioned on z is given by $\mathcal{GP}(\mu_{\mathfrak{D}}, \Sigma_{\mathfrak{D}})$ with:

$$\begin{aligned}\mu_{\mathfrak{D}} &= \kappa_{\theta\Theta}\kappa_{\Theta\Theta}^{-1}z \\ \Sigma_{\mathfrak{D}} &= \kappa_{\theta_1\theta_2} - \kappa_{\theta_1\Theta}\kappa_{\Theta\Theta}^{-1}\kappa_{\Theta\theta_2}^T\end{aligned}$$

By taking a Taylor expansion around the posterior mean of $g(\theta)$, $\mu_{\mathfrak{D}}$, we can approximate

$$\begin{aligned}f(g) &\approx f(\mu_{\mathfrak{D}}) + (g - \mu_{\mathfrak{D}})\frac{df}{dg}\Big|_{g=\mu_{\mathfrak{D}}} \\ &= \epsilon - \frac{1}{2}\mu_{\mathfrak{D}}^2 + \mu_{\mathfrak{D}}g.\end{aligned}$$

As this is a linear transform of g , we can approximate the distribution over f as

$\mathcal{GP}(\mathbf{m}_{\mathfrak{D}}, \mathfrak{K}_{\mathfrak{D}})$ with moments

$$\begin{aligned}\mathbf{m}_{\mathfrak{D}}(\theta) &= \epsilon + \frac{1}{2}\mu_{\mathfrak{D}}(\theta)^2, \\ \mathfrak{K}_{\mathfrak{D}}(\theta_1, \theta_2) &= \mu_{\mathfrak{D}}(\theta_1)\Sigma_{\mathfrak{D}}(\theta_1, \theta_2)\mu_{\mathfrak{D}}(\theta_2).\end{aligned}$$

Then the first two moments of the integral of interest are given by

$$\begin{aligned}\mathbb{E}\left[\int f(\theta)d\pi(\theta)\right] &= \int \mathbb{E}[f(\theta)]d\pi(\theta) = \int \left(\epsilon + \frac{\mu_{\mathfrak{D}}(\theta)^2}{2}\right)d\pi(\theta) \\ &= \epsilon + \frac{1}{2}\left(z^T \kappa_{\Theta\Theta}^{-1} \int \kappa_{\theta\Theta}^T \kappa_{\theta\Theta} d\pi(\theta) \kappa_{\Theta\Theta}^{-1} z\right) \\ &= \epsilon + \frac{1}{2}z^T Q z.\end{aligned}$$

and

$$\begin{aligned}\text{Cov}\left[\int f(\theta)d\pi(\theta)\right] &= \int \int \mu_{\mathfrak{D}}(\theta)\Sigma_{\mathfrak{D}}(\theta, \theta')\mu_{\mathfrak{D}}(\theta')d\pi(\theta)d\pi(\theta') \\ &= z^T \kappa_{\Theta\Theta}^{-1} \left(\int \int \kappa_{\theta\theta'} \kappa_{\theta\Theta}^T \kappa_{\theta'\Theta} d\pi(\theta)d\pi(\theta') \right. \\ &\quad \left. - \int \kappa_{\theta\Theta}^T \kappa_{\theta\Theta} d\pi(\theta) \kappa_{\Theta\Theta}^{-1} \int \kappa_{\theta'\Theta}^T \kappa_{\theta'\Theta} d\pi(\theta') \right) \kappa_{\Theta\Theta}^{-1} z\end{aligned}$$

Where the quadrature weights, $Q = \kappa_{\Theta\Theta}^{-1} \int \kappa_{\theta\Theta}^T \kappa_{\theta\Theta} d\pi(\theta) \kappa_{\Theta\Theta}^{-1}$, only depend on the particular kernel choice and locations of already observed function values.

For the numerator of Equation (4) in the main text we simply substitute $z_* = \sqrt{2(p(y_* | x_*, D, \theta)p(D | \theta) - \epsilon)}$ for $z = \sqrt{2(p(D | \theta) - \epsilon)}$. The predictive posterior is then a weighted sum of products of Gaussians.

3.8.2 A Remark on Posterior Inference

The predictive posterior given in (3.4) is typically viewed through the hierarchical Bayesian framework:

$$\begin{aligned}
p(y_* | x_*, D) &= \frac{\int p(y_* | x_*, \Theta, D) p(D | \Theta) p(\Theta) d\Theta}{\int p(D | \Theta) p(\Theta) d\Theta} \\
&= \frac{\int \int p(y_* | x_*, \theta^{(n)}, D) p(D | \theta^{(n)}) p(\theta^{(n)}, n) d\theta^{(n)} dn}{\int \int p(D | \theta^{(n)}) p(\theta^{(n)}, n) d\theta^{(n)} dn} \\
&= \frac{\sum_{n=1}^N p(n) \int p(y_* | x_*, \theta^{(n)}, D) p(D | \theta^{(n)}) p(\theta^{(n)}) d\theta^{(n)}}{\sum_{n=1}^N p(n) \int p(D | \theta^{(n)}) p(\theta^{(n)}) d\theta^{(n)}},
\end{aligned} \tag{3.9}$$

where $\theta^{(n)}$ is the description of a GMM with exactly n components. For our experiments we choose uniform priors over n , which cancel (note that this is not an inherent limitation of our method).

A naïve approach performs the integrals over $\theta^{(n)}$ separately before computing the sums over n . The ability of MASKERADE to share information across models enables the selection of acquisitions which are maximally informative about the *sum of integrals rather than each individual integral separately*, improving the convergence rate. Additionally, accounting for the covariance between integrals over $\theta^{(n)}$ improves the quality of the posterior over the result of the summations.

Derivation of Information Theoretic Sample Acquisition

The acquisition function is the expected information gained about the integral by making a set of likelihood observations

$$\alpha(\theta_*^{(n)}) = \int (H[Z | z, \Theta] - H[Z | z_*, \theta_*^{(n)}, z, \Theta]) p(z_* | \theta_*^{(n)}, z, \Theta) dz_*$$

Due to the symmetry of the mutual information, we can swap of Z and z_* [52]

$$\alpha(\theta_*^{(n)}) = H[z_* | z, \Theta] - \int H[z_* | \theta_*^{(n)}, Z, z, \Theta] p(Z | z, \Theta) dZ$$

$$= H[z_* | z, \Theta] - H[z_* | \theta_*^{(n)}, Z, z, \Theta]$$

where H is differential entropy (always of Gaussians). The second line follows because the entropy of the Gaussian $p(z_* | \theta_*^{(n)}, Z, z, \Theta)$ does not depend on the value of Z .

Monte Carlo approximation to the kernel integrals

The kernel integrals are approximated using Quasi Monte Carlo sampling under the prior. Table 3.6 shows the effect of varying the number of QMC samples.

No. Observations	No. of MC samples		
	100	1000	10000
100	0.02546 ± 0.00149	0.02542 ± 0.00020	0.02526 ± 0.00012
500	0.00369 ± 0.00035	0.00379 ± 0.00013	0.00358 ± 0.00003
1000	0.00333 ± 0.00095	0.00292 ± 0.00031	0.00292 ± 0.00006

Table 3.6: The effect of number of Monte Carlo samples used to estimate the kernel integrals (i.e. the those described in Section 3.8.1). For a MASKERADE model that marginalises over up to 5 mixture components we randomly sample sets of 100, 500 and 1000 hyperparameters from the prior, and compute their corresponding likelihoods on the UCI Airfoil Self-Noise dataset. We then infer the model evidence with WSABI using the MASKERADE hyper-kernel. For a given number of MC samples, we repeat this five times and report the mean and SEM for the posterior mean of the model evidence. We observe that the estimate for the model evidence is not highly sensitive to the number of monte carlo samples used to compute the kernel integrals.

3.8.3 Full Algorithm Description

For completeness we once more give the full generative model:

$$\begin{aligned}
n &\sim \text{Uniform}(\{1, 2, \dots, N\}), \\
w &\sim \text{Dirichlet}(\alpha), \\
m_{1..n} &\sim \mathcal{N}(\mu, \Sigma), \\
\sigma_{1..n} &\sim \text{Log-Normal}(\nu, \tau), \\
S(\omega) &= \sum_{j=1}^n \frac{w_j}{2} (\mathcal{N}(\omega; m_j, \sigma_j) + \mathcal{N}(\omega; -m_j, \sigma_j)), \\
k(x, x') &= \int e^{2\pi i |x-x'| \omega} S(\omega) d\omega, \\
f &\sim \mathcal{GP}(0, k).
\end{aligned} \tag{3.10}$$

Below, we provide a summary of the MASKERADE framework in Algorithm 1.

A schematic outlining our method is made available in Figure 3.5.

Algorithm 1 Pseudocode for both the learning and prediction phases of our algorithm.

```

obtain initial samples  $\Theta, z$  ▷  $z$  are likelihood evaluations at  $\Theta$ .
 $\lambda, l \leftarrow \text{argmax}_{\lambda, l} p(z \mid \Theta, \lambda, l)$  ▷ Optimise BQ Surrogate
while  $i \geq 0$  do
     $\{\theta_s^{(n)}\}_{1:b} \leftarrow \text{argmax}_{\{\theta_s^{(n)}\}_{1:b}} \alpha(\{\theta_s^{(n)}\}_{1:b})$  ▷ Optimise acquisition function
    append  $\{\theta_s^{(n)}\}_{1:b}$  to  $\Theta$ .
    append likelihood( $\{\theta_s^{(n)}\}_{1:b}$ ) to  $z$ .
     $\lambda, l \leftarrow \text{argmax}_{\lambda, l} p(z \mid \Theta, \lambda, l)$  ▷ Optimise BQ surrogate
     $i \leftarrow i - 1$ 
end while
 $Q \leftarrow \text{compute\_bq\_weights}(\Theta, z, \lambda, l, \phi)$  ▷  $Q$  defined as in Appendix on WSABI
above.  $\phi$  are all prior parameters.
for  $\theta_s^{(n)}$  in  $\Theta$  do
     $\mu_*, \Sigma_* \leftarrow \text{predict}(x_*, D, \theta_s^{(n)})$ 
end for
return predictive_posterior( $Q, \{\mu_*\}, \{\Sigma_*\}, z$ ) ▷ See Appendix on WSABI
for details.

```

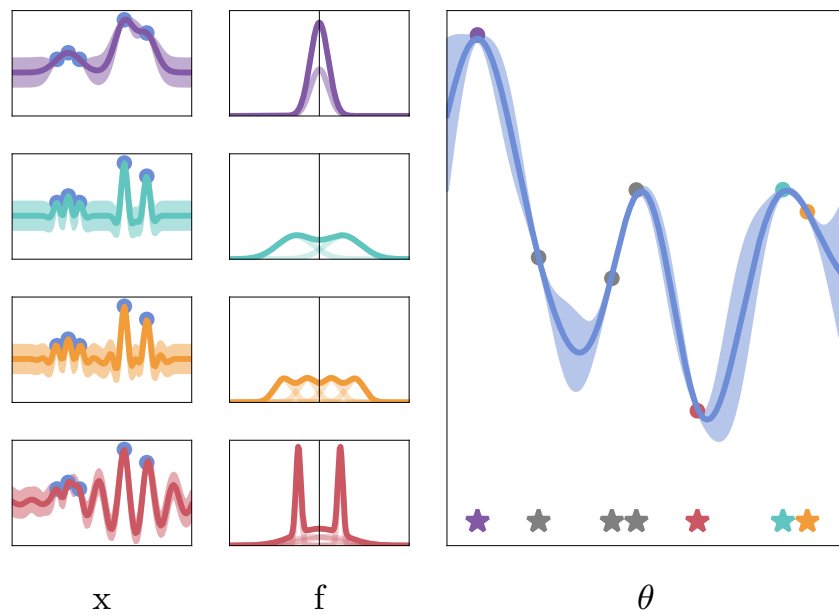


Figure 3.5: A schematic representation of the inference procedure under our model. The left column shows the GP posterior for different Spectral Mixture Kernels on the same dataset. The centre column shows the SM kernels in their spectral domain. The right column illustrates a GP posterior on a space indexed by the spectral densities of the SM kernels. BQ can then be used to marginalise over SM kernels which may have different numbers of mixture components.


Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).


Title of Paper	Marginalising over Stationary Kernels with Bayesian Quadrature
Publication Status	<input type="checkbox"/> Published <input checked="" type="checkbox"/> Accepted for Publication <input type="checkbox"/> Submitted for Publication <input type="checkbox"/> Unpublished and unsubmitted work written in a manuscript style
Publication Details	Hamid, S., Schulze, S., Osborne, M. & Roberts, S. (2022). Marginalising over Stationary Kernels with Bayesian Quadrature. In Proceedings of the 25 th International Conference on Artificial Intelligence and Statistics (AISTATS)

Student Confirmation

Student Name:	Saad Hamid		
Contribution to the Paper	<ul style="list-style-type: none">• Joint formulation of initial research direction.• Investigated approaches for incorporating salient invariances into BQ scheme.• Joint discussion of experimental design.• Majority of coding of experiments.• Contributed to writing-up the paper.		
Signature		Date	11/01/2023

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Prof Michael A. Osborne			
Supervisor comments I certify that the candidate made a substantial contribution to the publication, just as described above.			
Signature		Date	11 January 2023

This completed form should be included in the thesis, at the end of the relevant chapter.

Chapter 4

Bayesian Quadrature for Neural Ensemble Search

Contents

4.1	Abstract	58
4.2	Introduction	59
4.3	Background	61
4.3.1	Neural Architecture Search	61
4.3.2	Bayesian Optimisation for Neural Architecture Search	64
4.3.3	Neural Ensemble Search	65
4.3.4	Bayesian Quadrature	66
4.3.5	Recombination	68
4.4	Bayesian Quadrature for Neural Ensemble Search	68
4.4.1	Building the Candidate Set	69
4.4.2	Selecting the Ensemble	71
4.5	Experiments	73
4.5.1	Ablation Study	75
4.6	Discussion and Future Work	82
4.7	Supplement	83
4.7.1	Verification of Surrogate Quality	83
4.7.2	Additional Experiments	84

Preface

The previous chapter focussed on using Bayesian Quadrature for model selection in Gaussian Processes. In a similar vein, this chapter will explore the use of Bayesian Quadrature for model selection in Neural Networks.

Neural Networks (NNs) are a flexible class of models that have achieved state-of-the-art results in recent years on a range of tasks from image classification to semantic segmentation and speech recognition. The architecture of a neural network encodes strong inductive biases, and is well known to significantly affect the performance achievable by a network. This has motivated authors to design systems for automatic selection of NN architectures, a problem referred to as Neural Architecture Search. The community has developed several methods of specifying search spaces from which an architecture $\alpha \in \mathcal{A}$ is to be selected, along with a range of search strategies. More recently, it has been shown that ensembles of NNs can outperform single architectures, motivating the automatic selection of ensembles of NNs, a problem referred to as Neural Ensemble Search. The key challenges in this setting are that the search space of possible architectures can be very large, and that the evaluation of a given architecture is computationally expensive since it requires training that architecture (or marginalising over its weights).

In this chapter, we view ensembling as approximately performing marginalisation over architectures. This perspective allows us to bring the tools of Bayesian Quadrature to bear upon the problem of Neural Ensemble Search. Specifically, the contributions of this chapter are:

- To investigate the use of active Bayesian Quadrature as a means of selecting the candidate set of architectures to train.
- To compare methods of selecting the ensemble from this candidate set, and setting the relative weights of the members.

4.1 Abstract

Ensembling can improve the performance of Neural Networks, but existing approaches struggle when the architecture likelihood surface has dispersed, narrow peaks. Furthermore, existing methods construct equally weighted ensembles, and

this is likely to be vulnerable to the failure modes of the weaker architectures. By viewing ensembling as approximately marginalising over architectures we construct ensembles using the tools of Bayesian Quadrature – tools which are well suited to the exploration of likelihood surfaces with dispersed, narrow peaks. Additionally, the resulting ensembles consist of architectures weighted commensurate with their performance. We show empirically – in terms of test likelihood, accuracy, and expected calibration error – that our method outperforms state-of-the-art baselines, and verify via ablation studies that its components do so independently.

4.2 Introduction

Neural Networks (NNs) are extremely effective function approximators but their architectures are typically designed by hand, a painstaking process. To address this there has been significant interest in the definition of, and the automatic selection from, search spaces of NN architectures. Recent work shows that ensembles of architectures from a given search space can outperform the *single* best architecture (Shu et al. 2022; Zaidi et al. 2022). Such ensembles improve performance on a range of metrics, including the test set’s predictive accuracy, likelihood, and expected calibration error. The latter two metrics measure the quality of the model’s uncertainty estimates, which have been shown for single architectures to be poor (Guo et al. 2017). Performant models in this regard are crucial for systems which make critical decisions, such as self-driving vehicles. Naturally, ensemble selection is an even more difficult problem to tackle manually than selecting a single architecture. Hence, interest in methods for automatic ensemble construction is growing. This paper targets exactly this problem.

Conceptually, Neural Ensemble Search (NES) algorithms can be split into two stages. The first is the *candidate selection* stage, which seeks to characterise the posterior distribution, $p(\alpha | D)$, given the training data D , over architectures from

a given search space $\alpha \in \mathcal{A}$. Multiple approaches have been proposed. One such is an evolutionary strategy which seeks the modes of this distribution (Zaidi et al. 2022); another is training a “supernet” and using it to learn the parameters of a variational approximation to this distribution (Shu et al. 2022). This involves evaluating the likelihood of a set of architectures from the search space, an evaluation which requires first training the architecture weights. The second stage is *ensemble selection*, where the ensemble members are selected from the candidate set and each member’s weight is chosen. Several approaches have also been suggested for ensemble selection, such as beam search and sampling from the (approximate) posterior over architectures.

In this work, we investigate novel approaches to both stages of a NES algorithm. Taking a hierarchical Bayesian perspective, we view ensembling as approximately performing marginalisation over a given search space of architectures. This paradigm allows us to bring the tools of Bayesian Quadrature to bear upon the problem of Neural Ensemble Search. Specifically, the contributions of this work are to:¹

- Propose using an acquisition function for adaptive Bayesian Quadrature to select the candidate set of architectures to train. It is from these that the ensemble members are later selected.
- Show how recombination of the approximate posterior over architectures can be used to construct a weighted ensemble from the candidate set.
- Undertake an empirical comparison of our proposals against state-of-the-art baselines. Additionally, we conduct ablation studies to understand the effect of our proposals for each stage of the NES pipeline.

¹An implementation of our experiments can be found at https://github.com/saadhamidml/bq_nes.

4.3 Background

4.3.1 Neural Architecture Search

Neural Architecture Search (NAS) is typically formulated as an optimisation problem, i.e. maximising some measure of performance f over a space of NN architectures \mathcal{A} ,

$$\alpha_* = \operatorname{argmax}_{\alpha \in \mathcal{A}} f(\alpha). \quad (4.1)$$

(Elsken, Metzen and Hutter 2019) identify three conceptual elements of a NAS pipeline: a search space, a search strategy, and a performance estimation strategy.

The first part of a NAS pipeline – the search space – is the way in which the possible space of NN architectures is defined. In this work we require that the search space be such that a Gaussian Process (GP) can be defined upon it. In particular we will restrict our attention to cell-based search spaces that construct architectures by inserting stacks of a cell into a larger, fixed macro-skeleton (Dong, Liu et al. 2021), as recent results have shown that GPs are well suited to modelling functions on such spaces (Ru, Wan et al. 2021). (However, we investigate alternative search spaces in Appendix 4.7.2.) Searching over architectures then becomes equivalent to searching over cells. Cells are represented as labelled Directed Acyclic Graphs (DAG) that connect a fixed number of nodes with operations selected from a pre-specified operation set (usually also including the possibility of a “zeroise” connection, which is equivalent to removing the connection).

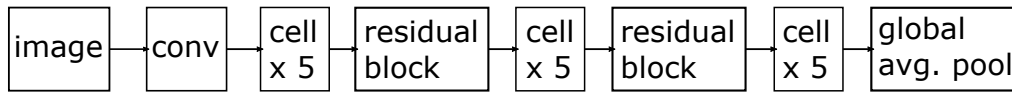
A NAS pipeline’s second phase is the search strategy. This is a procedure for selecting which architectures to query the performance of. All strategies will exhibit an exploration-exploitation trade-off, where exploration is covering the search space well, and exploitation is selecting architectures that are similar to the well-performing architectures in the already queried history.

The final element of a NAS pipeline is the performance estimation strategy, which is the method for querying the performance of a given architecture. Typically, this

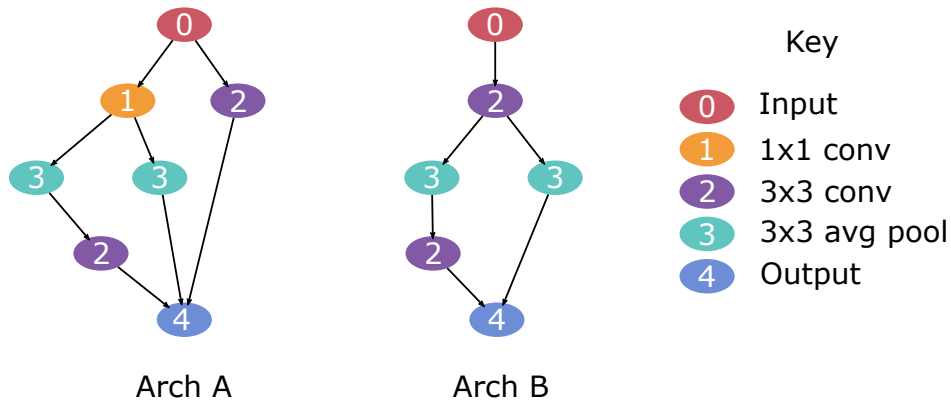
is done by training the NN weights, given the architecture, on a training dataset, and evaluating its performance on a validation set. However, this demands excessive computing and practically limits the total number of architecture evaluations available to the search strategy.

There is a large body of literature devoted to the problem of Neural Architecture Search, pursuing a range of strategies such as defining a differentiable search space (Chen, Xie et al. 2019; Liu, Simonyan and Yang 2019; Xu et al. 2020), evolutionary strategies (Liu, Sun et al. 2021; Real, Aggarwal et al. 2019; Real, Moore et al. 2017), and Bayesian Optimisation (Kandasamy et al. 2019; Ma, Cui and Yang 2019; Ru, Wan et al. 2021; White, Neiswanger and Savani 2020).

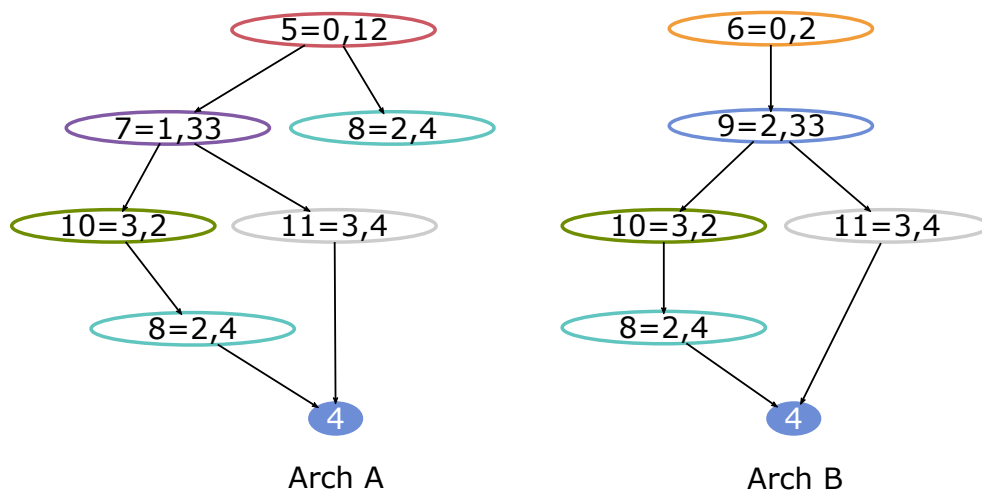
Macro-skeleton



Graph Representation of Cells



Label Aggregation



Feature Vectors

h=0 features	0	1	2	3	4	h=1 features	5	6	7	8	9	10	11
Arch 0	1	1	2	2	1	Arch 0	1	0	1	2	0	1	1
Arch 1	1	0	2	2	1	Arch 1	0	1	0	1	1	1	1

Figure 4.1: A diagram showing the process of building the WL Kernel’s feature vectors for NAS. At the top is the macro-skeleton which is varied by changing the cell. Defining the search space in this way allows us to model the objective function on the space of cells. Cells are represented as labelled DAGs. The $h = 0$ level features are simply histograms of the labels for each node in the graph. To compute the features at the next level, each node has its label appended with the aggregated labels of all nodes in its 1-out-neighbourhood. The $h = 1$ level features are histograms of these modified labels. Higher level features are computed by aggregating the labels for larger out-neighbourhoods.

4.3.2 Bayesian Optimisation for Neural Architecture Search

An effective approach to NAS is Bayesian Optimisation (BO). At a high level, BO models the objective function f and sequentially selects where to query next based on an acquisition function, with the goal of finding the optimal value of the objective function in a sample efficient manner. Typically, the objective function is modelled using a Gaussian Process (GP) – a stochastic process for which all finite subsets of random variables are joint normally distributed (Rasmussen and Williams 2006).

A GP is defined using a mean function $m(\alpha)$ that specifies the prior mean at α , and a kernel function $k(\alpha, \alpha')$ that specifies the prior covariance between $f(\alpha)$ and $f(\alpha')$. The posterior, conditioned on a set of observations $A = \{(\alpha_i, y_i)\}_i^N$ and $y = [f(\alpha_1), \dots, f(\alpha_N)]^T$, is also a GP with moments

$$m_A(\cdot) = m(\cdot) + K_{\cdot A} K_{AA}^{-1} (y - m(\alpha)) \quad \text{and} \quad (4.2)$$

$$k_A(\cdot, \cdot') = K_{\cdot \cdot'} - K_{\cdot A} K_{AA}^{-1} K_{A \cdot}. \quad (4.3)$$

The prior mean function m is typically set to zero.

Ru, Wan et al. (2021) showed that the Weisfeiler-Lehman graph kernel (WL kernel) (Shervashidze 2011) is an appropriate choice for modelling NN performance metrics on a cell-based NAS search space with a GP. To apply the WL kernel, a cell first needs to be represented as a labelled DAG. Next, a feature vector is built up in the following way:

1. Compute a histogram of the labels. These are the $h = 0$ level features.
2. Relabel each node with a new label generated by aggregating the labels of the nodes in its 1-out-neighbourhood (as we are now computing the $h = 1$ level features). The h -out-neighbourhood of a node is the set of nodes that are reachable by traversing h edges starting from that node. Then compute a histogram of these aggregated labels. These are the $h = 1$ level features.

3. Repeat (2) up to a pre-specified depth $h = H$.
4. Generate the feature representation as the aggregate of the histograms for each level $h = 0$ to $h = H$.

The kernel is then computed as the dot product of the feature vectors for a pair of graphs. This process is shown diagrammatically in Figure 4.1.

A common acquisition function for BO is Expected Improvement (Garnett 2021),

$$a_{EI}(\alpha) = \mathbb{E}_{p(f|D)} \left[\max(f(\alpha) - f(\hat{\alpha}), 0) \right] \quad (4.4)$$

where $\hat{\alpha}$ is the best architecture found so far. Using this acquisition function in conjunction with a GP using the WL kernel was shown by Ru, Wan et al. (2021) to be effective for NAS.

4.3.3 Neural Ensemble Search

Neural Ensemble Search (Zaidi et al. 2022) is a method for automatically constructing ensembles of a given size, M , from a NAS search space \mathcal{A} . First, a candidate set of architectures, $A \subset \mathcal{A}$, is selected using a regularised evolutionary strategy (NES-RE), or random sampling from the search space. The authors propose several ensemble selection methods to subsequently select a subset of M architectures $A_M \subset A$. Of particular interest in this work are Beam Search (BS) and Weighted Stacking (WS).

BS initially adds the best performing architecture to the ensemble and greedily adds the architecture from the candidate set (without replacement) that most improves the validation loss of the ensemble. WS optimises the ensemble weights over the whole candidate set on the validation loss (subject to the weights being non-negative and summing to one). The members with the highest M weights are included in the ensemble, and their corresponding weights renormalised. The authors compare BS to WS on the CIFAR-10 dataset, and find performance in terms

of the log likelihood of the test set to be better for BS for small ensembles, but similar for larger ensembles.

Neural Ensemble Search via Bayesian Sampling (Shu et al. 2022) approximates the posterior distribution over architectures $p(\alpha | D)$ with a variational distribution of the form $q(\alpha) = \prod_i q_i(o | \theta_i)$, where i iterates over the connections within a cell, o is the operation for connection i , and θ_i are the variational parameters for q_i . The form of q_i is chosen to be a softmax over θ_i . The ensemble is then selected by using Stein Variational Gradient Descent with Regularised Diversity to select a diverse set of M samples from (a continuous relaxation of) the variational distribution.

Relatedly, DeepEnsembles (Lakshminarayanan, Pritzel and Blundell 2017) seeks to approximately marginalise over the parameters of a given NN architecture. The architecture is trained from several random initialisations, and the ensemble makes a prediction as an equally weighed sum of these. This is orthogonal to the work above (and indeed our work), which seeks to construct ensembles of different architectures, rather than ensembles of different parameter settings of the same architecture.

4.3.4 Bayesian Quadrature

Bayesian Quadrature (BQ) (Minka 2000; O’Hagan 1991) is a probabilistic numerical integration technique that targets the computation of $Z = \int f(x)d\pi(x)$ based on evaluations of the integrand f (assuming a given prior π). Similar to BO, it maintains a surrogate model over the integrand f , which induces a posterior over the integral value Z . BQ also makes use of an acquisition function to iteratively select where next to query the integrand.

The surrogate model for BQ is usually chosen to be a GP, and this induces a Gaussian posterior over $Z \sim \mathcal{N}(\mu_Z, \sigma_Z)$. The moments of this posterior are given by

$$\mu_Z = \int K(x, X)d\pi(x)K_{XX}^{-1}f, \quad \text{and} \quad (4.5)$$

$$\sigma_Z = \int K(x, x') - K(x, X)K_{XX}^{-1}K(X, x')d\pi(x)d\pi(x'), \quad (4.6)$$

where X is the set of query points, and f are the corresponding integrand observations. Note that the posterior mean μ_Z takes the form of a quadrature rule – a weighted sum of function evaluations $\sum_i w_i f(x_i)$ where w_i are the elements of the vector $\int K(x, X)d\pi(x)K_{XX}^{-1}$.

Often non-negative integrand are of interest and warped Bayesian Quadrature (Chai and Garnett 2019; Gunter et al. 2014; Osborne, Duvenaud et al. 2012) allows practitioners to incorporate this prior information into the surrogate model. Of particular interest in this work will be the WSABI-L model (Gunter et al. 2014), which models the square-root of the integrand with a GP, $\sqrt{2(f(x) - \beta)} \sim \mathcal{GP}(\mu_D(x), \Sigma_D(x, x'))$. This induces a (non-central) chi-squared distribution over f which can be approximated with a GP, with moments

$$m(x) = \beta + \frac{1}{2}\mu_D(x)^2, \quad (4.7)$$

$$k(x, x') = \mu_D(x)\Sigma_D(x, x')\mu_D(x'). \quad (4.8)$$

(Gunter et al. 2014) established, empirically, that the uncertainty sampling acquisition function works well for Bayesian Quadrature. This acquisition function targets the variance of the integrand

$$a_{US}(x) = \Sigma_D(x, x)\mu_D(x)^2\pi(x)^2. \quad (4.9)$$

This naturally trades off between exploration (regions where $\Sigma_D(x, x)$ is high), and exploitation (regions where $\mu_D(x)$ is high – most of the volume under the integrand is concentrated here).

Just as BO is a natural choice for NAS – an expensive black-box optimisation problem – so BQ is a natural choice for NES – an expensive black-box marginalisation problem. It is this realisation that inspires our proposals in Section 4.4.

4.3.5 Recombination

Given a non-negative measure supported on N points $\{(w_n, x_n)\}_{n=1}^N$ where $w_n \geq 0$ and $\sum_{n=1}^N w_n = 1$, and $M - 1$ “test” functions $\{\phi_t(\cdot)\}_{t=1}^{M-1}$, it is possible to find a subset of $M < N$ points $\{x_n\}_{m=1}^M \subset \{x_n\}_{n=1}^N$ for which

$$\sum_{m=1}^M w_m \phi_t(x_m) = \sum_{n=1}^N w_n \phi_t(x_n) \quad (4.10)$$

for all ϕ_t , with $w_m \geq 0$ and $\sum_{m=1}^M w_m = 1$ (Tchernychova 2015).

For Kernel Quadrature, one can use the Nyström approximation of the kernel matrix to obtain a set of test functions (Hayakawa, Oberhauser and Lyons 2022). Using a subset, S , of $M - 1$ data points, the kernel can be approximated $\tilde{k}(x, x') = k(x, S)k(S, S)^{-1}k(S, x')$. By taking an eigendecomposition, $k(S, S) = U\Lambda U^T$, the approximate kernel can be expressed as

$$\tilde{k}(x, x') = \sum_t^{M-1} \frac{1}{\lambda_t} (u_t^T k(S, x)) (u_t^T k(S, x')) \quad (4.11)$$

where u_i are the columns of U , and λ_i the diagonal elements of Λ . We can then use $\phi_t(\cdot) = u_t^T k(S, \cdot)$ as test functions.

4.4 Bayesian Quadrature for Neural Ensemble Search

We decompose NES into two sub-problems:

1. The selection of a candidate set of architectures $\{\alpha_i\}_{i=1}^N = A \subset \mathcal{A}$ for which to train the architecture parameters.
2. The selection of a set of M members from the candidate set to include in the ensemble, and their weights, $\mathbf{w} \in \mathbb{R}^M$.

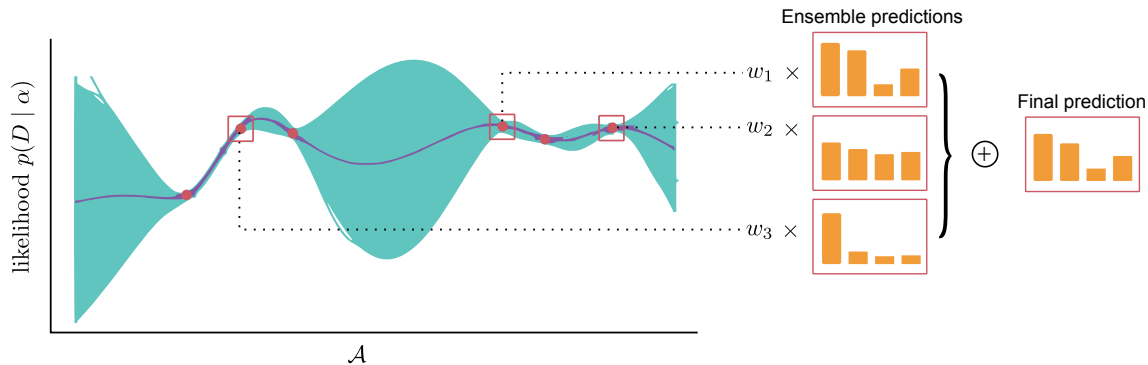


Figure 4.2: A schematic representation of our proposal. The plot on the left shows a Gaussian Process modelling the likelihood over the space of architectures. The architectures to train and evaluate the likelihood for are selected by maximising a Bayesian Quadrature acquisition function, as described in Section 4.4.1. One of the algorithms described in Section 4.4.2 is then used to select the subset of architectures to include in the ensemble, along with their weights. The final prediction is then a linear combination of the predictions of each ensemble member.

We take novel approaches to each of these sub-problems, described respectively in the following two subsections. Algorithms 2, 3 and 4 summarise our propositions.

4.4.1 Building the Candidate Set

Ensembling can appropriately be viewed as approximately performing marginalisation over the search space of architectures. In this light, we suggest using an acquisition function from the Bayesian Quadrature literature for building the candidate set of architectures to train (from which the ensemble members will later be selected).

The connection between ensembling and marginalisation can be seen by examining our ultimate quantity of interest – the posterior predictive – a distribution over class labels $c \in \{1, \dots, C\}$ given an input $x \in \mathcal{X}$ conditioned on a training dataset D , with architectures $\alpha \in \mathcal{A}$ marginalised out.

$$\begin{aligned}
 p(c \mid x, D) &= \sum_{\alpha \in \mathcal{A}} p(c \mid x, \alpha, D) p(\alpha \mid D) \\
 &= \frac{\sum_{\alpha \in \mathcal{A}} p(c \mid x, \alpha, D) p(D \mid \alpha) p(\alpha)}{\sum_{\alpha \in \mathcal{A}} p(D \mid \alpha) p(\alpha)}. \tag{4.12}
 \end{aligned}$$

We see that, to compute the posterior predictive, we need to compute C sums of products of functions of the architecture and the architecture likelihoods. Intuitively, we expect a quadrature scheme that approximates well the sum in the denominator of (4.12) will also approximate the sum in the numerator well. Therefore, we propose using a Bayesian Quadrature acquisition function to build up the candidate set, as these architectures will form the nodes of a query-efficient quadrature scheme for (4.12) and so a good basis for an ensemble. Note that we focus on single-point acquisition functions, resulting in a sequential method. However, batch acquisition functions can be used to parallelise our proposal.

The likelihood of an architecture $p(D | \alpha)$ is not typically available, as this would require marginalisation over the NN weights, $w | \alpha$, of the architecture. However, we can approximate this by assuming the prior over the architecture weights is a delta distribution at the maximiser of the (architecture weights’) likelihood function.

$$\begin{aligned}
 p(D | \alpha) &= \int p(D | w, \alpha) p(w | \alpha) dw \\
 &\approx p(D | \hat{w}, \alpha),
 \end{aligned}
 \tag{4.13}$$

$$\hat{w} = \operatorname{argmax}_w p(D | \hat{w}, \alpha) p(w | \alpha).
 \tag{4.14}$$

Computing $p(y | x, \alpha, D)$ requires an analogous intractable marginalisation. We approximate it similarly, noting that it depends only indirectly on the training data, through the optimisation procedure, i.e.

$$\begin{aligned}
 p(c | x, \alpha, D) &= \int p(c | x, w, \alpha, D) p(w | \alpha, D) dw \\
 &\approx p(c | x, \hat{w}, \alpha).
 \end{aligned}
 \tag{4.15}$$

Concretely, we place a functional prior on the architecture likelihood surface, warped using the square-root transform, $\sqrt{2(p(D | \hat{w}, \alpha) - \beta)} \sim \mathcal{GP}$, and use uncertainty sampling to make observations of the likelihood at a set of architectures

$\{\alpha_i\}_{i=1}^N = A \subset \mathcal{A}$.

This provides us with an estimate of the model evidence $Z = \sum_{\alpha \in \mathcal{A}} p(D | \hat{w}, \alpha) p(\alpha)$, which we denote \hat{Z} . The computation of this estimate requires Monte Carlo sampling to approximate sums of (products of) the WL-kernel over \mathcal{A} . Note this is far more feasible than approximating the original sums in Equation (4.12) with Monte Carlo sampling as $K(\alpha_j, A)$ is far cheaper to evaluate than $p(D | \hat{w}, \alpha_j)$ or $p(c | x, \hat{w}, \alpha_j)$ – either would require training architecture α_j .

Algorithm 2 NES candidate set selection algorithm using a BQ acquisition function. Returns architectures A and their corresponding validation likelihoods L .

```

A, L ← sample(n, A)                                ▷ Initial samples.
θ ← argmaxθ p(L | A, θ)                            ▷ Optimise WL kernel.
while i > 0 do
    α ← argmaxα ∈ A acquisition_function(α, A, L, θ)
    A ← {A, α}
    L ← {L, p(D |  $\hat{w}$ , α)}
    θ ← argmaxθ p(L | A, θ)
end while
return A, L

```

4.4.2 Selecting the Ensemble

In principle, the ensemble can be constructed using the weights provided by the quadrature scheme, as these weights naturally trade-off between member diversity and member performance. However, we wish to select a subset of the candidate set for the ensemble (as it is assumed that an ensemble of the whole candidate set is too costly to be practical for deployment). Concretely, we seek a subset $A_M \subset A$, along with weights $\mathbf{w} \in \mathbb{R}^M$ such that

$$p(c | x, D) \approx \sum_n^N \frac{p(D | \alpha_n) p(\alpha_n)}{\hat{Z}} p(c | x, \alpha_n, D) + \epsilon \quad (4.16)$$

$$\approx \sum_m^M w_m p(c | x, \alpha_m, D) + \epsilon. \quad (4.17)$$

We expect ϵ to be small if regions of high likelihood have been well-explored by the acquisition function in the building of the candidate set. To select the weights \mathbf{w} and the set A_M we can use any recombination algorithm, using the Nyström approximation to generate the test functions, as described in Section 4.3.5, and the estimated posterior over architectures as the measure to recombine. We refer to this algorithm as Posterior Recombination (PR).

A second approach, which we refer to as Re-weighted Stacking (RS), is a modification of Weighted Stacking. Like for WS, we optimise the weights of an ensemble of the whole candidate set to minimise the validation loss. The ensemble members are then chosen by selecting the members with the M highest weights. However, rather than renormalising the corresponding weights, as suggested in Zaidi et al. (2022), we reallocate the weight assigned to excluded architectures proportionally to the relative covariance between them and the ensemble members. Concretely, let $\{(\alpha_m, \omega_m)\}_{m=1}^M$ be the ensemble members and their optimised weights, and $\{(\alpha_l, \omega_l)\}_{l=1}^{N-M}$ be the excluded architectures and their optimised weights. The weights of the ensemble $\mathbf{w} \in \mathbb{R}^M$ are given by

$$\mathbf{w}_m = \omega_m + \sum_{l=1}^{N-M} \frac{k(\alpha_m, \alpha_l)}{\sum_{m=1}^M k(\alpha_m, \alpha_l)} \omega_l. \quad (4.18)$$

Algorithm 3 Posterior recombination.

$T \leftarrow \text{nystrom_test_functions}(K_{AA}, A)$ ▷ From Eq (4.11)
 $\mu \leftarrow \left[\frac{p(D|\alpha_n)p(\alpha_n)}{\hat{Z}} \right]_{n=1}^N$
 $\mathbf{w}, A_M \leftarrow \text{recombination}(T, \mu)$

Algorithm 4 Re-weighted stacking.

$\omega \leftarrow \text{argmin}_{\omega \in \Delta} \text{loss}(\sum_i \omega_i p(c | x, \alpha_n, D), D)$
 $I \leftarrow \text{select_top}(M, \omega)$ ▷ Select top M.
for m in I **do**
 $\mathbf{w}_m \leftarrow \text{reweight}(m, I, \omega, k(A, A))$ ▷ Eq (4.18).
end for

Our proposals can be combined to yield two possible NES algorithms. Both share the same candidate selection strategy that uses a WSABI-L surrogate model with the uncertainty sampling acquisition function to select the set of architectures to train (Algorithm 2). NES-BQ then uses posterior recombination (Algorithm 3) to select a subset of architectures from the candidate set to include in the ensemble, and choose their corresponding weights. NES-USS instead uses re-weighted stacking (Algorithm 4) to select, and weight, the ensemble members from the candidate set. Figure 4.2 is a schematic representation of these algorithms.

4.5 Experiments

We compare our proposed method to state-of-the-art baselines using the NATS-Bench benchmark (Dong, Liu et al. 2021). Specifically, we use the provided topology search space, which consists of cells with 4 nodes, 6 connections, and 5 possible operations (including “zeroise” which is equivalent to removing a connection) in a fixed macro-skeleton. The architecture weights are trained for 200 epochs on the CIFAR-10, CIFAR-100, and ImageNet16-120 (a smaller version of ImageNet with 16×16 pixel input images, and 120 classes) datasets. We compare ensemble performance as measured by test accuracy, test likelihood, and expected calibration error on the test set for a range of ensemble sizes.

We first compare the two variants of our algorithm – NES-BQ and NES-USS – with several baselines:

Random The ensemble is an evenly weighted combination of M architectures randomly sampled from the prior $p(\alpha)$ over the search space.

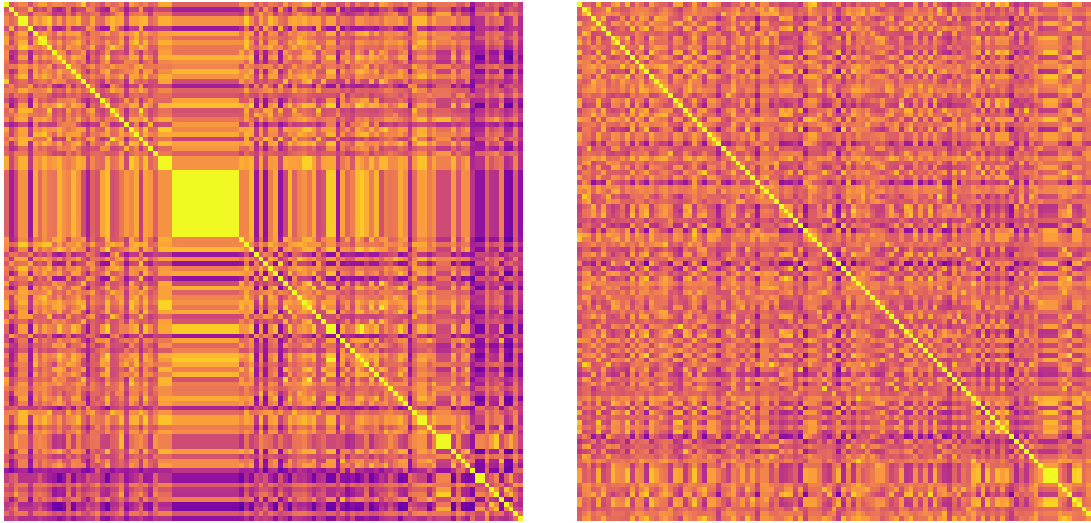
NES-RE The candidate set is selected using regularised evolution, and the ensemble members are chosen using beam search. The ensemble members are equally weighted.

NES-BS The posterior over architectures $p(\alpha | D)$ is approximated using a variational distribution. The ensemble is constructed by sampling M architectures from the variational distribution using Stein-Variational Gradient Descent.

We then conduct an ablation study to examine the relative contributions of our candidate selection and ensemble selection algorithms.

Tables 4.1 and 4.2 present the results on CIFAR-100 and ImageNet16-120 for a range of ensemble sizes. Whilst NES-RE matches or does slightly better than our proposals in terms of accuracy and LL on CIFAR-100, we find that both NES-USS and NES-BQ often perform better in terms of expected calibration error. NES-USS achieves the best performance on ImageNet16-120 in terms of LL across all ensemble sizes, is joint best with NES-RE in terms of accuracy, and often outperforms NES-RE in terms of ECE. The difference in the methods' relative performance between the two datasets is likely due to the nature of the architecture likelihood surfaces – for CIFAR-100 this has wider peaks, clustered closer together, as demonstrated in Figure 4.3. Interestingly, NES-BQ performs quite poorly for ensembles of size 3, but consistently achieves the best ECE for larger ensembles, for all datasets. This is at the cost of accuracy and LL, however, which are better than for NES-BS and Random, but worse than for NES-RE and NES-USS. Overall, these results suggest that:

- NES-BQ with a relatively large ensemble size is most appropriate if well calibrated uncertainty estimates are desired.
- NES-USS is best overall for ensembles over architecture likelihood surfaces with dispersed, narrow peaks, if accuracy and LL are the key performance metrics.
- NES-RE is best suited to architecture likelihood surfaces with broad peaks, if accuracy and LL are the key performance metrics.



(a) CIFAR-100

(b) ImageNet16-120

Figure 4.3: Visualisation of the (WL) covariance matrix for the 500 unique architectures with the highest likelihoods in the search space for each dataset, sorted by (a smoothed estimate, using a GP, of the) likelihood. The colour scale varies from 1 (yellow) to 0 (blue). We observe larger blocks of architectures within the top 500 that covary strongly for CIFAR-100 than for ImageNet16-120, which implies that the modes of the architecture likelihood surface are wider for CIFAR-100. This suggests that a more exploratory strategy will do better on ImageNet16-120, and a more exploitative strategy for CIFAR-100.

4.5.1 Ablation Study

Tables 4.3 and 4.4 shows the effect of the candidate selection algorithm. In all cases we use our variant of weighted stacking, described in Section 4.4.2, to select and weight the ensemble members. We compare Expected Improvement (EI) with a GP surrogate with a WL kernel, Uncertainty Sampling with a WSABI-L surrogate using a WL kernel (US), and Regularised Evolution (RE). We find that the US candidate set performs best for ImageNet16-120 in terms of accuracy and LL, but that the RE candidate set performs best for ECE on ImageNet16-120, and across all metrics for CIFAR-100. We expect that this is due to the difference in the shape of the architecture likelihood surface between the two datasets, as shown in Figure 4.3. The surface for ImageNet16-120 appears to have narrower modes than for CIFAR-100, which suggests that a more exploratory strategy, such as US, would be more

Algorithm	Accuracy	ECE	LL
Best Single	69.1	0.088	-5871
Ensemble Size 3			
Random	69.2 \pm 1.5	0.075 \pm 0.007	-5778 \pm 291.3
NES-RE	76.6 \pm 0.2	0.026 \pm 0.002	-4340 \pm 19.58
NES-BS	66.2 \pm 1.5	0.073 \pm 0.009	-6477 \pm 203.0
NES-BQ	71.9 \pm 0.8	0.075 \pm 0.025	-5259 \pm 300.9
NES-USS	76.6 \pm 0.2	0.021 \pm 0.001	-4417 \pm 35.85
Ensemble Size 5			
Random	72.2 \pm 0.9	0.111 \pm 0.009	-5304 \pm 180.9
NES-RE	78.2 \pm 0.1	0.042 \pm 0.002	-4002 \pm 17.11
NES-BS	65.9 \pm 1.5	0.073 \pm 0.009	-6481 \pm 208.7
NES-BQ	73.3 \pm 0.9	0.040 \pm 0.004	-4768 \pm 174.3
NES-USS	77.8 \pm 0.2	0.040 \pm 0.002	-4077 \pm 33.60
Ensemble Size 10			
Random	74.7 \pm 0.3	0.150 \pm 0.010	-5018 \pm 82.21
NES-RE	79.4 \pm 0.1	0.060 \pm 0.001	-3763 \pm 15.16
NES-BS	69.1 \pm 0.4	0.085 \pm 0.005	-6119 \pm 36.31
NES-BQ	75.5 \pm 0.9	0.037 \pm 0.002	-4309 \pm 172.6
NES-USS	78.6 \pm 0.2	0.059 \pm 0.001	-3843 \pm 22.71

Table 4.1: Test accuracy, expected calibration error (ECE), and log likelihood (LL) on CIFAR-100 for our proposals (NES-BQ and NES-USS) and baselines. For reference we also include the performance of the best architecture (measured by validation loss) on the test set (labelled Best Single). The numbers shown are means and standard error of the mean over 10 repeats. Where applicable, the candidate set selection method is initialised with 10 random architectures, and used to build a set of 150 architectures. We find that NES-RE performs best in terms of accuracy and LL. Particularly for larger ensembles, NES-BQ performs best in terms of ECE.

appropriate for ImageNet16-120 and a more exploitative strategy, such as RE, would be more appropriate for CIFAR-100. This is indeed what we observe.

Tables 4.5 and 4.6 shows the effect of the ensemble selection algorithm. In all cases we use uncertainty sampling with a WSABI-L surrogate to build the candidate set. We initialise with 10 architectures randomly selected from a uniform prior over the search space, and use the acquisition function to build a set of 150 architectures. We compare beam search (BS), weighted stacking (WS), recombination of the approximate posterior (PR), and re-weighted stacking (RS). We find that the stacking variants consistently perform best (with RS slightly improving upon WS) in terms

Algorithm	Accuracy	ECE	LL
Best Single	45.9	0.062	-6386
Ensemble Size 3			
Random	39.7 \pm 2.2	0.097 \pm 0.007	-7459 \pm 309.6
NES-RE	52.0 \pm 0.2	0.033 \pm 0.002	-5582 \pm 8.858
NES-BS	45.7 \pm 0.3	0.058 \pm 0.003	-6403 \pm 28.04
NES-BQ	46.7 \pm 2.3	0.052 \pm 0.021	-6347 \pm 480.5
NES-USS	52.2 \pm 0.1	0.029 \pm 0.001	-5543 \pm 10.87
Ensemble Size 5			
Random	42.7 \pm 1.5	0.129 \pm 0.008	-7135 \pm 216.1
NES-RE	53.4 \pm 0.2	0.051 \pm 0.001	-5404 \pm 12.59
NES-BS	45.7 \pm 0.3	0.058 \pm 0.003	-6403 \pm 28.04
NES-BQ	50.7 \pm 0.3	0.028 \pm 0.004	-5647 \pm 50.58
NES-USS	53.6 \pm 0.1	0.050 \pm 0.002	-5380 \pm 12.31
Ensemble Size 10			
Random	45.1 \pm 0.4	0.159 \pm 0.008	-6916 \pm 73.21
NES-RE	54.5 \pm 0.2	0.069 \pm 0.001	-5269 \pm 17.83
NES-BS	45.6 \pm 0.3	0.068 \pm 0.004	-6442 \pm 24.47
NES-BQ	52.3 \pm 0.3	0.018 \pm 0.001	-5412 \pm 22.96
NES-USS	54.7 \pm 0.1	0.072 \pm 0.001	-5262 \pm 9.964

Table 4.2: Test accuracy, expected calibration error (ECE), and log likelihood (LL) on ImageNet16-120 for our proposals (NES-BQ and NES-USS) and baselines. For reference we also include the performance of the best architecture (measured by validation loss) on the test set (labelled Best Single). The numbers shown are means and standard error of the mean over 10 repeats. Where applicable, the candidate set selection method is initialised with 10 random architectures, and used to build a set of 150 architectures. We see that NES-USS performs best across ensemble sizes in terms of LL, and joint best with NES-RE in terms of accuracy. Particularly for larger ensembles, NES-BQ performs best in terms of ECE.

of accuracy and LL, and PR in terms of ECE for larger datasets.

Algorithm	Accuracy	ECE	LL
Ensemble Size 3			
RE	77.1 ± 0.2	0.018 ± 0.001	-4385 ± 24.89
EI	76.1 ± 0.2	0.024 ± 0.001	-4472 ± 29.74
US	76.6 ± 0.2	0.021 ± 0.001	-4417 ± 35.85
Ensemble Size 5			
RE	78.5 ± 0.2	0.033 ± 0.001	-4013 ± 19.08
EI	77.4 ± 0.2	0.039 ± 0.001	-4126 ± 22.25
US	77.8 ± 0.2	0.040 ± 0.002	-4077 ± 33.60
Ensemble Size 10			
RE	79.4 ± 0.1	0.053 ± 0.002	-3759 ± 16.38
EI	78.2 ± 0.2	0.055 ± 0.002	-3889 ± 23.79
US	78.6 ± 0.2	0.059 ± 0.001	-3843 ± 22.71

Table 4.3: Test accuracy, expected calibration error, and log likelihood on CIFAR-100 for our candidate set selection method (US) and baselines. The numbers shown are means and standard error of the mean over 10 repeats. Each candidate set selection method is initialised with 10 random architectures, and used to build a set of 150 architectures. The ensemble is chosen and weighted using our variant of weighted stacking. We see that the RE candidate set performs best for CIFAR-100. We speculate that this is because the wide peaks of the likelihood surface for CIFAR-100 favour a more exploitative strategy.

Algorithm	Accuracy	ECE	LL
Ensemble Size 3			
RE	51.9 \pm 0.2	0.029 \pm 0.002	-5595 \pm 12.15
EI	51.4 \pm 0.2	0.034 \pm 0.002	-5632 \pm 11.91
US	52.2 \pm 0.1	0.029 \pm 0.001	-5543 \pm 10.87
Ensemble Size 5			
RE	53.3 \pm 0.2	0.043 \pm 0.002	-5417 \pm 12.90
EI	52.6 \pm 0.3	0.053 \pm 0.003	-5479 \pm 15.55
US	53.6 \pm 0.1	0.050 \pm 0.002	-5380 \pm 12.31
Ensemble Size 10			
RE	54.5 \pm 0.2	0.065 \pm 0.002	-5280 \pm 16.85
EI	53.4 \pm 0.2	0.071 \pm 0.002	-5368 \pm 19.47
US	54.7 \pm 0.1	0.072 \pm 0.001	-5262 \pm 9.964

Table 4.4: Test accuracy, expected calibration error, and log likelihood on ImageNet16-120 for our candidate set selection method (US) and baselines. The numbers shown are means and standard error of the mean over 10 repeats. Each candidate set selection method is initialised with 10 random architectures, and used to build a set of 150 architectures. The ensemble is chosen and weighted using our variant of weighted stacking. We see that the US candidate set performs best in terms of accuracy and LL across the ensemble sizes.

Algorithm	Accuracy	ECE	LL
Ensemble Size 3			
BS	75.2 \pm 0.2	0.030 \pm 0.002	-4500 \pm 41.04
WS	76.4 \pm 0.2	0.021 \pm 0.001	-4426 \pm 35.87
PR	71.9 \pm 0.8	0.075 \pm 0.025	-5259 \pm 300.9
RS	76.6 \pm 0.2	0.021 \pm 0.001	-4417 \pm 35.85
Ensemble Size 5			
BS	76.4 \pm 0.2	0.048 \pm 0.002	-4233 \pm 36.48
WS	77.7 \pm 0.2	0.036 \pm 0.002	-4088 \pm 34.13
PR	73.3 \pm 0.9	0.040 \pm 0.004	-4768 \pm 174.3
RS	77.8 \pm 0.2	0.040 \pm 0.002	-4077 \pm 33.60
Ensemble Size 10			
BS	76.9 \pm 0.3	0.063 \pm 0.001	-4079 \pm 50.29
WS	78.5 \pm 0.2	0.055 \pm 0.002	-3848 \pm 23.96
PR	75.5 \pm 0.9	0.037 \pm 0.002	-4309 \pm 172.6
RS	78.6 \pm 0.2	0.059 \pm 0.001	-3843 \pm 22.71

Table 4.5: Test accuracy, expected calibration error, and log likelihood on CIFAR-100 for Beam Search (BS), Weighted Stacking (WS), Posterior Recombination (PR), and Re-weighted Stacking (RS). The numbers shown are means and standard error of the mean over 10 repeats. The candidate set selection method is our method – Uncertainty Sampling with a WSABI-L surrogate – initialised with 10 random architectures, and used to build a set of 150 architectures. We see that the stacking variants consistently perform best for accuracy and LL, with RS slightly improving upon WS. For ECE, RS and WS perform well for small ensembles, but PR works best for larger ensembles.

Algorithm	Accuracy	ECE	LL
Ensemble Size 3			
BS	52.2 ± 0.1	0.036 ± 0.002	-5572 ± 13.17
WS	52.1 ± 0.1	0.029 ± 0.001	-5545 ± 10.57
PR	46.7 ± 2.3	0.052 ± 0.021	-6347 ± 480.5
RS	52.2 ± 0.1	0.029 ± 0.001	-5543 ± 10.87
Ensemble Size 5			
BS	76.4 ± 0.2	0.048 ± 0.002	-4233 ± 36.48
WS	77.7 ± 0.2	0.036 ± 0.002	-4088 ± 34.13
PR	73.3 ± 0.9	0.040 ± 0.004	-4768 ± 174.3
RS	77.8 ± 0.2	0.040 ± 0.002	-4077 ± 33.60
Ensemble Size 10			
BS	76.9 ± 0.3	0.063 ± 0.001	-4079 ± 50.29
WS	78.5 ± 0.2	0.055 ± 0.002	-3848 ± 23.96
PR	75.5 ± 0.9	0.037 ± 0.002	-4309 ± 172.6
RS	78.6 ± 0.2	0.059 ± 0.001	-3843 ± 22.71

Table 4.6: Test accuracy, expected calibration error, and log likelihood on ImageNet16-120 for Beam Search (BS), Weighted Stacking (WS), Posterior Recombination (PR), and Re-weighted Stacking (RS). The numbers shows are means and standard error of the mean over 10 repeats. The candidate set selection method is our method – Uncertainty Sampling with a WSABI-L surrogate – initialised with 10 random architectures, and used to build a set of 150 architectures. Again, we see that the stacking variants consistently perform best for accuracy and LL, but PR for ECE.

4.6 Discussion and Future Work

We have proposed a method for building ensembles of Neural Networks using the tools provided by Bayesian Quadrature. Specifically, by viewing ensembling as approximately performing marginalisation over architectures, we used the warped Bayesian Quadrature framework to select a candidate set of architectures to train. We then suggest two methods of constructing the ensemble based upon this candidate set: one based upon recombination of the approximate posterior over architectures (NES-BQ), and one based upon optimisation of the ensemble weights (NES-USS) using a validation set.

We demonstrate empirically that NES-BQ with a large ensemble size is the most performant method if expected calibration error of the test set is the key metric, as might be the case for systems which make critical decisions, such as self-driving vehicles. If, however, test set accuracy or log likelihood are the key metrics, we show that NES-USS is the best method for problems where the architecture likelihood surface has dispersed, narrow peaks, such as for ImageNet16-120.

We additionally conduct a range of ablation studies to independently investigate the contribution of our candidate set selection and ensemble selection methods compared to existing baselines. We find that the benefit of our candidate set selection algorithm depends on the nature of the likelihood surface, performing better for surfaces that require more exploration such as those with more dispersed peaks. This is because our proposal is based on active sampling with a GP which is inherently more exploratory than the existing baselines, which are based on evolutionary search. We also find that our ensemble selection methods consistently outperform alternatives. In particular, our proposal based on recombinations performs best for expected calibration error, and our proposal based on optimisation of the ensemble weights performs best for accuracy and log likelihood.

An important limitation of our work is that our proposals can be slightly outperformed by more exploitative alternatives when the architecture likelihood surface

has broad peaks, such as for CIFAR-100. A potential future direction concerns extensions of our method to larger architecture search spaces, over which modelling the architecture likelihood surface is likely to be challenging. Another possibility is to examine the effect of marginalising over architecture weights as well as over architectures.

This work concerns a general-purpose method, so its societal impact depends on the specific tasks that practitioners apply it to.

4.7 Supplement

4.7.1 Verification of Surrogate Quality

The quality of the GP posteriors, measured by RMSE and NLPD on a test set, is shown in Table 4.7. The test set is selected by ranking all the architectures in the search space by validation loss, and selecting every 25th architecture. This ensures that the test set contains architectures across the full range of performance. We build on the results of Ru et al. (2021), who showed that a GP with a WL kernel is able to model the architecture likelihood surface well. Our results show that WSABI-L is a consistently better model than an ordinary GP.

Model	CIFAR-100		ImageNet16-120	
	RMSE	NLPD	RMSE	NLPD
GP	6.165 ± 0.116	0.124 ± 0.013	9.610 ± 0.626	0.121 ± 0.012
WSABI-L	5.797 ± 0.043	-2.741 ± 0.095	4.078 ± 0.058	-3.437 ± 0.040

Table 4.7: The (normalised) RMSE and NLPD of a WSABI-L surrogate and a GP surrogate on the test sets.

4.7.2 Additional Experiments

Slimmable Network Search Space

We perform a study on a larger search space defined by a “slimmable network” (Yu et al. 2019), consisting of 614,625 architectures. Sub-networks or “slices” of this supernet constitute architectures within this search space. The architectures are structured as a chain of 7 blocks, each of which can have up to 4 layers. These sub-networks can be represented in a 28 dimensional ordinal space (with 4 options along each dimension). We use an RBF kernel with WSABI-L for Uncertainty Sampling with our method NES-USS, and compare to NES-RE. The results are shown in Table 4.8. We see that NES-USS consistently outperforms NES-RE in terms of log likelihood of the test set and, for CIFAR-100, in terms of expected calibration error as well.

Robustness to Dataset Shift

Previous work has provided evidence that ensembling of Neural Networks provides robustness to shifts in the underlying data distribution (Shu et al. 2022; Zaidi et al. 2022). However, these investigations have assumed the availability of a validation set from the shifted distribution, which we argue is unrealistic in practice. Instead, we examine the setting where only the test set is shifted, and the validation set is representative of the training set. We use the benchmark established by Hendrycks and Dietterich (2019) to generate shifted datasets by applying one of 30 corruption types to each image for CIFAR-10 and CIFAR-100. Each corruption type has a severity level on a 1 – 5 scale. Tables 4.9 and 4.10 show a comparison between NES-RE and NES-USS in this setting. We see that, whilst our proposal performs similarly in terms of accuracy, it produces ensembles that perform significantly better in terms of expected calibration error and test set log likelihood. This trend holds across corruption severity levels.

CIFAR-10			
Algorithm	Accuracy	ECE	LL
Ensemble Size 3			
NES-RE	93.8 \pm 0.0	0.029 \pm 0.001	-1165 \pm 5.602
NES-USS	93.7 \pm 0.1	0.030 \pm 0.000	-1152 \pm 5.215
Ensemble Size 5			
NES-RE	93.8 \pm 0.0	0.030 \pm 0.001	-1165 \pm 5.503
NES-USS	93.7 \pm 0.1	0.032 \pm 0.000	-1113 \pm 4.123
Ensemble Size 10			
NES-RE	93.8 \pm 0.0	0.030 \pm 0.001	-1159 \pm 5.959
NES-USS	93.8 \pm 0.0	0.031 \pm 0.000	-1098 \pm 3.752
CIFAR-100			
Algorithm	Accuracy	ECE	LL
Ensemble Size 3			
NES-RE	74.2 \pm 0.2	0.072 \pm 0.004	-5136 \pm 61.49
NES-USS	74.4 \pm 0.1	0.063 \pm 0.002	-5021 \pm 22.71
Ensemble Size 5			
NES-RE	74.3 \pm 0.2	0.071 \pm 0.004	-5134 \pm 60.72
NES-USS	74.5 \pm 0.1	0.055 \pm 0.002	-4897 \pm 25.66
Ensemble Size 10			
NES-RE	74.3 \pm 0.2	0.069 \pm 0.004	-5083 \pm 58.42
NES-USS	74.7 \pm 0.1	0.045 \pm 0.001	-4766 \pm 15.89

Table 4.8: Test accuracy, expected calibration error (ECE), and log likelihood (LL) on CIFAR-10 and CIFAR-100 for NES-USS (our proposal) and NES-RE (the strongest baseline).

Severity Level 1			
Algorithm	Accuracy	ECE	LL
Ensemble Size 3			
NES-RE	86.2 \pm 0.0	0.046 \pm 0.001	-59259.6 \pm 595.907
NES-USS	86.3 \pm 0.1	0.036 \pm 0.001	-54283.4 \pm 642.383
Ensemble Size 5			
NES-RE	86.3 \pm 0.0	0.046 \pm 0.001	-59178.3 \pm 851.719
NES-USS	86.2 \pm 0.1	0.032 \pm 0.001	-52173.6 \pm 202.350
Ensemble Size 10			
NES-RE	86.3 \pm 0.0	0.043 \pm 0.001	-57010.4 \pm 722.311
NES-USS	86.2 \pm 0.1	0.029 \pm 0.001	-50504.6 \pm 443.984
Severity Level 3			
Algorithm	Accuracy	ECE	LL
Ensemble Size 3			
NES-RE	73.2 \pm 0.1	0.147 \pm 0.002	-133205 \pm 1537.15
NES-USS	73.3 \pm 0.1	0.131 \pm 0.002	-123113 \pm 1498.55
Ensemble Size 5			
NES-RE	73.2 \pm 0.1	0.148 \pm 0.002	-133239 \pm 1904.50
NES-USS	73.2 \pm 0.1	0.125 \pm 0.001	-118756 \pm 614.899
Ensemble Size 10			
NES-RE	73.2 \pm 0.1	0.143 \pm 0.002	-128663 \pm 1664.07
NES-USS	73.4 \pm 0.1	0.120 \pm 0.002	-114613 \pm 1247.44
Severity Level 5			
Algorithm	Accuracy	ECE	LL
Ensemble Size 3			
NES-RE	55.5 \pm 0.1	0.285 \pm 0.002	-239927 \pm 2187.24
NES-USS	55.7 \pm 0.1	0.265 \pm 0.003	-226433 \pm 2523.79
Ensemble Size 5			
NES-RE	55.5 \pm 0.1	0.286 \pm 0.003	-240154 \pm 2835.89
NES-USS	55.5 \pm 0.1	0.260 \pm 0.002	-220196 \pm 1198.20
Ensemble Size 10			
NES-RE	55.5 \pm 0.1	0.279 \pm 0.002	-233441 \pm 2474.23
NES-USS	55.6 \pm 0.1	0.254 \pm 0.003	-214126 \pm 2030.20

Table 4.9: Test accuracy, expected calibration error (ECE), and log likelihood (LL) on CIFAR-10 for NES-RE (the strongest baseline), and NES-USS (our strongest proposal).

Severity Level 1			
Algorithm	Accuracy	ECE	LL
Ensemble Size 3			
NES-RE	62.4 ± 0.1	0.151 ± 0.004	-169235 ± 1632.43
NES-USS	62.5 ± 0.1	0.093 ± 0.002	-149022 ± 364.575
Ensemble Size 5			
NES-RE	62.4 ± 0.1	0.155 ± 0.003	-169999 ± 1553.96
NES-USS	62.6 ± 0.1	0.103 ± 0.004	-152466 ± 1249.97
Ensemble Size 10			
NES-RE	62.5 ± 0.1	0.145 ± 0.002	-164816 ± 975.790
NES-USS	62.5 ± 0.1	0.093 ± 0.002	-149022 ± 364.575
Severity Level 3			
Algorithm	Accuracy	ECE	LL
Ensemble Size 3			
NES-RE	49.2 ± 0.1	0.235 ± 0.005	-270710 ± 2628.75462
NES-USS	49.5 ± 0.1	0.193 ± 0.007	-250337 ± 3304.95
Ensemble Size 5			
NES-RE	49.2 ± 0.1	0.239 ± 0.004	-272004 ± 2482.09961
NES-USS	49.6 ± 0.1	0.175 ± 0.005	-241407 ± 2438.78
Ensemble Size 10			
NES-RE	49.3 ± 0.1	0.227 ± 0.003	-263639 ± 1596.01214
NES-USS	49.6 ± 0.1	0.163 ± 0.003	-235152 ± 881.120
Severity Level 5			
Algorithm	Accuracy	ECE	LL
Ensemble Size 3			
NES-RE	34.0 ± 0.1	0.339 ± 0.005	-415182 ± 3710.25
NES-USS	34.2 ± 0.1	0.291 ± 0.008	-385063 ± 5355.88
Ensemble Size 5			
NES-RE	34.0 ± 0.1	0.344 ± 0.004	-417110 ± 3476.95
NES-USS	34.3 ± 0.1	0.270 ± 0.006	-371575 ± 4083.67
Ensemble Size 10			
NES-RE	34.1 ± 0.1	0.331 ± 0.003	-405068 ± 2327.88
NES-USS	34.4 ± 0.1	0.257 ± 0.003	-361876 ± 1666.07

Table 4.10: Test accuracy, expected calibration error (ECE), and log likelihood (LL) on CIFAR-100 for NES-RE (the strongest baseline), and NES-USS (our strongest proposal).


Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).


Title of Paper	Bayesian Quadrature for Neural Ensemble Search
Publication Status	<input type="checkbox"/> Published <input type="checkbox"/> Accepted for Publication <input checked="" type="checkbox"/> Submitted for Publication <input type="checkbox"/> Unpublished and unsubmitted work written in a manuscript style
Publication Details	Hamid, S., Wan, X. Jørgensen, M., Ru B. & Osborne M. (2023). Bayesian Quadrature for Neural Ensemble Search. In Proceedings of the 26 th International Conference on Artificial Intelligence and Statistics

Student Confirmation

Student Name:	Saad Hamid		
Contribution to the Paper	<ul style="list-style-type: none">• Joint formulation of initial research direction.• Formulation of the details of the methodology.• Joint discussion of experimental design.• Majority of coding for the experiments.• Majority of the paper write-up.		
Signature		Date	11/01/2023

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Prof Michael A. Osborne			
Supervisor comments I certify that the candidate made a substantial contribution to the publication, just as described above.			
Signature		Date	11 January 2023

This completed form should be included in the thesis, at the end of the relevant chapter.

Chapter 5

Scalable Bayesian Quadrature with Gaussian Process Approximations

Contents

5.1	Abstract	90
5.2	Introduction	91
5.3	Background	92
5.3.1	Probabilistic Integration	92
5.3.2	Variational Gaussian Processes and the VISH Model	94
5.3.3	Bézier Gaussian Process	96
5.4	Methods	97
5.4.1	Using VISH for Active Learning of the Integrand	97
5.4.2	Bayesian Quadrature with the Bézier Log-Normal Process	99
5.5	Related Work	101
5.6	Experiments	102
5.7	Discussion	106
5.8	Supplement	106
5.8.1	Mixed Inducing Variables for VISH	106

Preface

Bayesian Quadrature is a powerful tool for numerically approximating integrals of integrands that are expensive to evaluate. The previous chapters sought to design Bayesian Quadrature schemes for model selection in two expressive and widely-used

model classes: Gaussian Processes and Neural Networks. In this chapter we will be considering the problem of extending Bayesian Quadrature to higher dimensional spaces, in particular for non-negative integrands of the sort encountered in machine learning.

The fundamental challenge of integration in high dimensional spaces is that the volume of the integration domain increases exponentially in the number of dimensions. Additionally, for marginalisation integrals, the likelihood functions of machine learning models typically have very narrow peaks and high dynamic ranges. As most of the mass of the integral is concentrated under these peaks, locating them is crucial for accurately estimating the integral. To achieve this in high dimensional spaces requires a large number of integrand observations, rendering ordinary GP surrogates unsuitable due to their cubic computational complexity. This chapter will explore extensions for BQ of two recently proposed scalable approximations to GPs:

VISH which is a Variational Gaussian Process with inter-domain inducing features that correspond to the spherical harmonics (Dutordoir, Durrande and Hensman 2020).

Bézier GP which is defined as a Bézier curve with Gaussians over the control points (Jørgensen and Osborne 2022).

Both methods scale linearly in the number of observations, and are therefore amenable to the high-dimensional integration problems of interest in this chapter.

5.1 Abstract

Scaling Bayesian Quadrature (BQ) is challenging because of the cubic computational complexity of Gaussian Process Regression. In this paper we explore the use of recently proposed scalable approximations to Gaussian Processes for scaling BQ. We first investigate using a Variational Gaussian Process with inducing variables

that are projections of the function onto the spherical harmonics to build up the design set. The posterior over the integral is calculated using a GP whose kernel matrices are approximately inverted using Black-Box Matrix-Matrix Multiplication, a method based on conjugate gradient descent. We then show how to use a Log-Bézier Gaussian Process, a Bézier curve with Log-Normal distributions over the control points, to perform scalable BQ. We empirically compare these proposals to existing methods for high-dimensional integration.

5.2 Introduction

Intractable high-dimensional integrals of non-negative integrands are a key challenge in machine learning. Typically, these are marginalisation integrals, required either to compute a model evidence, or characterise a posterior or posterior predictive distribution (Murphy 2022).

A promising approach to the numerical approximation such integrals is Bayesian Quadrature (BQ), also called probabilistic integration (PI), which approximates the integrand with a probabilistic model, typically a Gaussian Process (Diaconis 1988; Minka 2000; O’Hagan 1991). This surrogate then induces a posterior over the value of the integral. Additionally, the surrogate can be used to decision-theoretically select new evaluations of the integrand to improve the estimate of the integral (Osborne, Duvenaud et al. 2012) – a process called adaptive Bayesian Quadrature. However, due to the cubic (in the number of observations) time complexity of Gaussian Process regression (Rasmussen and Williams 2006), probabilistic integration is often restricted to relatively low dimensional spaces. Intuitively, this is because more observations are required to “cover” a high-dimensional space.

Recent advances in sparse variational approximations of Gaussian Processes (Leibfried et al. 2021; Titsias 2009) have yielded models for which the time complexity of inference scales significantly better in the number of observations. Addi-

tionally, there exist theoretical guarantees regarding the performance of such models (Burt, Rasmussen and Wilk 2020). In this paper, we investigate the use for PI of the recently proposed VISH model, a variational Gaussian Process defined on the hypersphere so that the spherical harmonics can be used as inducing features (Dutordoir, Durrande and Hensman 2020). We also investigate a modification of the recently proposed Bézier Gaussian Process (Jørgensen and Osborne 2022), a GP defined using a Bézier curve, for BQ. Importantly, both models scale linearly in the number of observations, significantly reducing the cost of adaptive Bayesian Quadrature as the surrogate model must be fit to the integrand every iteration.

We make the following contributions:

- We explore the use of the VISH model as a cheap surrogate for building up the design set – the set of locations at which the integrand is observed – for probabilistic integration.
- We propose the Bézier Log-Normal Process, and show how it can be used to approximate a posterior over the integral for non-negative integrands. We also investigate its use for adaptive Bayesian Quadrature.
- We undertake an empirical study to compare the performance of our proposals against state-of-the-art baselines for synthetic and real-world marginalisation integrals.

5.3 Background

5.3.1 Probabilistic Integration

Probabilistic Integration, also referred to as Bayesian Quadrature, is a class of methods to infer a distribution over $Z = \int f(x)d\pi(x)$, based upon potentially noisy observations of $f(\cdot)$, Y , at locations X . A probabilistic surrogate, typically a Gaussian Process, is used to model the integrand, $f(x) \mid D \sim \mathcal{GP}(m(x), k(x, x'))$, where

$D = (X, Y)$. As Gaussians are closed under bounded linear transformations, this induces a Gaussian distribution over Z whose moments are given by $\int m(x)\pi(x)dx$ and $\int \int k(x, x')\pi(x)\pi(x')dxdx'$ (Diaconis 1988; Minka 2000; O’Hagan 1991). New observations of the integrand are then chosen by maximising an acquisition function, a process we refer to as active sampling (Chai, Ton et al. 2019; Gunter et al. 2014; Osborne, Duvenaud et al. 2012).

It has been noted that active sampling for probabilistic integration is useful only if the hypothesis space of the surrogate model is imbalanced or non-convex (Kanagawa, Sriperumbudur and Fukumizu 2020; Novak 2016). When the hypothesis space is the unit ball of a Reproducing Kernel Hilbert Space, the asymptotic optimal rate can be achieved without an adaptive sampling strategy. As we focus on non-negative integrands, and explicitly incorporate this restriction by modelling the integrand in a warped space (Chai and Garnett 2019; Gunter et al. 2014; Osborne, Duvenaud et al. 2012), the hypothesis space is imbalanced. Therefore, adaptively selecting the design set can improve the rate of convergence.

The square-root transform (Gunter et al. 2014) and log transform (Chai, Ton et al. 2019; Osborne, Duvenaud et al. 2012) have been studied as warpings for probabilistic integration of non-negative integrands. The integrand is modelled in the warped space, inducing a posterior over the unwarped integrand which is a Chi-Squared (for the square-root transform) or Log-Normal (for the log transform) distribution. These posteriors are approximated with a GP by either moment-matching or by linearisation with a first-order Taylor expansion. In particular, in this work we will focus on the MMLT model, which uses a moment matching for the log transform $g(x) = \log(f(x))$, where f is the integrand. Placing a GP prior over g , and conditioning on the observations (in log space) results in a GP $g \sim \mathcal{GP}(m_g(x), k_g(x, x'))$. The distribution over f is therefore Log-Normal, and we can moment match a GP to its first two moments to approximate $f \sim \mathcal{GP}(\mu(x), \Sigma(x, x'))$

where

$$\mu(x) = \exp(m_{\mathbf{g}}(x) + 1/2k_{\mathbf{g}}(x, x)) \quad (5.1)$$

$$\Sigma(x, x') = m_{\mathbf{g}}(x) \left(\exp(k_{\mathbf{g}}(x, x')) - 1 \right) m_{\mathbf{g}}(x'). \quad (5.2)$$

Unfortunately, the moments of the Gaussian distribution induced over Z are no longer analytic available, and must be approximated by QMC sampling.

The computational cost of Bayesian Quadrature is $\mathcal{O}(n^3)$ where n is the number of observations of the integrand. For adaptive BQ the cost $\mathcal{O}(n^3 + (n-1)^3 + \dots + (n-i)^3)$ where i is the number of initial observations. In practice, this cost is not justifiable unless the integrand is very expensive to query. Additionally, it is not feasible for high-dimensional integrands, as a large number of observations are required to explore high-dimensional spaces.

5.3.2 Variational Gaussian Processes and the VISH Model

Sparse Variational Gaussian Processes are a variational approximation (Blei, Kucukelbir and McAuliffe 2018) for inference with Gaussian Processes (Titsias 2009). Following (Leibfried et al. 2021), we define the inducing variables as u . These inducing variable are either pseudo-observations or linear transforms of the function. The joint Gaussian over f and u then has covariance structure

$$\begin{bmatrix} f(\cdot) \\ u \end{bmatrix} \sim \mathcal{GP} \left(\begin{bmatrix} \mu(\cdot) \\ \mu_u \end{bmatrix}, \begin{bmatrix} \Sigma(\cdot, \cdot) & \Sigma_{u\cdot} \\ \Sigma_{u\cdot} & \Sigma_{uu} \end{bmatrix} \right), \quad (5.3)$$

and the posterior over $f(\cdot)$ conditioned on u is

$$p(f(\cdot) | u) = \mathcal{GP}(\mu(\cdot) + \Sigma_{u\cdot} \Sigma_{uu}^{-1} (u - \mu_u), \Sigma(\cdot, \cdot) - \Sigma_{u\cdot} \Sigma_{uu}^{-1} \Sigma_{u\cdot}). \quad (5.4)$$

We then place an additional variational distribution over u , $q(u) \sim \mathcal{N}(\nu_u, S_{uu})$. Marginalising over u with $q(u)$ then results in

$$q(f(\cdot)) = \mathcal{GP}(\mu(\cdot) + \Sigma_u \Sigma_{uu}^{-1}(\nu_u - \mu_u), \Sigma(\cdot, \cdot) - \Sigma_u \Sigma_{uu}^{-1}(\Sigma_{uu} - S_{uu})\Sigma_{uu}^{-1}\Sigma_u). \quad (5.5)$$

We note that, for regression with Gaussian noise, the optimal moments of $q(u)$ are analytically available (Titsias 2009),

$$\nu_s = \sigma^{-2}\Sigma_{uu}(\Sigma_{uu} + \sigma^{-2}\Sigma_{ux}\Sigma_{xu})^{-1}\Sigma_{ux}y \quad \text{and} \quad (5.6)$$

$$S_{uu} = \Sigma_{uu}(\Sigma_{uu} + \sigma^{-2}\Sigma_{ux}\Sigma_{xu})^{-1}\Sigma_{uu}. \quad (5.7)$$

The VISH model is a variational GP defined on the sphere, with the inducing variables as projections of the function onto the spherical harmonics (Dutordoir, Durrande and Hensman 2020). The spherical harmonics are a set of orthogonal basis functions on the hypersphere $\{\phi_{l,k}\}_{l \in \mathbb{N}, k \in \mathcal{K}_l}$, where \mathcal{K}_l is an integer between 1 and the number of spherical harmonics, N_l^d for a given level l (Dai and Xu 2013). Zonal kernels are a class of kernels defined on the hypersphere that are invariant to rotations. They can be decomposed, by Mercer's theorem, in terms of the spherical harmonics

$$\Sigma(x_s, x'_s) = \sum_{l=0}^{\infty} \sum_{k=1}^{N_l^d} \lambda_{l,k} \phi_{l,k}(x_s) \phi_{l,k}(x'_s). \quad (5.8)$$

Using the RKHS inner product (of a zonal kernel) to define the inducing variables, $u_{l,k} = \langle f(\cdot), \phi_{l,k}(\cdot) \rangle_{\mathcal{H}}$, then leads to covariances of the form

$$\Sigma(u_{l,k}, f(x)) = \phi_{l,k}(x) \quad (5.9)$$

$$\Sigma(u_{l,k}, u_{l',k'}) = \frac{\delta_{l,l'} \delta_{k,k'}}{\lambda_{l,k}} \quad (5.10)$$

The orthogonality of the spherical harmonics means that the K_{uu} matrix is diagonal, allowing for very cheap inference. Note that data is mapped from \mathbb{R}^d onto the

sphere in \mathbb{S}^{d+1} by first appending a scalar bias (so that $x_b = [x, b]$), and applying the transform $(x_s, y_s) = (x_b/\|x_b\|, y/\|x_b\|)$. This construction is inspired by neural networks with ReLU activation functions. As the ReLU, σ , is homogeneous the computation for a single layer can be expressed $\sigma(w^T x_b) = \|w\| \|x_b\| \sigma(\cos(\theta))$ where θ is the angle between the weights w and the input x_b . In the limit of infinite width, the inner product between outputs of the network are given by the arc-cosine kernel (Cho and Saul 2009). Additionally, many architectures – with a prior over weights and as the width tends to infinity – are equivalent to Gaussian Processes (Garriga-Alonso, Rasmussen and Aitchison 2019; Lee et al. 2018; Matthews et al. 2018; Neal 1996; Yang 2019). For fully connected networks, it has been shown, under assumptions about the data distribution, that the corresponding kernels (called Neural Tangent Kernels (Jacot, Gabriel and Hongler 2018)), are zonal (Belfer et al. 2021; Bietti and Mairal 2019).

5.3.3 Bézier Gaussian Process

A Bézier GP is a scalable GP based on Bézier curves (Jørgensen and Osborne 2022). Bézier curves are parametric models whose basis functions are the Bernstein polynomials. The i^{th} Bernstein polynomial of order ν is

$$B_i^\nu(x) = \frac{\nu!}{i!(\nu-i)!} x^i (1-x)^{\nu-i}, \quad x \in [0, 1]. \quad (5.11)$$

A Bézier curve on a multidimensional input $\mathbf{x} = (x_1, \dots, x_d) \in [0, 1]^d$ is then defined

$$f(\mathbf{x}) = \sum_{i_1=0}^{\nu_1} \dots \sum_{i_d=0}^{\nu_d} B_{i_1}^{\nu_1}(x_1) \dots B_{i_d}^{\nu_d}(x_d) \mathbf{p}_{i_1, \dots, i_d}, \quad (5.12)$$

where each of the $\mathbf{p}_{i_1, \dots, i_d} \in \mathbb{R}^d$ are referred to as *control points*. A Bézier GP is defined by placing independent Gaussians over each of the control points $\mathbf{p}_{i_1, \dots, i_d} \sim$

$\mathcal{N}(\boldsymbol{\mu}_{i_1, \dots, i_d}, \boldsymbol{\Sigma}_{i_1, \dots, i_d})$. The moments of the process are given by

$$\mathbb{E}(f(\mathbf{x})) = \sum_{i_1=0}^{\nu_1} \cdots \sum_{i_d=0}^{\nu_d} B_{i_1}^{\nu_1}(x_1) \cdots B_{i_d}^{\nu_d}(x_d) \boldsymbol{\mu}_{i_1, \dots, i_d}, \quad (5.13)$$

$$\text{Var}(f(\mathbf{x}), f(\mathbf{z})) = \sum_{i_1=0}^{\nu_1} \cdots \sum_{i_d=0}^{\nu_d} B_{i_1}^{\nu_1}(x_1) \cdots B_{i_d}^{\nu_d}(x_d) \boldsymbol{\Sigma}_{i_1, \dots, i_d} B_{i_1}^{\nu_1}(z_1) \cdots B_{i_d}^{\nu_d}(z_d). \quad (5.14)$$

Inference in this model is performed using Variational inference, by approximating the posterior over control points using independent Gaussians $q(\mathbf{p}_{i_1, \dots, i_d}) = \mathcal{N}(\hat{\boldsymbol{\mu}}_{i_1, \dots, i_d}, \hat{\boldsymbol{\Sigma}}_{i_1, \dots, i_d})$. Assuming additive Gaussian noise, the ELBO is analytically available. The key computational challenge is that the number of control points, and therefore the number of summands in Equations 5.13 and 5.14 increases exponentially in the number of dimensions. This is handled by parameterising $\boldsymbol{\mu}_{i_1, \dots, i_d} = \prod_{\gamma=1}^d w_{i_{\gamma-1}, i_{\gamma}, \gamma}$, a parameterisation known as a Bézier Buttress. This allows for the computation of the moments of $f(\mathbf{x})$ with d matrix multiplications. $\boldsymbol{\Sigma}_{i_1, \dots, i_d}$ can be parameterised similarly.

5.4 Methods

We investigate two possibilities for scaling Bayesian Quadrature: one based upon the VISH model, and one based upon the Bézier GP.

5.4.1 Using VISH for Active Learning of the Integrand

The VISH model has a computational complexity that is linear in the number of observations, so it can be used as a cheaper alternative for modelling the integrand. Unfortunately, the moments of the induced posterior over the integral are not analytically available. (This is due to the warping used to model the non-negativity of the integrand. Computing the moments would require integrating exponentials of sums of spherical harmonics.) However, it is not required that these be available for the most commonly used acquisition function for BQ – uncertainty sampling.

Algorithm 5 Pseudocode for VISH-PI.

obtain initial samples and integrand observations (X, Y)
 $(X_s, G_s) \leftarrow (X/\|X_b\|, g(Y/\|X_b\|))$ \triangleright Map onto hypersphere, and warp
fit VISH to (X_s, G_s)
while $i \geq 0$ **do** \triangleright i is remaining evaluation budget
 $\hat{x} = \operatorname{argmax}_x \alpha(x)$ \triangleright Optimise acquisition function
 append \hat{x}, \hat{y} to (X, Y) \triangleright \hat{y} is integrand observation at \hat{x}
 $(X_s, G_s) \leftarrow (X/\|X_b\|, g(Y/\|X_b\|))$
 re-optimize VISH
 $i \leftarrow i - 1$
end while
fit GP to (X, Y) .
infer posterior moments of Z with GP model.

We, therefore, suggest that the VISH model be used as a cheap surrogate for active learning of the integrand – i.e. to build up the design set only. The final estimate of the integral is then computed using an ordinary GP with this design set. The key advantage of this approach is the reduced cost per iteration, as it is far cheaper to optimise the VISH model’s hyperparameters and compute its posterior moments, compared to an ordinary GP. This difference becomes significant as the number of integrand observations increases. Such a setting arises for high-dimensional integration problems where a large number of observations are required to “cover” the space, and where the cost of making an observation does not make this prohibitively expensive. The strategy that we explore in such a setting is initialisation with a large set drawn using a low-discrepancy sequence to cover the space well; followed by active sampling with VISH to ensure that the modes of integrand are well explored.

Concretely, defining $D = (X, Y)$, $X \in \mathbb{R}^d$, $Y \in \mathbb{R}_+$, as the set of observations of the integrand, $f(x)$, we append a bias, b to each location in X , and project onto (half of) the hypersphere, as described in Section 5.3.2, to yield a dataset $D_s = (X_s, Y_s)$. We then apply the square-root transform to each element in Y_s to obtain the warped observations G_s . We place the VISH model over these warped observations, and minimise the ELBO to optimise the model’s hyperparameters,

including the bias, b . Denoting the posterior moments of the VISH model as $\tilde{\mu}(x_s)$ and $\tilde{\Sigma}(x_s, x'_s)$, where $x_s \in \mathbb{S}^{d+1}$, the posterior over the mapping of the integrand onto the sphere can be approximated by a GP with moments given by Equations (5.1) and (5.2). The projection back into \mathbb{R}^d is also a linear transform, as is multiplication by the prior. Therefore, the uncertainty sampling acquisition function under this model has the form

$$\alpha(x) = \tilde{\Sigma}(x_s, x_s) \tilde{\mu}(x_s)^2 \|x_b\|^2 \pi(x)^2 \quad (5.15)$$

where $x_b = [x, b]$, and $x_s = x/\|x_b\|$ as before.

In the setting of marginalisation integrals local structure is important, as the integrand often has narrow peaks. To address this challenge we explore using mixed inducing variables – a combination of pseudo-input and spherical harmonic inducing variables. This is explored in the supplement, Section 5.8.1.

A challenge remains, however, and that is performing inference in an ordinary GP following active learning of the integrand with VISH-PI. The resulting design set contains a large number of observations, so inverting the covariance matrix via a Cholesky decomposition is infeasible. We resort instead to a recently proposed variant of conjugate-gradient descent, Black-Box Matrix-Matrix Multiplication, which approximates this inversion in $\mathcal{O}(n^2)$ time, where n is the number of observations (Gardner et al. 2018).

5.4.2 Bayesian Quadrature with the Bézier Log-Normal Process

The Bézier Gaussian Process also has a linear computational complexity in the number of observations, and is therefore also a promising surrogate model for scaling Bayesian Quadrature. As the process is defined only over the unit hypercube, we restrict our attention to integrating a uniform prior over this domain. Alternative priors can be handled by modelling the product of the likelihood and the prior with

the surrogate model. As we are motivated by computing marginalisation integrals we propose several alterations to the model to incorporate salient prior information.

Firstly, we modify the the basis functions to make them more suitable for the “peaky” nature of the integrands typically encountered when marginalising over the parameters of machine learning models. To achieve this we introduce an additional the γ -Bernstein polynomials:

$$\mathfrak{B}_i^\nu(x, \gamma) : [0, 1] \times (0, \infty) \mapsto C (B_i^\nu(x))^\gamma, \quad (5.16)$$

where C is a constant only dependent on γ , ν , and i . It is given by,

$$C = \left\{ \nu^{-\nu} i^i (\nu - i)^{(\nu-i)} \binom{\nu}{i} \right\}^{-\gamma+1}, \quad (5.17)$$

and its purpose is to ensure that the maximum value of B_i^ν and \mathfrak{B}_i^ν is the same for any value of γ . This has the effect of maintaining the same function value range, but making the new basis functions peakier. Note that $\mathfrak{B}_i^\nu(\cdot, 1) = B_i^\nu(\cdot)$.

Modelling the integrand f with a Bézier Process induces a distribution over the integral Z whose expectation is

$$\begin{aligned} \mathbb{E}[Z] &= \mathbb{E} \left[\int_{[0,1]^d} \sum_{i_1=0}^{\nu_1} \cdots \sum_{i_d=0}^{\nu_d} B_{i_1}^{\nu_1}(x_1) \cdots B_{i_d}^{\nu_d}(x_d) \mathbf{p}_{i_1, \dots, i_d} d\mathbf{x} \right] \\ &= \sum_{i_1=0}^{\nu_1} \cdots \sum_{i_d=0}^{\nu_d} \mathbb{E} \left[\int_{[0,1]^d} B_{i_1}^{\nu_1}(x_1) \cdots B_{i_d}^{\nu_d}(x_d) \mathbf{p}_{i_1, \dots, i_d} d\mathbf{x} \right] \\ &= \sum_{i_1=0}^{\nu_1} \cdots \sum_{i_d=0}^{\nu_d} \int_{[0,1]^d} B_{i_1}^{\nu_1}(x_1) \cdots B_{i_d}^{\nu_d}(x_d) d\mathbf{x} \mathbb{E}[\mathbf{p}_{i_1, \dots, i_d}] \\ &= \frac{1}{(\nu_1 + 1) \cdots (\nu_d + 1)} \sum_{i_1=0}^{\nu_1} \cdots \sum_{i_d=0}^{\nu_d} \mathbb{E}[\mathbf{p}_{i_1, \dots, i_d}]. \end{aligned} \quad (5.18)$$

We are able to interchange the integral, expectation and sums because of Fubini’s theorem Fubini 1907, and the last equality follows from the fact that $\int_{[0,1]} B_i^\nu(x) dx =$

$(\nu + 1)^{-1}$ for all $i = 0$. A similar argument provides the variance

$$\text{Var}(Z) = \frac{1}{((\nu_1 + 1) \dots (\nu_d + 1))^2} \sum_{i_1=0}^{\nu_1} \dots \sum_{i_d=0}^{\nu_d} \text{Var}(\mathbf{p}_{i_1, \dots, i_d}). \quad (5.19)$$

Note that Equations (5.18) and (5.19) do not require distributional assumptions besides finite second moments of all control points. This provides a high degree of flexibility to incorporate prior information into the stochastic process defined by the control points. In particular, we incorporate the non-negativity of the integrand by placing Log-Normal distributions over the control points $\mathbf{p}_{i_1, \dots, i_d} \sim \mathcal{LN}(\boldsymbol{\mu}_{i_1, \dots, i_d}, \boldsymbol{\Sigma}_{i_1, \dots, i_d})$.

Note that uncertainty sampling with this model would result in samples always being selected at the control points. This is because these are locally the points with maximum variance. We argue that an acquisition function based on this model provides only a low-resolution indication of where to evaluate the integrand next. We, therefore, sample from the Bernstein polynomial corresponding to the control point which maximises the acquisition function.

5.5 Related Work

Several variants of Probabilistic Integration have been proposed in the literature, some of which build the design set cheaply, and some of which utilise cheaper surrogate models. Bayesian Monte Carlo (BMC) (Rasmussen and Ghahramani 2003) and Bayesian Quasi-Monte Carlo (BQMC) (Briol, Oates, Girolami, Osborne and Sejdinovic 2015) suggest simple MC and QMC sampling under the prior to build up the design set for PI, but adaptive strategies have outperformed these for non-negative integrands. The Random Fourier Feature approximation has been investigated in the context of PI (Briol, Oates, Girolami and Osborne 2015), but was found to perform poorly unless a very large number of Fourier features are used. Bayesian Additive Regression Trees have also been suggested as an alternative to Gaussian

Processes as surrogate models for Probabilistic Integration (Zhu et al. 2020), and these scale favourably but perform worse than GPs for smooth integrands.

5.6 Experiments

We empirically evaluate our proposals on a range of synthetic likelihoods and model evidence integrals for models with real-world data. Specifically, the problem settings are:

GMM- d A d -dimensional synthetic likelihood defined using GMM, integrated against a uniform prior over the unit hypercube. Following Gunter et al. (2014), we draw a random integer K from the integers 5–14, which defines the number of mixture components. The mixture means are randomly sampled from the inner quarter of the domain. The covariances are diagonal, with each variance randomly drawn between the range $[0.21, 0.29]$.

cancer Gaussian Process regression with an ARD Matern $1/2$ kernel on the UCI Breast Cancer Prognosis dataset. The task is to predict time to recurrence using 30 attributes. This is a 32-dimensional marginalisation problem over the kernel hyperparameters.

housing Gaussian Process regression with an ARD Matern $1/2$ kernel on the UCI Boston Housing dataset. The task is to predict the median value of owner-occupied homes using 13 attributes. This results in a 15-dimensional marginalisation problem.

ionosphere Bayesian logistic regression with the UCI Ionosphere dataset, where the task is to classify radar returns as “good” or “bad” from the ionosphere. To marginalise over the weights of this model is a 34-dimensional integration problem.

sonar Bayesian logistic regression with the UCI Connectionist Bench dataset. The task is to classify sonar signals as those that bounced off a metal cylinder, and those that bounced off a cylindrical rock. Marginalising over the weights of the model requires a 60-dimensional integration.

Table 5.1 compares the performance of our proposals against several baselines. VISH-PI and BLNBQ are our proposed methods. We evaluate VISH-PI for integrands up to 8-dimensions due to the limitations of available implementations. (These are not limitations inherent to the model – they are due to the fact that evaluating the spherical harmonics requires the computation of very large factorials, resulting in numerical overflow.) We allow the evaluation budget to increase commensurately with the number of dimensions of the integrand. We compare to MMLT and BQMC (Bayesian Quadrature with an ordinary GP with a design set selected using a sobol sequence). As the scalability (in the number of observations) of these two methods is limited, we set the evaluation budgets to 512. For methods that use an acquisition function we initialise half the evaluation budget with a sobol sequence, and allow the remainder to be built up using the acquisition function. For the remaining baselines we use Annealed Importance Sampling (AIS) and Nested Sampling (NS). Table 5.2, compares the runtime for VISH-PI, BLNBQ and MMLT. We use the maximum runtime between VISH-PI and BLNBQ to set the time budget for AIS and NS on the corresponding problem.

We find that VISH-PI performs well for low-dimensional integrands but its performance degrades as the dimensionality increases. This appears to be because of the difficulty in optimising the model’s parameters. Good performance requires tuning the priors of these parameters to restrict them to relatively narrow ranges, and this manual tuning becomes more challenging as the dimensionality increases. Another issue is that large numbers of samples are required to reduce the variance in a particular region, making the active learning scheme heavily exploitative. Higher dimensional integrands typically require a more exploratory approach as the initial

points are less likely to be in a region with high integrand values.

BLNBQ is often able to outperform MMLT, likely because it is able to leverage significantly more evaluations of the integrand. It is also consistently faster than MMLT in wall-clock time. For higher dimensional integrands, MMLT overestimates the integral value because it learns long lengthscales due to the large distances between integrand evaluations, the exploitative nature of the active sampling scheme and its explicit incorporation of the integrand’s non-negativity. (Together, these create the effect that the model believes the integrand’s peaks to be wider than they really are.) BQMC, on the other hand, underestimates the integral values as it does not do active sampling or model the non-negativity of the integrand. BLNBQ compares favourably to BQMC, both due to the larger number of integrand evaluations and the advantage of active sampling. However, especially as the dimensionality increases, BLNBQ is outperformed by AIS and NS. These baselines also have low overhead, and so are capable of gathering a large number of integrand queries in the wall-clock time budget. Additionally training of the Bézier Log-Normal Process is often unstable, and can result in very large moments for the variational posterior. In the highest dimension this leads to overestimation of the integral by a significant margin.

Problem	Budget (Eval.)	VISH-PI	BLNBQ
GMM-4	1024	0.00443 \pm 0.00036	0.02316 \pm 0.00193
GMM-8	2048	0.46865 \pm 0.08496	0.06993 \pm 0.00578
housing	4096	–	0.47164 \pm 0.09747
cancer	8192	–	0.18453 \pm 0.01285
ionosphere	8192	–	0.61357 \pm 0.38508
sonar	16384	–	1.24535 \pm 0.43564

Problem	Budget (Eval.)	MMLT	BQMC
GMM-4	512	0.00640 \pm 0.01008	0.01528 \pm 0.00488
GMM-8	512	0.29552 \pm 0.30302	0.40825 \pm 0.16655
housing	512	0.86154 \pm 0.06479	1.00000 \pm 0.00000
cancer	512	0.68953 \pm 0.03799	1.00000 \pm 0.00000
ionosphere	512	0.92627 \pm 0.05207	1.00000 \pm 0.00000
sonar	512	1.01423 \pm 0.05007	1.00000 \pm 0.00000

Problem	Budget (Time)	AIS	NS
GMM-4	200	0.01966 \pm 0.00682	0.03724 \pm 0.00589
GMM-8	400	0.30340 \pm 0.03879	0.08011 \pm 0.00947
housing	600	0.51321 \pm 0.02938	0.11465 \pm 0.01935
cancer	1260	0.33170 \pm 0.03758	0.08230 \pm 0.01741
ionosphere	600	0.99375 \pm 0.00314	0.24999 \pm 0.03310
sonar	2520	0.50424 \pm 0.17086	0.49598 \pm 0.03398

Table 5.1: The final fractional integration error after the evaluation budget is exhausted for our proposals, VISH-PI and BLNBQ, compared against several baselines. All values shown are means and standard error of the mean over 3 repeats. We see that our proposals often compare favourably to existing BQ methods, but do not perform well compared to other baselines when given the same wall-clock time budget.

Problem	d	VISH-PI	BLNBQ	MMLT
GMM-4	4	148.5 \pm 2.204	171.1 \pm 1.310	2163 \pm 3.458
GMM-8	8	400.8 \pm 9.404	230.8 \pm 16.81	2389 \pm 9.399
housing	15	–	498.6 \pm 0.207	2598 \pm 24.53
cancer	32	–	1249 \pm 0.382	2483 \pm 4.661
ionosphere	34	–	514.8 \pm 0.195	2420 \pm 0.775
sonar	60	–	2458 \pm 5.614	2671 \pm 2.372

Table 5.2: The runtime in seconds required for each algorithm to build up the design set for each problem. The values shown are means and standard deviations over 3 repeats.

5.7 Discussion

We have investigated the use of the VISH model as a cheaper surrogate for building up the design set for probabilistic integration. Additionally, we proposed the Bézier Log-Normal Process, and showed how to approximate its induced posterior over the integral of the modelled function. We also investigated the use of this model for adaptive Bayesian Quadrature.

Our empirical evaluation indicates that VISH-PI is competitive with alternative methods of cheaply building up design sets for probabilistic integration, especially in lower dimension. However, it does not scale well in terms of accuracy to higher dimensional spaces. BLNBQ scales better than VISH-PI and often improves upon existing Bayesian Quadrature methods, but performs poorly compared to NS as optimisation of the model parameters is often unstable.

5.8 Supplement

5.8.1 Mixed Inducing Variables for VISH

The VISH model uses inducing features which capture global properties of the function, i.e. the projection of the function onto the spherical harmonics. Local structure is very important, however, especially for the marginalisation integrals that are typical in machine learning, as they often have narrow peaks. Therefore, it may be advantageous to use both pseudo-input inducing variables – for modelling local structure – and spherical harmonic inducing variables – for modelling global structure. We refer to this model as “VISH-PI-M”.

Denote by $u_{l,k}^{(s)} = \langle f(\cdot), \phi_{l,k}(\cdot) \rangle_{\mathcal{H}}$ the projection of the function onto the spherical harmonics as before, by $u_m^{(i)}$ the pseudo-input inducing variable at location Z_m , and

by u the set of all inducing variables. The Σ_{uu} matrix has the structure

$$\Sigma_{uu} = \begin{bmatrix} \Sigma_{u^{(s)}u^{(s)}} & \Sigma_{u^{(s)}u^{(i)}} \\ \Sigma_{u^{(i)}u^{(s)}} & \Sigma_{u^{(i)}u^{(i)}} \end{bmatrix} \quad (5.20)$$

where $\Sigma_{u^{(s)}u^{(s)}}$ is a diagonal matrix, whose elements are given by Equation (5.10), and by Equation (5.9) the elements of $\Sigma_{u^{(s)}u^{(i)}}$ are given by

$$\Sigma(u_{l,k}^{(s)}, u_m^{(i)}) = \phi_{l,k}(Z_m). \quad (5.21)$$

We note that the diagonal structure of $\Sigma_{u^{(s)}u^{(s)}}$ can still be exploited to reduce the cost of inference (compared to an ordinary Variational GP which is cubic in the total number of inducing variables) by utilising its Schur complement, $C = \Sigma_{u^{(i)}u^{(i)}} - \Sigma_{u^{(i)}u^{(s)}}\Sigma_{u^{(s)}u^{(s)}}^{-1}\Sigma_{u^{(s)}u^{(i)}}$, to compute the Cholesky decomposition of Σ_{uu} . Denoting by $L_{u^{(s)}u^{(s)}}$ the Cholesky decomposition of $\Sigma_{u^{(s)}u^{(s)}}$, and by L_C the Cholesky decomposition of C , the Cholesky decomposition of Σ_{uu} is

$$L_{uu} = \begin{bmatrix} L_{u^{(s)}u^{(s)}} & 0 \\ \Sigma_{u^{(i)}u^{(s)}}L_{u^{(s)}u^{(s)}}^{-T} & L_C \end{bmatrix}. \quad (5.22)$$

The cost of inference with VISH-PI-M is dominated by the cost of computing the Cholesky decomposition of C and remains quadratic in the total number of inducing variables – it is $\mathcal{O}(M_i^3 + N(M_i + M_s)^2)$ where N is the number of data points and M_i and M_s are the number of pseudo-input and inter-domain inducing variables respectively.

As the local structure that we most care about modelling are the peaks of the integrand, we heuristically select the pseudo-input inducing variables to correspond to the M_i integrand observations with the highest values.

Empirical Comparison

We evaluate the introduction of mixed inducing variables by comparing the modeling performance of VISH with and without mixed inducing variables on fixed design sets. These datasets are drawn by sampling randomly from the prior, for each of the integrands that we are interested in. The results are shown in Table 5.3. Additionally, the integration error that results from using this model as a surrogate to build up the design set is shown in Table 5.4 under the column heading, “VISH-PI-M”. We examine performance on several problems:

GMM A 4D synthetic likelihood with a Gaussian Mixture Model and integrate this against a Gaussian prior.

BLR A 5D synthetic dataset from a Bayesian Logistic Regression model with a Gaussian prior over the weights. We compute its model evidence by marginalising over its weights.

GPR The marginalisation over hyperparameters for Gaussian Process regression on the UCI Yacht Hydrodynamics dataset with an ARD RBF kernel. This is an 8D integration problem (over 6 lengthscale hyperparameters, the signal variance hyperparameter, and the likelihood variance hyperparameter).

The results indicate that introducing pseudo-input inducing variables often degrades performance significantly in terms of the NLPD of the test set. For the largest dataset, the RMSE is also made worse. As expected, this means that the design sets produced by VISH-PI-M yield worse estimates of the integral than the design sets produced by VISH-PI.

					NLPD	
Problem	D	N	M_s	M_i	VISH-M	VISH
GMM	4	500	196	111	49.483 ± 19.962	15.414 ± 5.8553
BLR	5	750	378	122	382.38 ± 63.764	121.49 ± 9.4289
GPR	8	1250	660	222	2992700 ± 3651400	461360 ± 220430

					RMSE	
Problem					VISH-M	VISH
GMM					3.4304 ± 0.77233	3.4343 ± 0.72141
BLR					20.648 ± 1.8088	21.104 ± 0.95877
GPR					2085.0 ± 337.29	1879.4 ± 541.89

Table 5.3: The modelling performance of VISH and VISH-M, measured by the Negative Log Predictive Density (NLPD) and Root Mean Squared Error (RMSE) of a held-out test set sampled from the prior. The dimensionality of the integrand (i.e. the function being modelled) is shown under the heading “D”, the number of training points under “N”, the number of inter-domain inducing variables under “ M_s ”, and the number of pseudo-input inducing variables under “ M_i ”. The size of the test set is always a quarter of the size of the training set. The values shown are means and standard deviations over 5 repeats. Surprisingly, introducing pseudo-input inducing variables degrades the performance of the model in most cases.

Problem	D	N	VISH-PI-M	VISH-PI
GMM	4	500	0.00044 ± 0.00029	0.00024 ± 0.00023
BLR	5	750	0.02535 ± 0.00224	0.01255 ± 0.00208
GPR	8	1250	1.1982 ± 0.21376	1.0195 ± 0.20669

Table 5.4: A comparison of integration performance between VISH-PI and VISH-PI-M, which has additional pseudo-input inducing variables. The fractional error in the posterior mean for the integral is reported (in log space for BLR and GPR). The values shown are means and standard deviations over 5 repeats.


Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).


Title of Paper	Scalable Bayesian Quadrature with Scalable Gaussian Processes Approximations
Publication Status	<input type="checkbox"/> Published <input type="checkbox"/> Accepted for Publication <input type="checkbox"/> Submitted for Publication <input checked="" type="checkbox"/> Unpublished and unsubmitted work written in a manuscript style
Publication Details	Hamid, S., Jørgensen, M., Dutordoir, V. & Osborne, M. (2023). Scalable Bayesian Quadrature with Scalable Gaussian Process Approximations.

Student Confirmation

Student Name:	Saad Hamid		
Contribution to the Paper	<ul style="list-style-type: none">• Joint formulation of initial research direction.• Contributed to discussions iterating on the methodologies.• Joint discussion of experimental design.• Coding for the experiments (excluding Bézier Log-Normal Process and basic VISH implementations).• Write-up of the paper.		
Signature		Date	11/01/2023

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Prof Michael A. Osborne			
Supervisor comments I certify that the candidate made a substantial contribution to the publication, just as described above.			
Signature		Date	11 January 2023

This completed form should be included in the thesis, at the end of the relevant chapter.

Chapter 6

Conclusion

A core challenge in machine learning is performing inference in models whose likelihoods can be expensive to evaluate – as is common for large datasets – or for which the dimensionality of the parameters is large. This work has addressed this challenge by designing Bayesian Quadrature schemes. In particular, Chapter 3 has focussed on inferring kernels for Gaussian Process Regression, Chapter 4 on inferring architectures for Neural Networks, and Chapter 5 on inference in models with a large number of real-valued parameters.

To address the kernel learning problem for Gaussian Process regression we used the maximum mean discrepancy between distributions to define a hyper-kernel between kernels represented as spectral densities. We also extended an information-theoretic acquisition function for warped Bayesian Quadrature. These components were then brought together within the active Bayesian Quadrature framework to marginalise over a broad class of stationary kernels. An empirical investigation showed that our proposal outperformed state-of-the-art baselines.

We proposed Bayesian Quadrature as a means of selecting ensembles of Neural Networks from search spaces of network architectures. In particular, we showed that uncertainty sampling with a WSABI surrogate model is an effective means of selecting a candidate set of architectures to train. We investigated using recombination to select the ensemble members from this candidate set. However, we found that this underperformed compared to selecting the highest weighted architectures

after maximising the performance on a validation set with respect to the weights of a weighted sum of all candidate architectures in the candidate set.

To extend Bayesian Quadrature to higher-dimensional spaces we investigated the use of scalable approximations to Gaussian Processes as surrogates for BQ. In particular, we focussed on non-negative integrands, of the sort frequently encountered in machine learning. We explored the use of the recently proposed VISH – a variational GP with inter-domain inducing variables corresponding to the spherical harmonics – and Bézier GP – a Bézier Curve with Gaussians over the control points – models. We empirically compared our proposals to existing baselines, and found that they do not perform as well.

6.1 Future Directions

Our work makes progress towards tractable kernel learning for Gaussian Processes, ensemble search for Neural Networks, and inference for models with large numbers of real-valued parameters. Open questions remain, however, and several possible avenues for future research present themselves:

- For GP kernel learning, the curse of dimensionality limits the family of kernels that our Bayesian Quadrature scheme can marginalise over, but limiting the number of Gaussians that we can use to represent the spectral densities in practice. This limitation is made even more severe if we allow each Gaussian to have full (rather than diagonal) covariance structure. An exciting possible extension of our work would seek to address this limitation.
- Recall that Bochner’s theorem can be extended to non-stationary kernels. In this case, the spectral decompositions are bi-variate, and representing them will require higher-dimensional representations. However, this does provide a potential avenue for extending our proposed methodology for GP kernel learning to a class of kernels that includes non-stationary kernels.

- An exciting possible extension of our work on Bayesian Quadrature for Neural Ensemble Search is to marginalise over both architecture weights and architectures. Significant attention has been devoted in the literature to methods for marginalising over the weights of a given architecture. These are orthogonal to our approach and an investigation of which is most performant, and to what extent they improve performance over MLE estimates would be of value to the community. Our proposal for scalable BQ can also be used to perform the marginalisation over architecture weights. In this case, a promising direction is to explore using the information-theoretic acquisition function suggested by Chai, Ton et al. (2019). This targets the weight setting $w \mid \alpha$ that most reduces the entropy of the estimate of the model evidence $p(\mathcal{D})$. The advantage of such an acquisition function is that it naturally allocates a greater number of likelihood evaluations to architectures with a higher marginal likelihood $p(\mathcal{D} \mid \alpha)$. This is likely to lead to better performance given the same evaluation budget.
- An interesting alternative our proposal for scalable BQ is to use different a different model class as a surrogate, for example Bayesian Neural Networks. Recent work (Müller et al. 2022) has shown that NNs can approximate well the output of a GP, including uncertainty estimates. This is a promising direction for extending BQ to higher-dimensional spaces.

Bibliography

- Ambrogioni, Luca and Eric Maris (2018). ‘Integral Transforms from Finite Data: An Application of Gaussian Process Regression to Fourier Analysis’. In: *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics*, pp. 217–225.
- Bachoc, Francois, Alexandra Suvorikova, David Ginsbourger, Jean-Michel Loubes and Vladimir Spokoiny (2019). ‘Gaussian processes with multidimensional distribution inputs via optimal transport and Hilbertian embedding’. In: *arXiv:1805.00753 [stat]*.
- Bachoc, François, Fabrice Gamboa, Jean-Michel Loubes and Nil Venet (2018). ‘A Gaussian Process Regression Model for Distribution Inputs’. In: *IEEE Transactions on Information Theory* 64.10, pp. 6620–6637.
- Belfer, Yuval, Amnon Geifman, Meirav Galun and Ronen Basri (2021). ‘Spectral Analysis of the Neural Tangent Kernel for Deep Residual Networks’. In: *arXiv:2104.03093*.
- Benton, Gregory W., Wesley J. Maddox, Jayson P. Salkey, Julio Albinati and Andrew Gordon Wilson (2019). ‘Function-Space Distributions over Kernels’. In: *Advances in Neural Information Processing Systems*.
- Berlinet, Alain and Christine Thomas-Agnan (2004). *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Springer.
- Bietti, Alberto and Julien Mairal (2019). ‘On the Inductive Bias of Neural Tangent Kernels’. In: *Advances in Neural Information Processing Systems* 32.
- Bishop, Christopher M. (2006). *Pattern recognition and machine learning*. Information science and statistics. New York: Springer.
- Blei, David M., Alp Kucukelbir and Jon D. McAuliffe (2018). ‘Variational Inference: A Review for Statisticians’. In: *Journal of the American Statistical Association* 112.518, pp. 859–877.
- Bochner, Salomon (1959). *Lectures on Fourier Integrals*. Vol. 42. Princeton University Press.
- Bogachev, Vladimir Igorevich (1961). *Gaussian Measures*.

- Briol, François-Xavier, Chris J. Oates, Mark Girolami and Michael A. Osborne (2015). ‘Frank-Wolfe Bayesian Quadrature: Probabilistic Integration with Theoretical Guarantees’. In: *Advances in Neural Information Processing Systems*, pp. 1162–1170.
- Briol, François-Xavier, Chris J. Oates, Mark Girolami, Michael A. Osborne and Dino Sejdinovic (2015). ‘Probabilistic Integration: A Role in Statistical Computation?’ In: *arXiv:1512.00933 [cs, math, stat]*.
- Burt, David R., Carl Edward Rasmussen and Mark van der Wilk (2020). ‘Convergence of Sparse Variational Inference in Gaussian Processes Regression’. In: *Journal of Machine Learning Research* 21, pp. 1–63.
- Carathéodory, Constantin (1911). ‘Über den variabilitätsbereich der fourier’schen konstanten von positiven harmonischen funktionen’. In: *Rendiconti del Circolo Matematico di Palermo* 32, pp. 193–217.
- Chai, Henry and Roman Garnett (2019). ‘Improving Quadrature for Constrained Integrand’. In: *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*.
- Chai, Henry, Jean-Francois Ton, Michael A Osborne and Roman Garnett (2019). ‘Automated Model Selection with Bayesian Quadrature’. In: *36th International Conference on Machine Learning*, pp. 1555–1564.
- Chen, Ricky T. Q., Yulia Rubanova, Jesse Bettencourt and David Duvenaud (2018). ‘Neural Ordinary Differential Equations’. In: *Advances in Neural Information Processing Systems*. Vol. 33.
- Chen, Xin, Lingxi Xie, Jun Wu and Qi Tian (2019). ‘Progressive Differentiable Architecture Search: Bridging the Depth Gap between Search and Evaluation’. In: *arXiv:1904.12760*.
- Cho, Youngmin and Lawrence K Saul (2009). ‘Kernel Methods for Deep Learning’. In: *Advances in Neural Information Processing Systems*. Vol. 22, p. 9.
- Cohen, Taco S. and Max Welling (2016). ‘Group Equivariant Convolutional Networks’. In: *Proceedings of the 33rd International Conference on Machine Learning*.
- Cutajar, Kurt, Michael A Osborne, John P Cunningham and Maurizio Filippone (2016). ‘Preconditioning Kernel Matrices’. In: *Proceedings of the 33rd International Conference on Machine Learning*.
- Cuturi, Marco (2013). ‘Sinkhorn Distances: Lightspeed Computation of Optimal Transport’. In: *Advances in Neural Information Processing Systems 26*, pp. 2292–2300.

- Dai, Feng and Yuan Xu (2013). *Approximation Theory and Harmonic Analysis on Spheres and Balls*. Springer Monographs in Mathematics. New York, NY: Springer New York.
- Diaconis, Persi (1988). ‘Bayesian Numerical Analysis’. In: *Statistical Decision Theory and Related Topics IV*, pp. 163–175.
- Dong, Xuanyi, Lu Liu, Katarzyna Musial and Bogdan Gabrys (2021). ‘NATS-Bench: Benchmarking NAS Algorithms for Architecture Topology and Size’. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Dong, Xuanyi and Yi Yang (2020). ‘NAS-Bench-201: Extending the Scope of Reproducible Neural Architecture Search’. In: *arXiv:2001.00326*.
- Dua, Dheeru and Casey Graff (2017). *UCI Machine Learning Repository*.
- Dutordoir, Vincent, Nicolas Durrande and James Hensman (2020). ‘Sparse Gaussian Processes with Spherical Harmonic Features’. In: *Proceedings of the 37th International Conference on Machine Learning*.
- Duvenaud, David, James Robert Lloyd, Roger Grosse, Joshua B. Tenenbaum and Zoubin Ghahramani (2013). ‘Structure Discovery in Nonparametric Regression through Compositional Kernel Search’. In: *Proceedings of the 30th International Conference on Machine Learning*.
- Elsken, Thomas, Jan Hendrik Metzen and Frank Hutter (2019). ‘Neural Architecture Search: A Survey’. In: *Journal of Machine Learning Research* 20.55, pp. 1–21.
- England, Bank of (2020). *Broad Effective Exchange Rate Index, Sterling*.
- Feragen, Aasa, Francois Lauze and Søren Hauberg (2015). ‘Geodesic Exponential Kernels: When Curvature and Linearity Conflict’. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3032–3042.
- Feydy, Jean (2020). ‘Geometric data analysis, beyond convolutions’. Doctor of Philosophy. Paris-Saclay University.
- Fubini, Guido (1907). ‘Sugli integrali multipli’. In: *Rom. Acc. L. Rend. (5)* 16.1, pp. 608–614.
- Gal, Yarin and Richard Turner (2015). ‘Improving the Gaussian Process Sparse Spectrum Approximation by Representing Uncertainty in Frequency Inputs’. In: *32nd International Conference on Machine Learning*. Vol. 1, pp. 655–664.
- Gardner, Jacob R., Geoff Pleiss, David Bindel, Kilian Q. Weinberger and Andrew Gordon Wilson (2018). ‘GPYtorch: Blackbox Matrix-Matrix Gaussian Process

- Inference with GPU Acceleration’. In: *Proceedings of the 32nd Conference on Neural Information Processing Systems*.
- Garnett, Roman (2021). *Bayesian Optimization*. Cambridge University Press.
- Garriga-Alonso, Adrià, Carl Edward Rasmussen and Laurence Aitchison (2019). ‘Deep Convolutional Networks as shallow Gaussian Processes’. In: *International Conference on Learning Representations*.
- Genton, Marc G (2001). ‘Classes of Kernels for Machine Learning: A Statistics Perspective’. In: *Journal of Machine Learning Research* 2, pp. 299–312.
- Gessner, Alexandra, Javier Gonzalez and Maren Mahsereci (2019). ‘Active Multi-Information Source Bayesian Quadrature’. In: *Proceedings of the 35th Conference on Uncertainty in Artificial Intelligence*.
- Ghosh, Swarnendu, Nibaran Das, Teresa Gonçalves, Paulo Quaresma and Mahantapas Kundu (2018). ‘The journey of graph kernels through two decades’. In: *Computer Science Review* 27, pp. 88–111.
- Goodfellow, Ian, Yoshua Bengio and Aaron Courville (2016). *Deep Learning*. MIT Press.
- Green, Peter J (1995). ‘Reversible jump Markov chain Monte Carlo computation and Bayesian model determination’. In: *Biometrika* 82.4, pp. 711–732.
- Gretton, Arthur, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf and Alexander Smola (2012). ‘A Kernel Two-Sample Test’. In: *Journal of Machine Learning Research* 13, pp. 723–773.
- Gunter, Tom, Michael A. Osborne, Roman Garnett, Philipp Hennig and Stephen J. Roberts (2014). ‘Sampling for Inference in Probabilistic Models with Fast Bayesian Quadrature’. In: *Proceedings of the 28th Annual Conference on Neural Information Processing Systems*. Vol. 4, pp. 2789–2797.
- Guo, Chuan, Geoff Pleiss, Yu Sun and Kilian Q. Weinberger (2017). ‘On Calibration of Modern Neural Networks’. In: *arXiv:1706.04599*.
- Hayakawa, Satoshi, Harald Oberhauser and Terry Lyons (2022). ‘Positively Weighted Kernel Quadrature via Subsampling’. In: *arXiv:2107.09597*.
- Hendrycks, Dan and Thomas Dietterich (2019). ‘Benchmarking Neural Network Robustness to Common Corruptions and Perturbations’. In: *Proceedings of the International Conference on Learning Representations*.
- Hennig, P and M A Osborne (2021). *Probabilistic Numerics*. Cambridge University Press.

- Hensman, James, Nicolas Durrande and Arno Solin (2018). ‘Variational Fourier features for Gaussian processes’. In: *Journal of Machine Learning Research* 18, pp. 1–52.
- Hensman, James, Nicolo Fusi and Neil D Lawrence (2013). ‘Gaussian Processes for Big Data’. In: *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence*, p. 9.
- Jacot, Arthur, Franck Gabriel and Clement Hongler (2018). ‘Neural Tangent Kernel: Convergence and Generalization in Neural Networks’. In: *Advances in Neural Information Processing Systems* 31.
- Jang, Phillip A, Andrew Loeb, Matthew Davidow and Andrew G Wilson (2017). ‘Scalable Levy Process Priors for Spectral Kernel Learning’. In: *Advances in Neural Information Processing Systems*, pp. 3941–3950.
- Jayasumana, Sadeep, Richard Hartley, Mathieu Salzmann, Hongdong Li and Mehrtash Harandi (2015). ‘Kernel Methods on Riemannian Manifolds with Gaussian RBF Kernels’. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.12, pp. 2464–2477.
- Jørgensen, Martin and Michael A. Osborne (2022). ‘Bézier Gaussian Processes for Tall and Wide Data’. In: *arXiv:2209.00343*.
- Kakihara, Yûichirô (1985). ‘A note on harmonizable and V-bounded processes’. In: *Journal of Multivariate Analysis* 16.1, pp. 140–156.
- Kanagawa, Motonobu, Bharath K. Sriperumbudur and Kenji Fukumizu (2020). ‘Convergence Analysis of Deterministic Kernel-Based Quadrature Rules in Misspecified Settings’. In: *Foundations of Computational Mathematics* 20.1, pp. 155–194.
- Kandasamy, Kirthevasan, Willie Neiswanger, Jeff Schneider, Barnabas Poczos and Eric Xing (2019). ‘Neural Architecture Search with Bayesian Optimisation and Optimal Transport’. In: *arXiv:1802.07191*.
- Kaul, Manohar (2013). *3D Road Network (North Jutland, Denmark) Data Set*.
- Kriege, Nils M., Fredrik D. Johansson and Christopher Morris (2020). ‘A Survey on Graph Kernels’. In: *Applied Network Science* 5.1, p. 6.
- Lakshminarayanan, Balaji, Alexander Pritzel and Charles Blundell (2017). ‘Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles’. In: *Advances in Neural Information Processing Systems*.
- Lázaro-Gredilla, Miguel, Joaquin Quiñero-Candela, Carl Edward Rasmussen and Aníbal R Figueiras-Vidal (2010). ‘Sparse Spectrum Gaussian Process Regression’. In: *Journal of Machine Learning Research* 11, pp. 1–17.

- Lean, J. (2004). *Solar Irradiance Reconstruction. IGBP PAGES/World Data Center for Paleoclimatology, Data Contribution Series \# 2004-035. NOAA/NGDC Paleoclimatology Program, Boulder CO, USA.*
- Lee, Jaehoon, Yasaman Bahri, Roman Novak, Samuel S. Schoenholz, Jeffrey Pennington and Jascha Sohl-Dickstein (2018). ‘Deep Neural Networks as Gaussian Processes’. In: *6th International Conference on Learning Representations*.
- Leibfried, Felix, Vincent Dutordoir, S. T. John and Nicolas Durrande (2021). ‘A Tutorial on Sparse Gaussian Processes and Variational Inference’. In: *arXiv:2012.13962 [cs, stat]*.
- Lim, Bryan, Sercan Ö. Arik, Nicolas Loeff and Tomas Pfister (2021). ‘Temporal Fusion Transformers for interpretable multi-horizon time series forecasting’. In: *International Journal of Forecasting* 37.4, pp. 1748–1764.
- Liu, Hanxiao, Karen Simonyan and Yiming Yang (2019). ‘DARTS: Differentiable Architecture Search’. In: *arXiv:1806.09055*.
- Liu, Yuqiao, Yanan Sun, Bing Xue, Mengjie Zhang, Gary G. Yen and Kay Chen Tan (2021). ‘A Survey on Evolutionary Neural Architecture Search’. In: *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–21.
- Lopez, Roberto (2013). *Yacht Hydrodynamics Data Set*.
- (2014). *Airfoil Self-Noise Data Set*.
- Lugosi, Gabor (2006). ‘Universal Kernels’. In: *Journal of Machine Learning Research* 7, pp. 2651–2667.
- Ma, Lizheng, Jiaxu Cui and Bo Yang (2019). ‘Deep Neural Architecture Search with Deep Graph Bayesian Optimization’. In: *IEEE/WIC/ACM International Conference on Web Intelligence*. Thessaloniki Greece: ACM, pp. 500–507.
- MacKay, David J. C. (1999). ‘Comparison of Approximate Methods for Handling Hyperparameters’. In: *Neural Computation* 11.5, pp. 1035–1068.
- Makridakis, S., S. Wheelwright and R. Hyndman (1998). *Airline Passenger Data Set*.
- Matthews, Alexander G. de G., Mark Rowland, Jiri Hron, Richard E. Turner and Zoubin Ghahramani (2018). ‘Gaussian Process Behaviour in Wide Deep Neural Networks’. In: *6th International Conference on Learning Representations*.
- Matthews, Alexander Graeme de Garis (2016). ‘Scalable Gaussian process inference using variational methods’. Doctor of Philosophy. University of Cambridge.
- Minka, Thomas (2000). *Deriving quadrature rules from Gaussian processes*. Technical. Statistics Department, Carnegie Mellon University, pp. 1–21.

- Muandet, Krikamol, Kenji Fukumizu, Bharath Sriperumbudur and Bernhard Schölkopf (2017). ‘Kernel Mean Embedding of Distributions: A Review and Beyond’. In: *Foundations and Trends in Machine Learning* 10.1, pp. 1–141.
- Müller, Samuel, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka and Frank Hutter (2022). ‘Transformers Can Do Bayesian Inference’. In: *Proceedings of the International Conference on Learning Representations*.
- Murphy, Kevin P. (2012). *Machine learning: a probabilistic perspective*. Adaptive computation and machine learning series. Cambridge, MA: MIT Press.
- (2022). *Probabilistic machine learning: an introduction*. Adaptive computation and machine learning series. Cambridge, Massachusetts: The MIT Press.
- Murray, Iain, Ryan Prescott Adams and David J C MacKay (2010). ‘Elliptical slice sampling’. In: *Journal of Machine Learning Research* 9, pp. 541–548.
- Neal, Radford (1996). *Bayesian Learning for Neural Networks*. Vol. 118. Lecture Notes in Statistics. Springer.
- Nikolentzos, Giannis, Giannis Siglidis and Michalis Vazirgiannis (2021). ‘Graph Kernels: A Survey’. In: *Journal of Artificial Intelligence Research* 72, pp. 943–1027.
- NOAA, USA (2020). *Mauna Loa Atmospheric CO₂ Observations*.
- Novak, Erich (2016). ‘Some Results on the Complexity of Numerical Integration’. In: *Monte Carlo and Quasi-Monte Carlo Methods*. Vol. 163. Springer Proceedings in Mathematics & Statistics, pp. 161–183.
- O’Hagan, A. (1991). ‘Bayes–Hermite quadrature’. In: *Journal of Statistical Planning and Inference* 29.3, pp. 245–260.
- (1992). ‘Some Bayesian Numerical Analysis’. In: *Bayesian Statistics* 4, pp. 345–363.
- Oliva, Junier B, Avinava Dubey, Andrew G Wilson, Barnabas Poczos, Jeff Schneider and Eric P Xing (2016). ‘Bayesian Nonparametric Kernel-Learning’. In: *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pp. 1078–1086.
- Osborne, Michael A, David Duvenaud, Roman Garnett, Carl E Rasmussen, Stephen J Roberts and Zoubin Ghahramani (2012). ‘Active Learning of Model Evidence Using Bayesian Quadrature’. In: *Advances in Neural Information Processing Systems* 26, pp. 46–54.
- Osborne, Michael A, Roman Garnett, Stephen J Roberts, Christopher Hart, Suzanne Aigrain and Neale P Gibson (2012). ‘Bayesian Quadrature for Ratios’. In:

- Proceedings of the 15th International Conference on Artificial Intelligence and Statistics*, pp. 832–840.
- Ott, Myle, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier and Michael Auli (2019). ‘fairseq: A Fast, Extensible Toolkit for Sequence Modeling’. In: *Proceedings of the 2019 Conference of the North*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 48–53.
- Peyré, Gabriel and Marco Cuturi (2019). ‘Computational Optimal Transport’. In: *Foundations and Trends in Machine Learning* 11.5, pp. 1–257.
- Rahimi, Ali and Benjamin Recht (2008). ‘Random Features for Large-Scale Kernel Machines’. In: *Advances in Neural Information Processing Systems 20*. Curran Associates, Inc., pp. 1177–1184.
- Rasmussen, Carl and Zoubin Ghahramani (2003). ‘Bayesian Monte Carlo’. In: *Advances in Neural Information Processing Systems 16*.
- Rasmussen, Carl and Christopher Williams (2006). *Gaussian Processes for Machine Learning*. MIT Press.
- Real, Esteban, Alok Aggarwal, Yanping Huang and Quoc V. Le (2019). ‘Regularized Evolution for Image Classifier Architecture Search’. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33, pp. 4780–4789.
- Real, Esteban, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V Le and Alexey Kurakin (2017). ‘Large-Scale Evolution of Image Classifiers’. In: *Proceedings of the 34th International Conference on Machine Learning*.
- Remes, Sami, Markus Heinonen and Samuel Kaski (2017). ‘Non-Stationary Spectral Kernels’. In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc., pp. 4642–4651.
- Ren, Pengzhen, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen and Xin Wang (2020). ‘A Comprehensive Survey of Neural Architecture Search: Challenges and Solutions’. In: *ACM Computing Surveys* 37.4.
- Ronneberger, Olaf, Philipp Fischer and Thomas Brox (2015). ‘U-Net: Convolutional Networks for Biomedical Image Segmentation’. In: *Medical Image Computing and Computer-Assisted Intervention (MICCAI)* 9351.234.
- Ross, Quinlan (1993). *Auto MPG Data Set*.
- Ru, Binxin, Mark McLeod, Diego Granziol and Michael A Osborne (2018). ‘Fast Information-theoretic Bayesian Optimisation’. In: *Proceedings of the 35th International Conference on Machine Learning*, p. 9.

- Ru, Binxin, Xingchen Wan, Xiaowen Dong and Michael Osborne (2021). ‘Interpretable Neural Architecture Search via Bayesian Optimisation with Weisfeiler-Lehman Kernels’. In: *Proceedings of the 9th International Conference on Learning Representations*.
- Samo, Yves-Laurent Kom (2017). ‘Advances in Kernel Methods’. DPhil. Oxford, UK: University of Oxford.
- Samo, Yves-Laurent Kom and Stephen Roberts (2015). ‘Generalized Spectral Kernels’. In: *arXiv:1506.02236 [stat]*.
- Shervashidze, Nino (2011). ‘Weisfeiler-Lehman Graph Kernels’. In: *Journal of Machine Learning Research* 12, pp. 2539–2561.
- Shu, Yao, Yizhou Chen, Zhongxiang Dai and Bryan Kian Hsiang Low (2022). *Neural Ensemble Search via Bayesian Sampling*.
- Simpson, Fergus, Vidhi Lalchand and Carl Rasmussen (2021). ‘Marginalised Spectral Mixture Kernels with Nested Sampling’. In: *Advances in Neural Information Processing Systems*. Vol. 34.
- Snelson, Edward, Zoubin Ghahramani and Carl E Rasmussen (2004). ‘Warped Gaussian Processes’. In: *Advances in Neural Information Processing Systems 17*, pp. 1–8.
- Sriperumbudur, Bharath K., Kenji Fukumizu, Arthur Gretton, Bernhard Schölkopf and Gert R. G. Lanckriet (2012). ‘On integral probability metrics, ϕ -divergences and binary classification’. In: *Electronic Journal of Statistics* 6, pp. 1550–1599.
- Svensson, Andreas, Johan Dahlin and Thomas B. Schön (2015). ‘Marginalizing Gaussian Process Hyperparameters using Sequential Monte Carlo’. In: *6th IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, pp. 477–480.
- Tchernychova, Maria (2015). ‘Caratheodory cubature measures’. Doctor of Philosophy. University of Oxford.
- Teng, Tong, Jie Chen, Yehong Zhang and Kian Hsiang Low (2019). ‘Scalable Variational Bayesian Kernel Selection for Sparse Gaussian Process Regression’. In: *Proceedings of the 34th AAAI Conference on Artificial Intelligence*.
- Titsias, Michalis K (2009). ‘Variational Learning of Inducing Variables in Sparse Gaussian Processes’. In: *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, pp. 567–574.
- Tobar, Felipe (2018). ‘Bayesian Nonparametric Spectral Estimation’. In: *Proceedings of the 32nd Conference on Neural Information Processing Systems*, p. 11.

- Wang, Chien-Yao, Alexey Bochkovskiy and Hong-Yuan Mark Liao (2022). ‘YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors’. In: *arXiv:2207.02696*.
- White, Colin, Willie Neiswanger and Yash Savani (2020). ‘BANANAS: Bayesian Optimization with Neural Architectures for Neural Architecture Search’. In: *arXiv:1910.11858*.
- Wilson, Andrew Gordon and Ryan Prescott Adams (2013). ‘Gaussian Process Kernels for Pattern Discovery and Extrapolation’. In: *30th International Conference on Machine Learning*, pp. 2104–2112.
- Worrall, Daniel E., Stephan J. Garbin, Daniyar Turmukhambetov and Gabriel J. Brostow (2017). ‘Harmonic Networks: Deep Translation and Rotation Equivariance’. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Xi, Xiaoyue, François-Xavier Briol and Mark Girolami (2018). ‘Bayesian Quadrature for Multiple Related Integrals’. In: *35th International Conference on Machine Learning*. Vol. 12, pp. 8533–8564.
- Xu, Yuhui, Lingxi Xie, Xiaopeng Zhang, Xin Chen, Guo-Jun Qi, Qi Tian and Hongkai Xiong (2020). ‘PC-DARTS: Partial Channel Connections for Memory-Efficient Architecture Search’. In: *arXiv:1907.05737*.
- Yang, Greg (2019). ‘Tensor Programs I: Wide Feedforward or Recurrent Neural Networks of Any Architecture are Gaussian Processes’. In: *Advances in Neural Information Processing Systems 32*.
- Yeh, I-Cheng (2007). *Concrete Compressive Strength Data Set*.
- Yu, Jiahui, Linjie Yang, Ning Xu, Jianchao Yang and Thomas Huang (2019). ‘Slimmable Neural Networks’. In: *International Conference on Learning Representations*.
- Zaheer, Manzil, Satwik Kottu, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov and Alex Smola (2017). ‘Deep Sets’. In: *Advances in Neural Information Processing Systems*. Vol. 31.
- Zaidi, Sheheryar, Arber Zela, Thomas Elsken, Chris Holmes, Frank Hutter and Yee Whye Teh (2022). ‘Neural Ensemble Search for Uncertainty Estimation and Dataset Shift’. In: *arXiv:2006.08573*.
- Zhu, Harrison, Xing Liu, Ruya Kang, Zhichao Shen, Seth Flaxman and François-Xavier Briol (2020). ‘Bayesian Probabilistic Numerical Integration with Tree-Based Models’. In: *Proceedings of the 34th Conference on Neural Information Processing Systems*.