Automatic Bayesian Cubature manuscript No.
(will be inserted by the editor)

Fast Automatic Bayesian Cubature Using Lattice Sampling

R. Jagadeeswaran · Fred J. Hickernell

Received: date / Accepted: date

Abstract Automatic cubatures approximate multidimensional integrals to user-specified error tolerances. For high dimensional problems, it makes sense to fix the sampling density but determine the sample size, n, automatically. Bayesian cubature postulates that the integrand is an instance of a stochastic process. Here we assume a Gaussian process parameterized by a constant mean and a covariance function defined by a scale parameter times a parameterized function specifying how the integrand values at two different points in the domain are related. These parameters are estimated from integrand values or are given non-informative priors. The sample size, n, is chosen to make the half-width of the credible interval for the Bayesian posterior mean no greater than the error tolerance.

The process just outlined typically requires vector-matrix operations with a computational cost of $O(n^3)$. Our innovation is to pair low discrepancy nodes with matching kernels that lower the computational cost to $O(n \log n)$. This approach is demonstrated using rank-1 lattice sequences and shift-invariant kernels. Our algorithm is implemented in the Guaranteed Automatic Integration Library (GAIL).

Keywords Bayesian cubature · Fast automatic cubature · GAIL · Probabilistic numeric methods

R. Jagadeeswaran
Department of Applied Mathematics,
Illinois Institute of Technology
10 W. 32nd St., Room 208
Chicago IL 60616
E-mail: jrathin1@iit.edu

Fred J. Hickernell
Center for Computational Science and Department of Applied Mathematics, Illinois Institute of Technology
PS 106, 3105 S. Dearborn St., Chicago IL 60616
E-mail: hickernell@iit.edu

1 Introduction

Cubature is the problem of inferring a numerical value for an integral, $\mu := \int_{\mathbb{R}^d} g(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}$, where μ has no closed form analytic expression. Typically, g is accessible as a black-box algorithm. Cubature is a key component of many problems in scientific computing, finance, statistical modeling, and machine learning.

The integral may often be expressed as

$$\mu := \mathbb{E}[f(\boldsymbol{X})] = \int_{[0,1]^d} f(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}, \tag{1}$$

where $f : [0,1]^d \to \mathbb{R}$ is the integrand, and $\boldsymbol{X} \sim \mathcal{U}[0,1]^d$. The process of transforming the original integral into the form of (1) is not addressed here. See [6, Section 2.11] for a discussion of variable transformations. The cubature may be an affine function of integrand values:

$$\widehat{\mu} := w_0 + \sum_{i=1}^{n} f(\boldsymbol{x}_i) w_i, \tag{2}$$

where the weights, w_0, and $\boldsymbol{w} = (w_i)_{i=1}^n \in \mathbb{R}^n$, and the nodes, $\{\boldsymbol{x}_i\}_{i=1}^n \subset [0,1]^d$, are chosen to make the error, $|\mu - \widehat{\mu}|$, small. The integration domain $[0,1]^d$ is convenient for the low discrepancy node sets [6,24] that we use. The nodes are assumed to be deterministic.

We construct a reliable stopping criterion that determines the number of integrand values, n, required to ensure that the error is no greater than a user-defined error tolerance denoted by ε, i.e.,

$$|\mu - \widehat{\mu}| \le \varepsilon. \tag{3}$$

Rather than relying on strong assumptions about the integrand, such as an upper bound on its variance or total variation, we construct a stopping criterion that is based on a credible interval arising from a Bayesian approach to the problem. We build upon the work of Briol et al. [1], Diaconis [5], O'Hagan [18], Ritter [22],

Rasmussen and Ghahramani [20], and others. Our algorithm is an example of *probabilistic numerics*.

The primary contribution of this article is to demonstrate how the choice of a family of covariance kernels that match the low discrepancy sampling nodes facilitates fast computation of the cubature and the data-driven stopping criterion. Our cubature requires n function values—at a cost of \$(f) each—plus $\mathcal{O}(n\log(n))$ operations to check whether the error tolerance is satisfied. The total cost of our algorithm is then $\mathcal{O}(n[\$(f)+\log(n)])$. This is significantly fewer operations than the $\mathcal{O}(n^3)$ typically required for Bayesian cubature. If function evaluation is expensive, then \$(f) might be similar in magnitude to $\log(n)$.

Hickernell [11] compares different approaches to cubature error analysis depending on whether the rule is deterministic or random and whether the integrand is assumed to be deterministic or random. Error analysis that assumes a deterministic integrand lying in a Banach space leads to an error bound that is typically impractical for deciding how large n must be to satisfy (3). The deterministic error bound includes a (semi-) norm of the integrand, often called the variation, which is often more complex to compute than the original integral.

Hickernell and Jiménez-Rugama [12, 15] have developed stopping criteria for cubature rules based on low discrepancy nodes by tracking the decay of the discrete Fourier coefficients of the integrand. The algorithm proposed here also relies on discrete Fourier coefficients, but in a different way. Although we only explore automatic Bayesian cubature for absolute error tolerances, the recent work by Hickernell, Jiménez-Rugama, and Li [13] suggests how one might accommodate more general error criteria, such as relative error tolerances.

Section 2 explains the Bayesian approach to estimate the posterior cubature error and defines our automatic Bayesian cubature. Although much of this material is known, it is included for completeness. We end Section 2 by demonstrating why Bayesian cubature is typically computationally expensive. Section 3 introduces the concept of covariance kernels that match the nodes and expedite the computations required by our automatic Bayesian cubature. Section 4 implements this concept for shift invariant kernels and rank-1 lattice nodes. It also describes how to avoid cancellation error for kernels of product form. Numerical examples are provided in Section 5 to demonstrate our new algorithm. We conclude with a brief discussion.

2 Bayesian Cubature

2.1 Bayesian posterior error

Suppose that the integrand, f, is drawn from a Gaussian process, i.e., $f \sim \mathcal{GP}(m, s^2 C_{\boldsymbol{\theta}})$. Specifically, f has real-valued constant mean m and covariance function $s^2 C_{\boldsymbol{\theta}}$, where s is a non-negative scale factor, and $C_{\boldsymbol{\theta}} : [0,1]^d \times [0,1]^d \to \mathbb{R}$ is a symmetric, positive-definite function and parameterized by $\boldsymbol{\theta}$:

$$\mathsf{C}^T = \mathsf{C}, \quad \boldsymbol{a}^T \mathsf{C} \boldsymbol{a} > 0, \quad \text{where } \mathsf{C} = (C_{\boldsymbol{\theta}}(\boldsymbol{x}_i, \boldsymbol{x}_j))_{i,j=1}^n, \quad \text{for all } \boldsymbol{a} \neq 0,\ n \in \mathbb{N},\ \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in [0,1]^d. \quad (4)$$

The function, C, and the Gram matrix, C depend implicitly on $\boldsymbol{\theta}$, but the notation may omit this parameter for simplicity's sake.

For a Gaussian process, all vectors of linear functionals of f have a multivariate Gaussian distribution. Defining $\boldsymbol{f} := (f(\boldsymbol{x}_i))_{i=1}^n$ as the multivariate normal vector of function values, it follows that

$$\boldsymbol{f} \sim \mathcal{N}(m\mathbf{1}, s^2\mathsf{C}), \quad (5a)$$

where $\mathbf{1}$ is a vector of all ones,

$$\mu \sim \mathcal{N}(m, s^2 c_0), \quad (5b)$$

$$\text{where } c_0 = \int_{[0,1]^d \times [0,1]^d} C_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{t}) \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}\boldsymbol{t}, \quad (5c)$$

$$\operatorname{cov}(\boldsymbol{f}, \mu) = \left(\int_{[0,1]^d} C(\boldsymbol{t}, \boldsymbol{x}_i) \, \mathrm{d}\boldsymbol{t} \right)_{i=1}^n =: \boldsymbol{c}. \quad (5d)$$

We need the following lemma to derive the distribution of the posterior error of our cubature.

Lemma 1 *[21, (A.6), (A.11–13)] If* $\boldsymbol{Y} = (\boldsymbol{Y}_1, \boldsymbol{Y}_2)^T \sim \mathcal{N}(\boldsymbol{m}, \mathsf{C})$*, where* $\boldsymbol{Y}_1$ *and* $\boldsymbol{Y}_2$ *are random vectors of arbitrary length, and*

$$\boldsymbol{m} = \begin{pmatrix} \boldsymbol{m}_1 \\ \boldsymbol{m}_2 \end{pmatrix} = \begin{pmatrix} \mathbb{E}(\boldsymbol{Y}_1) \\ \mathbb{E}(\boldsymbol{Y}_2) \end{pmatrix},$$

$$\mathsf{C} = \begin{pmatrix} \mathsf{C}_{11} & \mathsf{C}_{21}^T \\ \mathsf{C}_{21} & \mathsf{C}_{22} \end{pmatrix} = \begin{pmatrix} \operatorname{var}(\boldsymbol{Y}_1) & \operatorname{cov}(\boldsymbol{Y}_1, \boldsymbol{Y}_2) \\ \operatorname{cov}(\boldsymbol{Y}_2, \boldsymbol{Y}_1) & \operatorname{var}(\boldsymbol{Y}_2) \end{pmatrix}$$

then

$$\boldsymbol{Y}_1 | \boldsymbol{Y}_2 \sim \mathcal{N}\big(\boldsymbol{m}_1 + \mathsf{C}_{21}^T \mathsf{C}_{22}^{-1} (\boldsymbol{Y}_2 - \boldsymbol{m}_2), \ \mathsf{C}_{11} - \mathsf{C}_{21}^T \mathsf{C}_{22}^{-1} \mathsf{C}_{21}\big).$$

Moreover, the inverse of the matrix C *may be partitioned as*

$$\mathsf{C}^{-1} = \begin{pmatrix} \mathsf{A}_{11} & \mathsf{A}_{21}^T \\ \mathsf{A}_{21} & \mathsf{A}_{22} \end{pmatrix},$$

$$\mathsf{A}_{11} = (\mathsf{C}_{11} - \mathsf{C}_{12}\mathsf{C}_{22}^{-1}\mathsf{C}_{21})^{-1}, \qquad \mathsf{A}_{21} = -\mathsf{C}_{22}^{-1}\mathsf{C}_{21}\mathsf{A}_{11},$$
$$\mathsf{A}_{22} = \mathsf{C}_{22}^{-1} + \mathsf{C}_{22}^{-1}\mathsf{C}_{21}\mathsf{A}_{11}\mathsf{C}_{21}^T\mathsf{C}_{22}^{-1}.$$

It follows from Lemma 1 that the *conditional* distribution of the integral given observed function values,

$\boldsymbol{f} = \boldsymbol{y}$ is also Gaussian:

$$\mu|(\boldsymbol{f} = \boldsymbol{y}) \sim \mathcal{N}\big(m(1 - \boldsymbol{c}^T\mathsf{C}^{-1}\mathbf{1}) + \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{y}, \; s^2(c_0 - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{c})\big). \quad (6)$$

The natural choice for the cubature is the posterior mean of the integral, namely,

$$\widehat{\mu}|(\boldsymbol{f} = \boldsymbol{y}) = m(1 - \mathbf{1}^T\mathsf{C}^{-1}\boldsymbol{c}) + \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{y}, \quad (7)$$

which takes the form of (2). Under this definition, the cubature error has zero mean and a variance depending on the choice of nodes:

$$(\mu - \widehat{\mu})|(\boldsymbol{f} = \boldsymbol{y}) \sim \mathcal{N}\left(0, \; s^2(c_0 - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{c})\right).$$

A credible interval for the integral is given given by

$$\mathbb{P}_f\left[|\mu - \widehat{\mu}| \le \mathrm{err}_{\mathrm{CI}}\right] = 99\%, \quad (8a)$$

$$\mathrm{err}_{\mathrm{CI}} = 2.58 s \sqrt{c_0 - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{c}} \quad (8b)$$

Naturally, 2.58 and 99% can be replaced by other quantiles and credible levels.

2.2 Parameter estimation

The credible interval in (8) suggests how our automatic Bayesian cubature proceeds. Integrand data is accumulated until the width of the credible interval, $\mathrm{err}_{\mathrm{CI}}$, is no greater than the error tolerance. As n increases, one expects $\sqrt{c_0 - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{c}}$ to decrease for well-chosen nodes, $\{\boldsymbol{x}_i\}_{i=1}^n$.

Note that $\mathrm{err}_{\mathrm{CI}}$ has no explicit dependence on the integrand values, even though one would intuitively expect that larger integrand should imply a larger $\mathrm{err}_{\mathrm{CI}}$. This is because parameters, m, s, and $\boldsymbol{\theta}$, have not yet been inferred from integrand data. After inferring the parameters, $\mathrm{err}_{\mathrm{CI}}$ does reflect the size of the integrand values. This section describes three approaches to parameter estimation.

2.2.1 Empirical Bayes

One approach is to estimate the parameters is via maximum likelihood estimation (MLE). The log-likelihood function of the parameters given the function data $\boldsymbol{y}$ is:

$$l(s, m, \boldsymbol{\theta}|\boldsymbol{y}) = -\frac{1}{2}s^{-2}(\boldsymbol{y} - m\mathbf{1})^T\mathsf{C}^{-1}(\boldsymbol{y} - m\mathbf{1}) - \frac{1}{2}\log(\det\,\mathsf{C}) - \frac{n}{2}\log(s^2) + \text{constants}.$$

Maximizing the log-likelihood first with respect to m, then with respect to s, and finally with respect to $\boldsymbol{\theta}$ yields

$$m_{\mathrm{MLE}} = \frac{\mathbf{1}^T\mathsf{C}^{-1}\boldsymbol{y}}{\mathbf{1}^T\mathsf{C}^{-1}\mathbf{1}}, \quad (9)$$

$$s^2_{\mathrm{MLE}} = \frac{1}{n}(\boldsymbol{y} - m_{\mathrm{MLE}}\mathbf{1})^T\mathsf{C}^{-1}(\boldsymbol{y} - m_{\mathrm{MLE}}\mathbf{1}) = \frac{1}{n}\boldsymbol{y}^T\left[\mathsf{C}^{-1} - \frac{\mathsf{C}^{-1}\mathbf{1}\mathbf{1}^T\mathsf{C}^{-1}}{\mathbf{1}^T\mathsf{C}^{-1}\mathbf{1}}\right]\boldsymbol{y}, \quad (10)$$

$$\boldsymbol{\theta}_{\mathrm{MLE}} = \underset{\boldsymbol{\theta}}{\mathrm{argmin}}\left\{\log\left(\boldsymbol{y}^T\left[\mathsf{C}^{-1} - \frac{\mathsf{C}^{-1}\mathbf{1}\mathbf{1}^T\mathsf{C}^{-1}}{\mathbf{1}^T\mathsf{C}^{-1}\mathbf{1}}\right]\boldsymbol{y}\right) + \frac{1}{n}\log(\det(\mathsf{C}))\right\}. \quad (11)$$

The MLE estimate of $\boldsymbol{\theta}$ balances minimizing the covariance scale factor, s^2_{MLE}, against minimizing $\det(\mathsf{C})$.

Under these estimates of the parameters, the cubature (7) and the credible interval (8) simplify to

$$\widehat{\mu}_{\mathrm{MLE}} = \left(\frac{(1 - \mathbf{1}^T\mathsf{C}^{-1}\boldsymbol{c})\mathbf{1}}{\mathbf{1}^T\mathsf{C}^{-1}\mathbf{1}} + \boldsymbol{c}\right)^T\mathsf{C}^{-1}\boldsymbol{y}, \quad (12)$$

$$\mathrm{err}^2_{\mathrm{MLE}} := \frac{2.58^2}{n}\boldsymbol{y}^T\left[\mathsf{C}^{-1} - \frac{\mathsf{C}^{-1}\mathbf{1}\mathbf{1}^T\mathsf{C}^{-1}}{\mathbf{1}^T\mathsf{C}^{-1}\mathbf{1}}\right]\boldsymbol{y} \times (c_0 - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{c}), \quad (13)$$

$$\mathbb{P}_f\left[|\mu - \widehat{\mu}_{\mathrm{MLE}}| \le \mathrm{err}_{\mathrm{MLE}}\right] = 99\%. \quad (14)$$

Here c_0, $\boldsymbol{c}$, and C are assumed implicitly to be based on $\boldsymbol{\theta} = \boldsymbol{\theta}_{\mathrm{MLE}}$.

2.2.2 Full Bayes

Rather than use maximum likelihood to determine m and s one can treat them as hyperparameters with a non-informative, conjugate prior, namely $\boldsymbol{\rho}_{m,s^2}(\xi, \lambda) \propto 1/\lambda$. Then the posterior density for the integral given the data using Bayes theorem is

$$\begin{aligned}
&\rho_\mu(z|\boldsymbol{f} = \boldsymbol{y}) \\
&\propto \int_0^\infty \int_{-\infty}^\infty \rho_\mu(z|\boldsymbol{f} = \boldsymbol{y}, m = \xi, s^2 = \lambda) \\
&\qquad \times \rho_{\boldsymbol{f}}(\boldsymbol{y}|\xi, \lambda)\rho_{m,s^2}(\xi, \lambda)\,\mathrm{d}\xi\mathrm{d}\lambda \\
&\propto \int_0^\infty \frac{1}{\lambda^{(n+3)/2}} \\
&\times \int_{-\infty}^\infty \exp\left(-\frac{1}{2\lambda}\left\{\frac{[z - \xi(1 - \boldsymbol{c}^T\mathsf{C}^{-1}\mathbf{1}) - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{y}]^2}{c_0 - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{c}} + (\boldsymbol{y} - \xi\mathbf{1})^T\mathsf{C}^{-1}(\boldsymbol{y} - \xi\mathbf{1})\right\}\right)\mathrm{d}\xi\mathrm{d}\lambda \\
&\qquad \text{by (5), (6) and } \rho_{m,s^2}(\xi, \lambda) \propto 1/\lambda \\
&\propto \int_0^\infty \frac{1}{\lambda^{(n+3)/2}} \cdots \\
&\quad \cdots \int_{-\infty}^\infty \exp\left(-\frac{\alpha\xi^2 - 2\beta\xi + \gamma}{2\lambda(c_0 - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{c})}\right)\mathrm{d}\xi\mathrm{d}\lambda,
\end{aligned}$$

where

$$\alpha = (1 - \boldsymbol{c}^T\mathsf{C}^{-1}\mathbf{1})^2 + \mathbf{1}^T\mathsf{C}^{-1}\mathbf{1}(c_0 - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{c}),$$
$$\beta = (1 - \boldsymbol{c}^T\mathsf{C}^{-1}\mathbf{1})(z - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{y}) + \mathbf{1}^T\mathsf{C}^{-1}\boldsymbol{y}(c_0 - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{c}),$$
$$\gamma = (z - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{y})^2 + \boldsymbol{y}^T\mathsf{C}^{-1}\boldsymbol{y}(c_0 - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{c}).$$

In the derivation above and below, factors that are independent of ξ, λ, or z can be discarded since we only need to preserve the proportion. But, factors that depend on ξ, λ, or z must be kept. Completing the square, $\alpha\xi^2 - 2\beta\xi + \gamma = \alpha(\xi - \beta/\alpha)^2 - (\beta^2/\alpha) + \gamma$, allows us to evaluate the integrals with respect to ξ and λ:

$$\begin{aligned}
&\rho_\mu(z|\boldsymbol{f} = \boldsymbol{y}) \\
&\quad\propto \int_0^\infty \frac{1}{\lambda^{(n+3)/2}} \exp\left(-\frac{\gamma - \beta^2/\alpha}{2\lambda(c_0 - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{c})}\right) \cdots \\
&\qquad\qquad \cdots \int_{-\infty}^\infty \exp\left(-\frac{\alpha(\xi - \beta/\alpha)^2}{2\lambda(c_0 - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{c})}\right) \mathrm{d}\xi\mathrm{d}\lambda \\
&\quad\propto \int_0^\infty \frac{1}{\lambda^{(n+2)/2}} \exp\left(-\frac{\gamma - \beta^2/\alpha}{2\lambda(c_0 - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{c})}\right) \mathrm{d}\lambda \\
&\quad\propto \left(\gamma - \frac{\beta^2}{\alpha}\right)^{-n/2} \propto \left(\alpha\gamma - \beta^2\right)^{-n/2}.
\end{aligned}$$

Finally, we simplify the key term via straightforward calculations to the following:

$$\alpha\gamma - \beta^2 \propto 1 + \frac{(z - \widehat{\mu}_{\mathrm{MLE}})^2}{(n-1)\widehat{\sigma}^2_{\mathrm{full}}},$$

where

$$\widehat{\sigma}^2_{\mathrm{full}} := \frac{1}{n-1}\boldsymbol{y}^T\left[\mathsf{C}^{-1} - \frac{\mathsf{C}^{-1}\mathbf{1}\mathbf{1}^T\mathsf{C}^{-1}}{\mathbf{1}^T\mathsf{C}^{-1}\mathbf{1}}\right]\boldsymbol{y} \times \left[\frac{(1 - \boldsymbol{c}^T\mathsf{C}^{-1}\mathbf{1})^2}{\mathbf{1}^T\mathsf{C}^{-1}\mathbf{1}} + (c_0 - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{c})\right].$$

This means that $\mu|(\boldsymbol{f} = \boldsymbol{y})$, properly centered and scaled, has a Student's t-distribution with $n-1$ degrees of freedom. The estimated integral is the same as in the empirical Bayes case, $\widehat{\mu}_{\mathrm{full}} = \widehat{\mu}_{\mathrm{MLE}}$, but the confidence interval is wider:

$$\mathbb{P}_f\left[|\mu - \widehat{\mu}_{\mathrm{full}}| \le \mathrm{err}_{\mathrm{full}}\right] = 99\%, \tag{15}$$

where

$$\mathrm{err}_{\mathrm{full}} := t_{n_j-1,0.995}\widehat{\sigma}_{\mathrm{full}} > \mathrm{err}_{\mathrm{MLE}}. \tag{16}$$

Here $t_{n-1,0.995}$ denotes the 99.5 percentile of a standard Student's t-distribution with $n-1$ degrees of freedom.

Because the shape parameter, $\boldsymbol{\theta}$, enters the definition of the covariance kernel in a non-trivial way, the only way to treat it as a hyperparameter and assign a tractable prior would be for the prior to be discrete. We believe in practice that choosing such a prior involves more guesswork than using the empirical Bayes estimate of $\boldsymbol{\theta}$ in (11) or the cross-validation approach described next.

2.2.3 Generalized Cross-Validation

A third parameter optimization technique is *leave-one-out cross-validation* (CV). Let $\widetilde{y}_i = \mathbb{E}[f(\boldsymbol{x}_i)|\boldsymbol{f}_{-i} = \boldsymbol{y}_{-i}]$, where the subscript $-i$ denotes the vector excluding the i^{th} component. This is the conditional expectation of $f(\boldsymbol{x}_i)$ given all data but the function value at $\boldsymbol{x}_i$. The cross-validation criterion, which is to be minimized, is sum of squares of the difference between these conditional expectations and the observed values:

$$\mathrm{CV} = \sum_{i=1}^n (y_i - \widetilde{y}_i)^2. \tag{17}$$

Let $\mathsf{A} = \mathsf{C}^{-1}$, let $\boldsymbol{\zeta} = \mathsf{A}(\boldsymbol{y} - m\mathbf{1})$, and partition C, A, and $\boldsymbol{\zeta}$ as

$$\mathsf{C} = \begin{pmatrix} c_{ii} & \boldsymbol{C}_{-i,i}^T \\ \boldsymbol{C}_{-i,i} & \mathsf{C}_{-i,-i} \end{pmatrix}, \qquad \mathsf{A} = \begin{pmatrix} a_{ii} & \boldsymbol{A}_{-i,i}^T \\ \boldsymbol{A}_{-i,i} & \mathsf{A}_{-i,-i} \end{pmatrix}, \qquad \boldsymbol{\zeta} = \begin{pmatrix} \zeta_i \\ \boldsymbol{\zeta}_{-i} \end{pmatrix},$$

where the subscript i denotes the i^{th} row or column, and the subscript $-i$ denotes all rows or columns except the i^{th}. Following this notation, Lemma 1 implies that

$$\begin{aligned}
\widetilde{y}_i &= m + \boldsymbol{C}_{-i,i}^T\mathsf{C}_{-i,-i}^{-1}(\boldsymbol{y}_{-i} - m\mathbf{1}) \\
\zeta_i &= a_{ii}(y_i - m) + \boldsymbol{A}_{-i,i}^T(\boldsymbol{y}_{-i} - m\mathbf{1}) \\
&= a_{ii}[(y_i - m) - \boldsymbol{C}_{-i,i}^T\mathsf{C}_{-i,-i}^{-1}(\boldsymbol{y}_{-i} - m\mathbf{1})] \\
&= a_{ii}(y_i - \widetilde{y}_i).
\end{aligned}$$

Thus, (17) may be re-written as

$$\mathrm{CV} = \sum_{i=1}^n \left(\frac{\zeta_i}{a_{ii}}\right)^2, \qquad \boldsymbol{\zeta} = \mathsf{C}^{-1}(\boldsymbol{y} - m\mathbf{1}).$$

The *generalized cross-validation* criterion (GCV) replaces the i^{th} diagonal element of A in the denominator by the average diagonal element of A [4,9,25]:

$$\begin{aligned}
\mathrm{GCV} &= \frac{\sum_{i=1}^n \zeta_i^2}{\left(\frac{1}{n}\sum_{i=1}^n a_{ii}\right)^2} \\
&= \frac{(\boldsymbol{y} - m\mathbf{1})^T\mathsf{C}^{-2}(\boldsymbol{y} - m\mathbf{1})}{\left(\frac{1}{n}\mathrm{trace}(\mathsf{C}^{-1})\right)^2}.
\end{aligned}$$

The loss function GCV depends on m and $\boldsymbol{\theta}$, but not on s. Minimizing the GCV yields

$$m_{\mathrm{GCV}} = \frac{\mathbf{1}^T\mathsf{C}^{-2}\boldsymbol{y}}{\mathbf{1}^T\mathsf{C}^{-2}\mathbf{1}},$$

$$\boldsymbol{\theta}_{\mathrm{GCV}} = \operatorname*{argmin}_{\boldsymbol{\theta}}\left\{\log\left(\boldsymbol{y}^T\left[\mathsf{C}^{-2} - \frac{\mathsf{C}^{-2}\mathbf{1}\mathbf{1}^T\mathsf{C}^{-2}}{\mathbf{1}^T\mathsf{C}^{-2}\mathbf{1}}\right]\boldsymbol{y}\right) - 2\log\left(\mathrm{trace}(\mathsf{C}^{-1})\right)\right\}. \tag{18}$$

Plugging this value of m into (7) yields

$$\widehat{\mu}_{\mathrm{GCV}} = \left(\frac{(1 - \mathbf{1}^T\mathsf{C}^{-1}\boldsymbol{c})\mathsf{C}^{-1}\mathbf{1}}{\mathbf{1}^T\mathsf{C}^{-2}\mathbf{1}} + \boldsymbol{c}\right)^T\mathsf{C}^{-1}\boldsymbol{y}. \tag{19}$$

An estimate for s may be obtained by noting that by Lemma 1,

$$\mathrm{var}[f(\boldsymbol{x}_i)|\boldsymbol{f}_{-i} = \boldsymbol{y}_{-i}] = s^2 a_{ii}^{-1}.$$

Thus, we may estimate s using an argument similar to that used in deriving the GCV and then substituting m_{GCV} for m:

$$\begin{aligned} s^2 &= \mathrm{var}[f(\boldsymbol{x}_i)|\boldsymbol{f}_{-i} = \boldsymbol{y}_{-i}]a_{ii} \\ &\approx \frac{1}{n}\sum_{i=1}^{n}(y_i - \widetilde{y}_i)^2 a_{ii} = \frac{1}{n}\sum_{i=1}^{n}\frac{\zeta_i^2}{a_{ii}} \\ &\approx \frac{\frac{1}{n}\sum_{i=1}^{n}\zeta_i^2}{\frac{1}{n}\sum_{i=1}^{n}a_{ii}} = \frac{(\boldsymbol{y}-m\boldsymbol{1})^T\mathsf{C}^{-2}(\boldsymbol{y}-m\boldsymbol{1})}{\mathrm{trace}(\mathsf{C}^{-1})} \\ &\approx s^2_{\mathrm{GCV}}, \end{aligned}$$

where

$$s^2_{\mathrm{GCV}} := \boldsymbol{y}^T\left[\mathsf{C}^{-2} - \frac{\mathsf{C}^{-2}\boldsymbol{1}\boldsymbol{1}^T\mathsf{C}^{-2}}{\boldsymbol{1}^T\mathsf{C}^{-2}\boldsymbol{1}}\right]\boldsymbol{y}\left[\mathrm{trace}(\mathsf{C}^{-1})\right]^{-1}.$$

The confidence interval based on GCV corresponds to (8) with the estimated m, s, and $\boldsymbol{\theta}$:

$$\mathrm{err}_{\mathrm{GCV}} = 2.58 s_{\mathrm{GCV}}\sqrt{c_0 - \boldsymbol{c}^T\mathsf{C}^{-1}\boldsymbol{c}}, \tag{20}$$

$$\mathbb{P}_f\left[|\mu - \widehat{\mu}_{\mathrm{GCV}}| \le \mathrm{err}_{\mathrm{GCV}}\right] = 99\%. \tag{21}$$

Looking back over the results of Sections 2.2.1–2.2.3, it is noted that if the original covariance function, C, is replaced by bC for some positive constant b, the cubature, $\widehat{\mu}$, the estimates of $\boldsymbol{\theta}$, and the credible interval widths, $\mathrm{err}_{\mathrm{CI}}$, all remain unchanged. The estimates of s^2 are multiplied by b^{-1}, as would be expected.

2.3 The automatic Bayesian cubature algorithm

The previous section presents three credible intervals, (14), (15), and (21), for the μ, the desired integral. Each credible interval is based on different assumptions about the hyperparameters m, s, and $\boldsymbol{\theta}$. We stress that one must estimate these hyperparameters or assume a prior distribution on them because the credible intervals are used as stopping criteria for our cubature rule. Since a credible intervals makes a statement about a typical function—not an outlier—one must try to ensure that the integrand is a typical draw from the assumed Gaussian process.

Our Bayesian cubature algorithm increases the sample size until the width of the credible interval is small enough. This is accomplished through successively doubling the sample size. The steps are detailed in Algorithm 1.

Algorithm 1 Automatic Bayesian Cubature

Require: a generator for the sequence $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots$; a black-box function, f; an absolute error tolerance, $\varepsilon > 0$; the positive initial sample size, n_0; the maximum sample size $n_{\max}$

1: $n \leftarrow n_0$, $n' \leftarrow 0$, $\mathrm{err}_{\mathrm{CI}} \leftarrow \infty$
2: **while** $\mathrm{err}_{\mathrm{CI}} > \varepsilon$ and $n \le n_{\max}$ **do**
3: Generate $\{\boldsymbol{x}_i\}_{i=n'+1}^{n}$ and sample $\{f(\boldsymbol{x}_i)\}_{i=n'+1}^{n}$
4: Compute $\boldsymbol{\theta}$ by (11) or (18)
5: Compute $\mathrm{err}_{\mathrm{CI}}$ according to (13), (16), or (20)
6: $n' \leftarrow n$, $n \leftarrow 2n'$
7: **end while**
8: Update sample size to compute $\widehat{\mu}$, $n \leftarrow n'$
9: Compute $\widehat{\mu}$, the approximate integral, according to (12) or (19)
10: **return** $\widehat{\mu}$, n and $\mathrm{err}_{\mathrm{CI}}$

2.4 Example with the Matérn kernel

To demonstrate automatic Bayesian cubature consider a Matérn covariance kernel:

$$C_\theta(\boldsymbol{x}, \boldsymbol{t}) = \prod_{k=1}^{d}\exp(-\theta|\boldsymbol{x}_k - \boldsymbol{t}_k|)(1 + \theta|\boldsymbol{x}_k - \boldsymbol{t}_k|),$$

and Sobol points as the nodes. Also, consider the integration problem of evaluating *multivariate normal probabilities*:

$$\mu = \int_{(\boldsymbol{a},\boldsymbol{b})}\frac{\exp\left(-\frac{1}{2}\boldsymbol{t}^T\Sigma^{-1}\boldsymbol{t}\right)}{\sqrt{(2\pi)^d\det(\Sigma)}}\,\mathrm{d}\boldsymbol{t}, \tag{22}$$

where $(\boldsymbol{a}, \boldsymbol{b})$ is a finite, semi-infinite or infinite box in $\mathbb{R}^d$. This integral does not have an analytic expression for general Σ, so cubatures are required.

Genz [8] introduced a variable transformation to transform (22) into an integral on the unit cube. Let $\Sigma = \mathsf{L}\mathsf{L}^T$ be the Cholesky decomposition where $\mathsf{L} = (l_{jk})_{j,k=1}^{d}$ is a lower triangular matrix. Iteratively define

$$\alpha_1 = \Phi(a_1), \qquad \beta_1 = \Phi(b_1)$$

$$\alpha_j(x_1, \ldots, x_{j-1}) = \Phi\left(\frac{1}{l_{jj}}\left(a_j - \sum_{k=1}^{j-1} l_{jk}\Phi^{-1}(\alpha_k + x_k(\beta_k - \alpha_k))\right)\right), \quad j = 2, \ldots, d,$$

$$\beta_j(x_1, \ldots, x_{j-1}) = \Phi\left(\frac{1}{l_{jj}}\left(b_j - \sum_{k=1}^{j-1} l_{jk}\Phi^{-1}(\alpha_k + x_k(\beta_k - \alpha_k))\right)\right), \quad j = 2, \ldots, d,$$

$$f_{\mathrm{Genz}}(\boldsymbol{x}) = \prod_{j=1}^{d}[\beta_j(\boldsymbol{x}) - \alpha_j(\boldsymbol{x})]. \tag{23}$$

where Φ is the cumulative standard normal distribution function. Then, $\mu = \int_{[0,1]^{d-1}} f_{\mathrm{Genz}}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$.

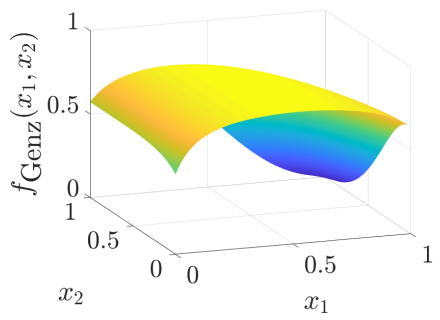

Fig. 1: The $d = 3$ multivariate normal probability transformed to an integral of f_{Genz} with $d = 2$. This plot can be reproduced using `IntegrandPlots.m` in GAIL

We use the following parameter values in the simulation:

$$d = 3, \quad \boldsymbol{a} = \begin{pmatrix} -6 \\ -2 \\ -2 \end{pmatrix}, \quad \boldsymbol{b} = \begin{pmatrix} 5 \\ 2 \\ 1 \end{pmatrix}, \quad \mathsf{L} = \begin{pmatrix} 4 & 1 & 1 \\ 0 & 1 & 0.5 \\ 0 & 0 & 0.25 \end{pmatrix}.$$

The node sets are randomly scrambled Sobol points [6, 7]. The results for $\varepsilon = 10^{-2}, 10^{-3}, 10^{-4}$, and 10^{-5} and 100 scrambles for each ε are shown in Figure 2. We observe the algorithm meets the error criterion 95% of the time. As shown in Figure 2, computation time increases rapidly with n. The maximum likelihood estimation of $\boldsymbol{\theta}$, which requires repeated evaluation of the objective function, is the most time consuming of all. It takes tens of seconds to compute $\widehat{\mu}_n$ once with $\varepsilon = 10^{-5}$. In contrast, this example in Section 5 take less than a hundredth of a second to compute $\widehat{\mu}_n$ once with $\varepsilon = 10^{-5}$ using our new algorithm. Not only is the Bayesian cubature with the the Matérn kernel slow, but also C becomes highly ill-conditioned as n increases. So, Algorithm 1 in its current form is impractical.

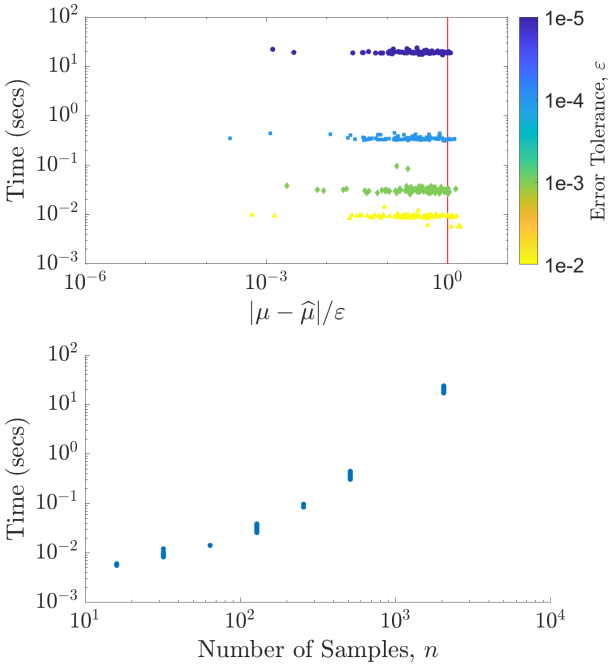

Fig. 2: Multivariate normal probability estimated using Matérn kernel with $d = 2$ using empirical stopping criterion. Top: Guaranteed integration within error tolerance ε. Bottom: Computation time rapidly increases with increase of n. These figures can be reproduced using `matern_guaranteed_plots.m` in GAIL.

3 Fast Automatic Bayesian Cubature

The generic automatic Bayesian cubature algorithm described in the last section requires $\mathcal{O}(n^3)$ operations to estimate $\boldsymbol{\theta}$, compute the credible interval width, and compute the cubature. Now we explain how to speed up the calculations. A key is to choose kernels that match the nodes, $\{\boldsymbol{x}_i\}_{i=1}^n$, so that the vector-matrix operations required by Bayesian cubature can be accomplished using fast transforms at a cost of $\mathcal{O}(n \log(n))$.

3.1 Fast Transform Kernel

We make some assumptions about the relationship between the covariance kernel and the nodes, which will be shown to hold in Section 4 for rank-1 lattices and shift-invariant kernels. First we introduce the notation

$$\begin{aligned} \mathsf{C} &= \left(C_{\boldsymbol{\theta}}(\boldsymbol{x}_i, \boldsymbol{x}_j)\right)_{i,j=1}^n = (\boldsymbol{C}_1, \ldots, \boldsymbol{C}_n) \\ &= \frac{1}{n}\mathsf{V}\Lambda\mathsf{V}^H, \qquad \mathsf{V}^H = n\mathsf{V}^{-1}, \\ \mathsf{V} &= (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n)^T = (\boldsymbol{V}_1, \ldots, \boldsymbol{V}_n), \\ \mathsf{C}^p &= \frac{1}{n}\mathsf{V}\Lambda^p\mathsf{V}^H, \qquad \forall p \in \mathbb{Z}, \end{aligned} \tag{24}$$

where V^H is the Hermitian of V. The columns of matrix V are eigenvectors of C, and Λ is a diagonal matrix of eigenvalues of C. For any $n \times n$ vector $\boldsymbol{b}$, define the notation $\widetilde{\boldsymbol{b}} := \mathsf{V}^H\boldsymbol{b}$.

We make three assumptions that allow the fast computation:

$$\mathsf{V} \text{ may be identified analytically,} \tag{25a}$$

$$\boldsymbol{v}_1 = \boldsymbol{V}_1 = \boldsymbol{1}, \tag{25b}$$

$$\mathsf{V}^H\boldsymbol{b} \text{ requires only } \mathcal{O}(n \log(n)) \text{ operations } \forall \boldsymbol{b}. \tag{25c}$$

We call the transformation $\boldsymbol{b} \mapsto \mathsf{V}^H\boldsymbol{b}$ a *fast transform* and C a *fast transform kernel*.

Under assumptions (25) the eigenvalues may be identified as the fast transform of the first column of C:

$$\boldsymbol{\lambda} = \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_n \end{pmatrix} = \Lambda \mathbf{1} = \Lambda \boldsymbol{v}_1^* = \underbrace{\left(\frac{1}{n} \mathsf{V}^H \mathsf{V}\right)}_{\mathsf{I}} \Lambda \boldsymbol{v}_1^*$$

$$= \mathsf{V}^H \left(\frac{1}{n} \mathsf{V} \Lambda \boldsymbol{v}_1^*\right) = \mathsf{V}^H \boldsymbol{C}_1 = \widetilde{\boldsymbol{C}}_1, \tag{26}$$

Where I is the identity matrix. Also note that the fast transform of $\mathbf{1}$ has a simple form

$$\widetilde{\mathbf{1}} = \mathsf{V}^H \mathbf{1} = \mathsf{V}^H \boldsymbol{V}_1 = \begin{pmatrix} n \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Many of the terms that arise in the calculations in Algorithm 1 take the form $\boldsymbol{a}^T \mathsf{C}^p \boldsymbol{b}$ for real $\boldsymbol{a}$ and $\boldsymbol{b}$ and integer p. These can be calculated via the transforms $\widetilde{\boldsymbol{a}} = \mathsf{V}^H \boldsymbol{a}$ and $\widetilde{\boldsymbol{b}} = \mathsf{V}^H \boldsymbol{b}$ as

$$\boldsymbol{a}^T \mathsf{C}^p \boldsymbol{b} = \frac{1}{n} \boldsymbol{a}^T \mathsf{V} \Lambda^p \mathsf{V}^H \boldsymbol{b} = \frac{1}{n} \widetilde{\boldsymbol{a}}^H \Lambda^p \widetilde{\boldsymbol{b}} = \frac{1}{n} \sum_{i=1}^n \lambda_i^p \widetilde{a}_i^* \widetilde{b}_i.$$

In particular,

$$\mathbf{1}^T \mathsf{C}^{-p} \mathbf{1} = \frac{n}{\lambda_1^p}, \qquad \mathbf{1}^T \mathsf{C}^{-p} \boldsymbol{y} = \frac{\widetilde{y}_1}{\lambda_1^p},$$

$$\boldsymbol{y}^T \mathsf{C}^{-p} \boldsymbol{y} = \frac{1}{n} \sum_{i=1}^n \frac{|\widetilde{y}_i|^2}{\lambda_i^p}, \qquad \boldsymbol{c}^T \mathsf{C}^{-1} \mathbf{1} = \frac{\widetilde{c}_1}{\lambda_1},$$

$$\boldsymbol{c}^T \mathsf{C}^{-1} \boldsymbol{y} = \frac{1}{n} \sum_{i=1}^n \frac{\widetilde{c}_i^* \widetilde{y}_i}{\lambda_i}, \qquad \boldsymbol{c}^T \mathsf{C}^{-1} \boldsymbol{c} = \frac{1}{n} \sum_{i=1}^n \frac{|\widetilde{c}_i|^2}{\lambda_i},$$

where $\widetilde{\boldsymbol{y}} = \mathsf{V}^H \boldsymbol{y}$ and $\widetilde{\boldsymbol{c}} = \mathsf{V}^H \boldsymbol{c}$. For any real $\boldsymbol{b}$, with $\widetilde{\boldsymbol{b}} = \mathsf{V}^H \boldsymbol{b}$, it follows that $\widetilde{b}_1$ is real since the first row of V^H is $\mathbf{1}$.

The covariance kernel used in practice also may satisfy an additional assumption:

$$\int_{[0,1]^d} C(\boldsymbol{t}, \boldsymbol{x}) \, \mathrm{d}\boldsymbol{t} = 1 \qquad \forall \boldsymbol{x} \in [0,1]^d, \tag{27}$$

which implies that $c_0 = 1$ and $\boldsymbol{c} = \mathbf{1}$. Under (27), the expressions above may be further simplified:

$$\boldsymbol{c}^T \mathsf{C}^{-1} \mathbf{1} = \boldsymbol{c}^T \mathsf{C}^{-1} \boldsymbol{c} = \frac{n}{\lambda_1}.$$

3.2 Empirical Bayes

Under assumptions (25), the empirical Bayes parameters in (9), (10), (11) (12), and (13) can be expressed in terms of the fast transforms of the function data, the first column of the Gram matrix, and $\boldsymbol{c}$ as follows:

$$m_{\mathrm{MLE}} = \frac{\widetilde{y}_1}{n} = \frac{1}{n} \sum_{i=1}^n y_i.$$

$$s^2_{\mathrm{MLE}} = \frac{1}{n^2} \sum_{i=2}^n \frac{|\widetilde{y}_i|^2}{\lambda_i}.$$

$$\boldsymbol{\theta}_{\mathrm{MLE}} = \operatorname*{argmin}_{\boldsymbol{\theta}} \left[\log \left(\sum_{i=2}^n \frac{|\widetilde{y}_i|^2}{\lambda_i} \right) + \frac{1}{n} \sum_{i=1}^n \log(\lambda_i) \right]. \tag{28}$$

$$\widehat{\mu}_{\mathrm{MLE}} = \frac{\widetilde{y}_1}{n} + \frac{1}{n} \sum_{i=2}^n \frac{\widetilde{c}_i^* \widetilde{y}_i}{\lambda_i}.$$

$$\mathrm{err}_{\mathrm{MLE}} = \frac{2.58}{n} \sqrt{\sum_{i=2}^n \frac{|\widetilde{y}_i|^2}{\lambda_i} \left(c_0 - \frac{1}{n} \sum_{i=1}^n \frac{|\widetilde{c}_i|^2}{\lambda_i} \right)}.$$

Since all the quantities on the right hand sides can be obtained in $\mathcal{O}(n \log(n))$ operations by fast transforms, the left hand sides are all computable using the asymptotic computational cost.

Under the further assumption (27), it follows that

$$\widehat{\mu}_{\mathrm{MLE}} = \frac{\widetilde{y}_1}{n} = \frac{1}{n} \sum_{i=1}^n y_i,$$

$$\mathrm{err}_{\mathrm{MLE}} = \frac{2.58}{n} \sqrt{\sum_{i=2}^n \frac{|\widetilde{y}_i|^2}{\lambda_i} \left(1 - \frac{n}{\lambda_1}\right)}. \tag{29}$$

Thus, in this case $\widehat{\mu}$ is simply the sample mean.

3.3 Full Bayes

For the full Bayes approach the cubature is the same as for empirical Bayes. We also defer to empirical Bayes to estimate the parameter $\boldsymbol{\theta}$. The width of the confidence interval is $\mathrm{err}_{\mathrm{full}} := t_{n_j - 1, 0.995} \widehat{\sigma}_{\mathrm{full}}$, where $\widehat{\sigma}^2_{\mathrm{full}}$ can also be computed swiftly under assumptions (25):

$$\widehat{\sigma}^2_{\mathrm{full}} = \frac{1}{n(n-1)} \sum_{i=2}^n \frac{|\widetilde{y}_i|^2}{\lambda_i} \times \left[\frac{\lambda_1}{n} \left(1 - \frac{\widetilde{c}_1}{\lambda_1}\right)^2 + \left(c_0 - \frac{1}{n} \sum_{i=1}^n \frac{|\widetilde{c}_i|^2}{\lambda_i} \right) \right],$$

Under assumption (27) further simplification can be made:

$$\widehat{\sigma}^2_{\mathrm{full}} = \frac{1}{n(n-1)} \sum_{i=2}^n \frac{|\widetilde{y}_i|^2}{\lambda_i} \left(\frac{\lambda_1}{n} - 1 \right),$$

It follows that

$$\mathrm{err}_{\mathrm{full}} = t_{n_j - 1, 0.995} \sqrt{\frac{1}{n(n-1)} \sum_{i=2}^n \frac{|\widetilde{y}_i|^2}{\lambda_i} \left(\frac{\lambda_1}{n} - 1 \right)}. \tag{30}$$

3.4 Generalized Cross-Validation

GCV yields a different cubature, which nevertheless can also be computed quickly using the fast transform. Under assumptions (25):

$$
\begin{aligned}
m_{\text{GCV}} &= m_{\text{MLE}} = \frac{\widetilde{y}_1}{n} = \frac{1}{n}\sum_{i=1}^{n} y_i,\\
s^2_{\text{GCV}} &:= \frac{1}{n}\sum_{i=2}^{n}\frac{|\widetilde{y}_i|^2}{\lambda_i^2}\left[\sum_{i=1}^{n}\frac{1}{\lambda_i}\right]^{-1},\\
\boldsymbol{\theta}_{\text{GCV}} &= \operatorname*{argmin}_{\boldsymbol{\theta}}\left[\log\left(\sum_{i=2}^{n}\frac{|\widetilde{y}_i|^2}{\lambda_i^2}\right)\right.\\
&\qquad\left.-2\log\left(\sum_{i=1}^{n}\frac{1}{\lambda_i}\right)\right], \qquad (31)\\
\widehat{\mu}_{\text{GCV}} &= \widehat{\mu}_{\text{MLE}} = \frac{\widetilde{y}_1}{n} + \frac{1}{n}\sum_{i=2}^{n}\frac{\widetilde{c}_i^*\widetilde{y}_i}{\lambda_i},\\
\text{err}_{\text{GCV}} &= \frac{2.58}{n}\left\{\sum_{i=2}^{n}\frac{|\widetilde{y}_i|^2}{\lambda_i^2}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{\lambda_i}\right]^{-1}\right.\\
&\qquad\left.\times\left(c_0 - \frac{1}{n}\sum_{i=1}^{n}\frac{|\widetilde{c}_i|^2}{\lambda_i}\right)\right\}^{1/2}.
\end{aligned}
$$

Moreover, under further assumption (27), it follows that

$$
\begin{aligned}
\widehat{\mu}_{\text{GCV}} &= \widehat{\mu}_{\text{MLE}} = \widehat{\mu}_{\text{full}} = \frac{\widetilde{y}_1}{n} = \frac{1}{n}\sum_{i=1}^{n} y_i, \qquad (32)\\
\text{err}_{\text{GCV}} &= \frac{2.58}{n}\left\{\sum_{i=2}^{n}\frac{|\widetilde{y}_i|^2}{\lambda_i^2}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{\lambda_i}\right]^{-1}\right.\\
&\qquad\left.\times\left(1 - \frac{n}{\lambda_1}\right)\right\}^{1/2}. \qquad (33)
\end{aligned}
$$

In this case too, $\widehat{\mu}$ is simply the sample mean.

4 Integration Lattices and Shift Invariant Kernels

The preceding sections lay out an automatic Bayesian cubature algorithm whose computational cost is only $\mathcal{O}(n\log(n))$ if n function values are used. However, this algorithm relies on covariance kernel functions, C and node sets, $\{\boldsymbol{x}_i\}_{i=1}^n$ that satisfy assumptions (25). We also want to satisfy assumption (27). To facilitate the fast transform, n must be power of 2.

4.1 Extensible Integration Lattice Node Sets

The set of nodes used is defined by a shifted extensible integration lattice node sequence, which takes the form

$$
\boldsymbol{x}_i = \boldsymbol{h}\phi(i-1) + \boldsymbol{\Delta} \mod \mathbf{1}, \qquad i \in \mathbb{N}.
$$

Here, $\boldsymbol{h}$ is a d-dimensional generating vector of positive integers, $\boldsymbol{\Delta}$ is some point in $[0,1)^d$, often chosen at random, and $\{\phi(i)\}_{i=0}^n$ is the van der Corput sequence, defined by reflecting the binary digits of the integer about the decimal point, i.e.,

i	0	1	2	3	4	5	6	7	$\cdots$
i	0_2	1_2	10_2	11_2	100_2	101_2	110_2	111_2	$\cdots$
$\phi(i)$	$_2.0$	$_2.1$	$_2.01$	$_2.11$	$_2.001$	$_2.101$	$_2.011$	$_2.111$	$\cdots$
$\phi(i)$	0	0.5	0.25	0.75	0.125	0.625	0.375	0.875	$\cdots$

(34)

An example of 64 nodes is given in Figure 3. The even coverage of the unit cube is ensured by a well chosen generating vector. The choice of generating vector is typically done offline by computer search. See [6,14] for more on extensible integration lattices.

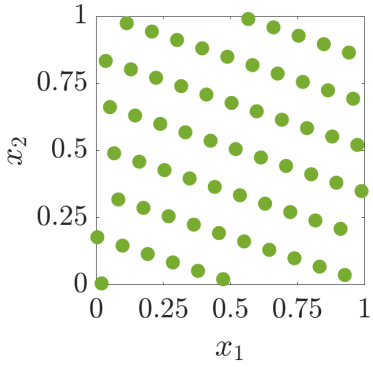

Fig. 3: Example of a shifted integration lattice node set in $d = 2$. This figure can be reproduced using `PlotPoints.m` in GAIL

4.2 Shift Invariant Kernels

The covariance functions C that match integration lattice node sets have the form

$$
C(\boldsymbol{t},\boldsymbol{x}) = K(\boldsymbol{t}-\boldsymbol{x} \mod \mathbf{1}). \qquad (35)
$$

This is called a *shift invariant kernel* because shifting both arguments of the covariance function by the same amount leaves the value unchanged. By a proper scaling of the kernel K it follows that assumption (27) is satisfied. Of course, K must also be of the form that ensures that C is symmetric and positive definite, as assumed in (4).

A family of shift invariant kernels is constructed via even degree Bernoulli polynomials:

$$
\begin{aligned}
C_{\boldsymbol{\theta}}(\boldsymbol{t},\boldsymbol{x}) &= \prod_{l=1}^{d}\Big[1 - (-1)^r\gamma B_{2r}(|x_l - t_l|)\Big],\\
&\forall \boldsymbol{t},\boldsymbol{x} \in [0,1]^d,\ \boldsymbol{\theta} = (r,\gamma),\ r \in \mathbb{N},\ \gamma > 0. \qquad (36)
\end{aligned}
$$

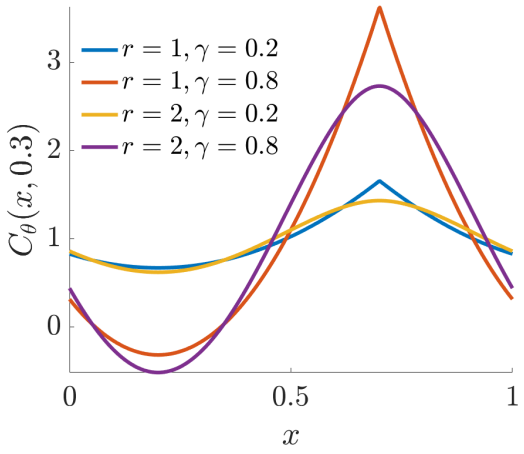

Fig. 4: Shift invariant kernel in 1D shifted by 0.3 to show the discontinuity. This figure can be reproduced using `plot_fourier_kernel.m` in GAIL.

Symmetric, periodic, positive definite kernels of this form appear in [6,10]. Bernoulli polynomials are described in [19, Chapter 24].

Larger r implies a greater degree of smoothness of the kernel. Larger γ implies greater fluctuations of the output with respect to the input. Plots of $C(\cdot, 0.3)$ are given in Figure 4 for various r and γ values.

4.3 Eigenvectors

For general shift-invariance covariance functions, the Gram matrix takes the form

$$\begin{aligned}\mathsf{C} &= \big(C(\boldsymbol{x}_i, \boldsymbol{x}_j)\big)_{i,j=1}^n \\ &= \Big(K(\boldsymbol{h}(\phi(i-1) - \phi(j-1)) \bmod \mathbf{1})\Big)_{i,j=1}^n.\end{aligned}$$

We now demonstrate that the eigenvector matrix for C is

$$\mathsf{V} = \Big(\mathrm{e}^{2\pi n\sqrt{-1}\phi(i-1)\phi(j-1)}\Big)_{i=1}^n. \tag{37}$$

Assumption (25b) follows automatically. Now, note that the k, j element of $\mathsf{V}^H\mathsf{V}$ is

$$\sum_{i=1}^n \mathrm{e}^{2\pi n\sqrt{-1}\phi(i-1)[\phi(j-1)-\phi(k-1)]}.$$

Noting that the sequence $\{\phi(i-1)\}_{i=1}^n$ is a re-ordering of $0, \dots, 1 - 1/n$ for n a power of 2, this sum may be re-written by replacing $\phi(i-1)$ by $(i-1)/n$:

$$\sum_{i=1}^n \mathrm{e}^{2\pi\sqrt{-1}(i-1)[\phi(j-1)-\phi(k-1)]}.$$

Since $\phi(j-1)-\phi(k-1)$ is some integer multiple of $1/n$, it follows that this sum is $n\delta_{j,k}$, where δ is the Kronecker delta function. This establishes that $\mathsf{V}^H = n\mathsf{V}^{-1}$ as in (24).

Next, let $\omega_{k,\ell}$ denote the k, ℓ element of $\mathsf{V}^H\mathsf{C}\mathsf{V}$, which is given by the double sum

$$\begin{aligned}\omega_{k,\ell} = \sum_{i,j=1}^n & K(\boldsymbol{h}(\phi(i-1) - \phi(j-1)) \bmod \mathbf{1}) \\ & \times \mathrm{e}^{-2\pi n\sqrt{-1}\phi(k-1)\phi(i-1)}\mathrm{e}^{2\pi n\sqrt{-1}\phi(j-1)\phi(l-1)}\end{aligned}$$

Noting that the sequence $\{\phi(i-1)\}_{i=1}^n$ is a re-ordering of $0, \dots, 1 - 1/n$ for n a power of 2, this sum may be re-written by replacing $\phi(i-1)$ by $(i-1)/n$ and $\phi(j-1)$ by $(j-1)/n$:

$$\begin{aligned}\omega_{k,\ell} = \sum_{i,j=1}^n & K\left(\boldsymbol{h}\left(\frac{i-j}{n}\right) \bmod \mathbf{1}\right) \\ & \times \mathrm{e}^{-2\pi\sqrt{-1}\phi(k-1)(i-1)}\mathrm{e}^{2\pi\sqrt{-1}(j-1)\phi(\ell-1)}.\end{aligned}$$

This sum also remains unchanged if i is replaced by $i+m$ and j is replaced by $j+m$ for any integer m:

$$\begin{aligned}\omega_{k,\ell} = \sum_{i,j=1}^n & K\left(\boldsymbol{h}\left(\frac{i-j}{n}\right) \bmod \mathbf{1}\right) \\ & \times \mathrm{e}^{-2\pi\sqrt{-1}\phi(k-1)(i+m-1)}\mathrm{e}^{2\pi\sqrt{-1}(j+m-1)\phi(\ell-1)} \\ & = \omega_{k,\ell}\mathrm{e}^{2\pi\sqrt{-1}m(\phi(\ell-1)-\phi(k-1))}.\end{aligned}$$

For this last equality to hold for all integers m, we must have $k = \ell$ or $\omega_{k,\ell} = 0$. Thus,

$$\begin{aligned}\omega_{k,\ell} &= \delta_{k,\ell}\sum_{i,j=1}^n K\left(\boldsymbol{h}\left(\frac{i-j}{n}\right) \bmod \mathbf{1}\right) \\ &\qquad \times \mathrm{e}^{-2\pi\sqrt{-1}(i-j)\phi(k-1)} \\ &= n\delta_{k,\ell}\sum_{i=1}^n K\left(\left(\frac{i\boldsymbol{h}}{n}\right) \bmod \mathbf{1}\right)\mathrm{e}^{-2\pi\sqrt{-1}i\phi(k-1)}.\end{aligned}$$

This establishes $\mathsf{V}^H\mathsf{C}\mathsf{V}$ as a diagonal matrix whose diagonal elements are n times the eigenvalues, i.e., $\lambda_k = \omega_{k,k}/n$. Furthermore, V is the matrix of eigenvectors, which satisfies assumption (25a).

4.4 Iterative Computation of the Fast Transform

Assumption (25a) is that computing $\mathsf{V}^H\boldsymbol{b}$ requires only $\mathcal{O}(n\log(n))$ operations. Recall that we assume that n is a power of 2. This can be accomplished by an iterative algorithm. Let $\mathsf{V}^{(n)}$ denote the $n \times n$ matrix V defined in (37). We show how to compute $\mathsf{V}^{(2n)H}\boldsymbol{b}$ quickly for all $\boldsymbol{b} \in \mathbb{R}^{2n}$ assuming that $\mathsf{V}^{(n)H}\boldsymbol{b}$ can be computed quickly for all $\boldsymbol{b} \in \mathbb{R}^n$.

From the definition of the van der Corput sequence in (34), it follows that

$$\phi(2i) = \phi(i)/2, \ \phi(2i+1) = [\phi(i)+1]/2, \quad i \in \mathbb{N}_0 \tag{38}$$

$$\phi(i+n) = \phi(i) + 1/(2n), \qquad i = 0, \dots, n-1, \tag{39}$$

$$n\phi(i) \in \mathbb{N}_0, \qquad i = 0, \dots, n-1, \tag{40}$$

still assuming that n is an integer power of two. Let $\widetilde{\boldsymbol{b}} = \mathsf{V}^{(2n)H}\boldsymbol{b}$ for some arbitrary $\boldsymbol{b} \in \mathbb{R}^{2n}$, and define

$$\boldsymbol{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_{2n} \end{pmatrix}, \quad \boldsymbol{b}^{(1)} = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}, \quad \boldsymbol{b}^{(2)} = \begin{pmatrix} b_{n+1} \\ \vdots \\ b_{2n} \end{pmatrix},$$

$$\widetilde{\boldsymbol{b}} = \begin{pmatrix} \widetilde{b}_1 \\ \vdots \\ \widetilde{b}_{2n} \end{pmatrix}, \quad \widetilde{\boldsymbol{b}}^{(1)} = \begin{pmatrix} b_1 \\ b_3 \\ \vdots \\ b_{2n-1} \end{pmatrix}, \quad \widetilde{\boldsymbol{b}}^{(2)} = \begin{pmatrix} b_2 \\ b_4 \\ \vdots \\ b_{2n} \end{pmatrix}.$$

It follows from these definitions and the definition of V in (37) that

$$\begin{aligned}
\widetilde{\boldsymbol{b}}^{(1)} &= \left(\sum_{j=1}^{2n} \mathrm{e}^{-4\pi n\sqrt{-1}\phi(2i-2)\phi(j-1)} b_j \right)_{i=1}^{n} \\
&= \left(\sum_{j=1}^{2n} \mathrm{e}^{-2\pi n\sqrt{-1}\phi(i-1)\phi(j-1)} b_j \right)_{i=1}^{n} \qquad \text{by (38)} \\
&= \left(\sum_{j=1}^{n} \mathrm{e}^{-2\pi n\sqrt{-1}\phi(i-1)\phi(j-1)} b_j \right)_{i=1}^{n} \\
&\qquad + \left(\sum_{j=1}^{n} \mathrm{e}^{-2\pi n\sqrt{-1}\phi(i-1)\phi(n+j-1)} b_{n+j} \right)_{i=1}^{n} \\
&= \mathsf{V}^{(n)H}\boldsymbol{b}^{(1)} + \Bigg(\mathrm{e}^{-\pi\sqrt{-1}\phi(i-1)} \\
&\qquad \times \sum_{j=1}^{n} \mathrm{e}^{-2\pi n\sqrt{-1}\phi(i-1)\phi(j-1)} b_{n+j} \Bigg)_{i=1}^{n} \qquad \text{by (39)} \\
&= \mathsf{V}^{(n)H}\boldsymbol{b}^{(1)} + \left(\mathrm{e}^{-\pi\sqrt{-1}\phi(i-1)} \right)_{i=1}^{n} \odot \left(\mathsf{V}^{(n)H}\boldsymbol{b}^{(2)} \right),
\end{aligned}$$

where $\odot$ denotes the Hadamard (term-by-term) product. By a similar argument,

$$\begin{aligned}
\widetilde{\boldsymbol{b}}^{(2)} &= \left(\sum_{j=1}^{2n} \mathrm{e}^{-4\pi n\sqrt{-1}\phi(2i-1)\phi(j-1)} b_j \right)_{i=1}^{n} \\
&= \left(\sum_{j=1}^{2n} \mathrm{e}^{-2\pi n\sqrt{-1}[\phi(i-1)+1]\phi(j-1)} b_j \right)_{i=1}^{n} \qquad \text{by (38)} \\
&= \left(\sum_{j=1}^{n} \mathrm{e}^{-2\pi n\sqrt{-1}[\phi(i-1)+1]\phi(j-1)} b_j \right)_{i=1}^{n} \\
&\qquad + \left(\sum_{j=1}^{n} \mathrm{e}^{-2\pi n\sqrt{-1}[\phi(i-1)+1]\phi(n+j-1)} b_{n+j} \right)_{i=1}^{n} \\
&= \mathsf{V}^{(n)H}\boldsymbol{b}^{(1)} + \Big(\mathrm{e}^{-\pi\sqrt{-1}[\phi(i-1)+1]} \\
&\qquad \times \sum_{j=1}^{n} \mathrm{e}^{-2\pi n\sqrt{-1}\phi(i-1)\phi(j-1)} b_{n+j} \Big)_{i=1}^{n} \\
&\qquad\qquad \text{by (39) and (40)} \\
&= \mathsf{V}^{(n)H}\boldsymbol{b}^{(1)} - \left(\mathrm{e}^{-\pi\sqrt{-1}\phi(i-1)} \right)_{i=1}^{n} \odot \left(\mathsf{V}^{(n)H}\boldsymbol{b}^{(2)} \right).
\end{aligned}$$

The computational cost to compute $\mathsf{V}^{(2n)H}\boldsymbol{b}$ is then twice the cost of computing $\mathsf{V}^{(n)H}\boldsymbol{b}^{(1)}$ plus $2n$ multiplications plus $2n$ additions. An inductive argument shows that $\mathsf{V}^{(n)H}\boldsymbol{b}$ requires only $\mathcal{O}(n\log(n))$ operations.

4.5 Overcoming Cancellation Error

For the kernels used in our computation, it may happen that n/λ_1 is close to 1. Thus, the term $1 - n/\lambda_1$, which appears in the credible interval widths, $\mathrm{err}_{\mathrm{MLE}}$, $\mathrm{err}_{\mathrm{full}}$, and $\mathrm{err}_{\mathrm{GCV}}$, may suffer from cancellation error. We can avoid this cancellation error by modifying how we compute the Gram matrix and its eigenvalues.

Define a new function $\mathring{C} := C - 1$, and its associated Gram matrix $\mathring{\mathsf{C}} = \mathsf{C} - \mathbf{1}\mathbf{1}^T$. Note that $\mathring{C}$ inherits the shift-invariant properties of C. Since $\mathbf{1}$ is the first eigenvector of C, it follows that the eigenvalues of $\mathring{\mathsf{C}}$ are $\mathring{\lambda}_1 = \lambda_1 - n, \lambda_2, \ldots, \lambda_n$. Moreover,

$$1 - \frac{n}{\lambda_1} = \frac{\lambda_1 - n}{\lambda_1} = \frac{\mathring{\lambda}_1}{\mathring{\lambda}_1 + n},$$

where now the right hand side is free of cancellation error.

We show how to compute $\mathring{C}$ without introducing round-off error. The covariance functions that we use are of product form, namely,

$$C(\boldsymbol{t}, \boldsymbol{x}) = \prod_{\ell=1}^{d} \left[1 + \mathring{C}_\ell(t_\ell, x_\ell) \right], \qquad \mathring{C}_\ell : [0,1]^2 \to \mathbb{R}.$$

Direct computation of $\mathring{C}(\boldsymbol{t},\boldsymbol{x}) = C(\boldsymbol{t},\boldsymbol{x}) - 1$ introduces cancellation error if the $\mathring{C}_\ell$ are small. So, we employ the iteration

$$\begin{aligned}\mathring{C}^{(1)} &= \mathring{C}_1(t_1,x_1),\\ \mathring{C}^{(\ell)} &= \mathring{C}^{(\ell-1)}[1+\mathring{C}_\ell(t_\ell,x_\ell)] + \mathring{C}_\ell(t_\ell,x_\ell), \qquad \ell = 2,\ldots,d,\\ \mathring{C}(\boldsymbol{t},\boldsymbol{x}) &= \mathring{C}^{(d)}.\end{aligned}$$

In this way, the Gram matrix $\mathring{\mathsf{C}}$, whose i,j-element is $\mathring{C}(\boldsymbol{x}_i,\boldsymbol{x}_j)$ can be constructed with minimal round-off error.

Computing the eigenvalues of $\mathring{\mathsf{C}}$ via the procedure given in (26) yields $\mathring{\lambda}_1 = \lambda_1 - n, \lambda_2, \ldots, \lambda_n$. The estimates of $\boldsymbol{\theta}$ are computed in terms of the eigenvalues of $\mathring{\mathsf{C}}$, so (28) and (31) become

$$\boldsymbol{\theta}_{\mathrm{MLE}} = \operatorname*{argmin}_{\boldsymbol{\theta}} \left[\log\left(\sum_{i=2}^{n} \frac{|\widetilde{y}_i|^2}{\lambda_i} \right) + \frac{1}{n}\sum_{i=1}^{n} \log(\lambda_i) \right], \tag{41a}$$

$$\boldsymbol{\theta}_{\mathrm{GCV}} = \operatorname*{argmin}_{\boldsymbol{\theta}} \left[\log\left(\sum_{i=2}^{n} \frac{|\widetilde{y}_i|^2}{\lambda_i^2} \right) - 2\log\left(\sum_{i=1}^{n} \frac{1}{\lambda_i} \right) \right], \tag{41b}$$

where $\lambda_1 = n + \mathring{\lambda}_1$. The widths of the credible intervals in (29), (30), and (33) become

$$\mathrm{err}_{\mathrm{MLE}} = \frac{2.58}{n} \sqrt{ \frac{\mathring{\lambda}_1}{\lambda_1} \sum_{i=2}^{n} \frac{|\widetilde{y}_i|^2}{\lambda_i} }, \tag{42a}$$

$$\mathrm{err}_{\mathrm{full}} = \frac{t_{n_j-1,0.995}}{n} \sqrt{ \frac{\mathring{\lambda}_1}{n-1} \sum_{i=2}^{n} \frac{|\widetilde{y}_i|^2}{\lambda_i} }, \tag{42b}$$

$$\mathrm{err}_{\mathrm{GCV}} = \frac{2.58}{n} \sqrt{ \frac{\mathring{\lambda}_1}{\lambda_1} \sum_{i=2}^{n} \frac{|\widetilde{y}_i|^2}{\lambda_i^2} \left[\frac{1}{n}\sum_{i=1}^{n} \frac{1}{\lambda_i} \right]^{-1} }. \tag{42c}$$

Since $\mathring{\lambda}_1 = \lambda_1 - n$ and $\lambda_1 \sim n$ it follows $\mathring{\lambda}_1/\lambda_1 \approx \mathring{\lambda}_1/(n-1)$ and is small for large n. Moreover, the credible intervals via empirical Bayes and full Bayes are similar, since $t_{n_j-1,0.995}$ is approximately 2.58. The computational steps for the improved, faster, automatic Bayesian cubature are detailed in Algorithm 2.

We summarize the results of this section and the previous one as follows:

Proposition 1 *Any periodic, symmetric, positive definite, shift-invariant covariance kernel of the form* (35) *scaled to satisfy* (27)*, when matched with rank-1 lattice data-sites, must satisfy assumptions* (25)*. The* fast Fourier transform *(FFT) can be used to expedite the estimates of* $\boldsymbol{\theta}$ *in* (41) *and the credible interval widths* (42) *in* $\mathcal{O}(n\log(n))$ *operations. The cubature,* $\widehat{\mu}$*, is just the sample mean.*

We have implemented the fast adaptive Bayesian cubature algorithm in MATLAB as part of the Guaranteed Adaptive Integration Library (GAIL) [2] as `cubBayesLattice_g`. This algorithm uses the kernel defined in (36) with $r = 1, 2$ and the periodizing variable transforms in Section 5.1. The rank-1 lattice node generator is taken from [17] (`exod2_base2_m20`).

Algorithm 2 Fast Automatic Bayesian Cubature

Require: a generator for the rank-1 Lattice sequence $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots$; a shift-invariant periodic kernel, C; a black-box function, f; an absolute error tolerance, $\varepsilon > 0$; the positive initial sample size, n_0; the maximum sample size $n_{\max}$

1: $n \leftarrow n_0$, $n' \leftarrow 0$, $\mathrm{err}_{\mathrm{CI}} \leftarrow \infty$
2: **while** $\mathrm{err}_{\mathrm{CI}} > \varepsilon$ and $n \le n_{\max}$ **do**
3: Generate $\{\boldsymbol{x}_i\}_{i=n'+1}^{n}$ and sample $\{f(\boldsymbol{x}_i)\}_{i=n'+1}^{n}$
4: Compute $\boldsymbol{\theta}$ by (41a) or (41b)
5: Compute $\mathrm{err}_{\mathrm{CI}}$ according to (42a), (42b), or (42c)
6: $n' \leftarrow n$, $n \leftarrow 2n'$
7: **end while**
8: Update sample size to compute $\widehat{\mu}$, $n \leftarrow n'$
9: Compute $\widehat{\mu}$, the approximate integral, according to (32)
10: **return** $\widehat{\mu}$, n and $\mathrm{err}_{\mathrm{CI}}$

5 Numerical Experiments

5.1 Periodizing Variable Transformations

The shift-invariant covariance kernels underlying our Bayesian cubature assume that the integrand has a degree of periodicity, with the smoothness assumed depending on the smoothness of the kernel. While integrands arising in practice may be smooth, they might not be periodic. Variable transformations can be used to ensure periodicity.

Suppose that the original integral has been expressed as

$$\mu := \int_{[0,1]^d} g(\boldsymbol{t})\,\mathrm{d}\boldsymbol{t}$$

where g has sufficient smoothness, but lacks periodicity. The Baker's transform,

$$\boldsymbol{\Psi} : \boldsymbol{x} \mapsto (\Psi(x_1),\ldots,\Psi(x_d)), \qquad \Psi(x) = 1 - 2\,|x - 1/2|\,, \tag{43}$$

allows us to write μ in the form of (1), where $f(\boldsymbol{x}) = g(\boldsymbol{\Psi}(\boldsymbol{x}))$.

A family of variable transforms take the form

$$\boldsymbol{\Psi} : \boldsymbol{x} \mapsto (\Psi(x_1),\ldots,\Psi(x_d)), \quad \Psi : [0,1] \mapsto [0,1],$$

which allows us to write μ in the form of (1) with

$$f(\boldsymbol{x}) = g(\boldsymbol{\Psi}(\boldsymbol{x})) \prod_{\ell=1}^{d} \Psi'(x_l).$$

If Ψ is sufficiently smooth, $\lim_{x\downarrow 0} x^{-r}\Psi'(x) = \lim_{x\uparrow 1}(x-1)^{-r}\Psi'(x) = 0$ for $r \in \mathbb{N}_0$, and $g \in C^{(r,\ldots,r)}[0,1]^d$, then f has continuous, periodic derivatives up to order r in

each direction. Examples of this kind of transform include [23]:

$$\text{Sidi's } C^1: \Psi(x) = x - \frac{\sin(2\pi x)}{2\pi}, \quad \Psi'(x) = 1 - \cos(2\pi x),$$

$$\text{Sidi's } C^2: \Psi(x) = \frac{8 - 9\cos(\pi x) + \cos(3\pi x)}{16}, \quad \Psi'(x) = \frac{3\pi[3\sin(\pi x) - \sin(3\pi x)]}{16}.$$

Periodizing variable transforms are used for the numerical examples below. In some cases, they can speed the convergence of the Bayesian cubature.

5.2 Test Results and Observations

Three integrals were evaluated using the GAIL algorithm `cubBayesLattice_g`: a multivariate normal probability, the Keister's example, and an option pricing example. Four different error tolerances, ε, were set for each example, with the tolerances chosen depending on the difficulty of the problem. The sequences $\{\boldsymbol{x}_i\}_{i=1}^{\infty}$ were the randomly shifted lattice node sequences supplied by GAIL. The accuracy of the algorithm differs depends on the shift. For each integral, each tolerance, and each of our stopping criteria—empirical Bayes, full Bayes, and generalized cross-validation—our algorithm was run for 100 different random shifts. For each test, the execution times were plotted against $|\mu - \widehat{\mu}|/\varepsilon$. We expect $|\mu - \widehat{\mu}|/\varepsilon$ to be no greater than one, but hope that it is not too much smaller than one, which would indicate a stopping criterion that is too conservative.

Figures 5 to 13 can be reproduced using the script `cubBayesLattice_guaranteed_plots.m` in GAIL.

Multivariate Normal Probability. This example was already introduced in Section 2.4, where we used the Matérn covariance kernel. Here we apply Sidi's C^2 periodization to f_{Genz} (23), choose $d = 3$ and $r = 2$. The simulation results for this example function are summarized in Figures 5, 6, and 7. In all cases, the Bayesian cubature returns an approximation within the prescribed error tolerance. We used the same setting as before with generic slow Bayesian cubature in Section 2.4 for comparison. For error threshold $\varepsilon = 10^{-5}$ with the empirical Bayes stopping criterion, our fast algorithm takes just under 0.01 second as shown in Figure 5 whereas the basic algorithm takes over 20 seconds as shown in Figure 2.

Amongst the three stopping criteria, GCV achieves the desired tolerance faster than the others. One can also observe from the figures, the credible intervals are in general much wider than the true error. This could

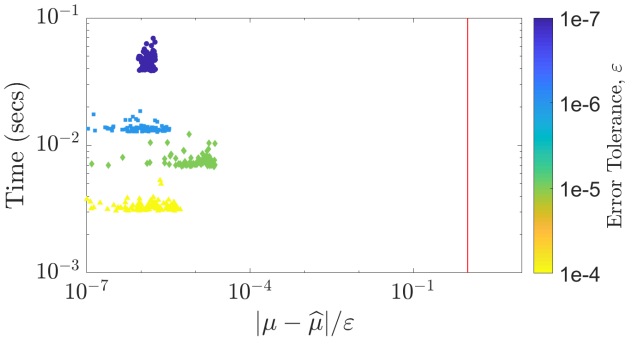

Fig. 5: Multivariate normal probability example using the empirical Bayes stopping criterion.

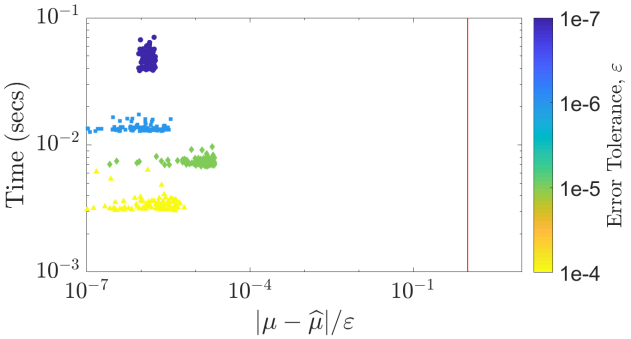

Fig. 6: Multivariate normal probability example using the full Bayes stopping criterion.

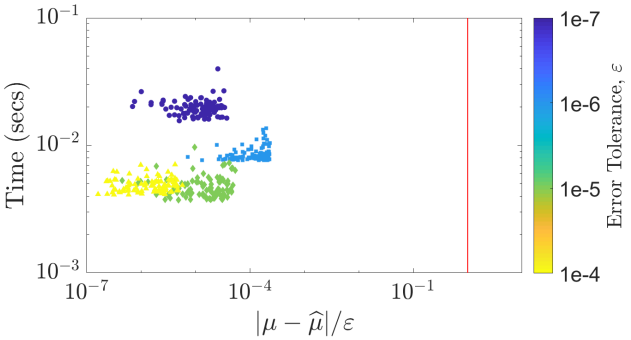

Fig. 7: Multivariate normal probability example using the GCV stopping criterion.

be due to the periodized integrand being smoother than the $r = 2$ kernel assumes. Perhaps one should consider smoother covariance kernels.

Keister's Example. This multidimensional integral function comes from [16] and is inspired by a physics application:

$$\mu = \int_{\mathbb{R}^d} \cos(\|\boldsymbol{t}\|) \exp(-\|\boldsymbol{t}\|^2)\, \mathrm{d}\boldsymbol{t} = \int_{[0,1]^d} f_{\text{Keister}}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x},$$

where

$$f_{\text{Keister}}(\boldsymbol{x}) = \pi^{d/2} \cos\left(\left\|\Phi^{-1}(\boldsymbol{x})/2\right\|\right),$$

and again Φ is the standard normal distribution. The true value of μ can be calculated iteratively in terms of a quadrature as follows:

$$\mu = \frac{2\pi^{d/2} I_c(d)}{\Gamma(d/2)}, \quad d = 1, 2, \ldots$$

where Γ denotes the gamma function, and

$$\begin{aligned}
I_c(1) &= \frac{\sqrt{\pi}}{2\exp(1/4)}, \\
I_s(1) &= \int_{x=0}^{\infty} \exp(-\boldsymbol{x}^T\boldsymbol{x}) \sin(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} \\
&= 0.4244363835020225, \\
I_c(2) &= \frac{1 - I_s(1)}{2}, \qquad I_s(2) = \frac{I_c(1)}{2} \\
I_c(j) &= \frac{(j-2)I_c(j-2) - I_s(j-1)}{2}, \qquad j = 3, 4, \ldots \\
I_s(j) &= \frac{(j-2)I_s(j-2) - I_c(j-1)}{2}, \qquad j = 3, 4, \ldots.
\end{aligned}$$

Figures 8, 9 and 10 summarize the numerical tests for this integral. We used the Sidi's C^1 periodization, dimension $d = 4$, and $r = 2$. As we can see, the GCV stopping criterion achieves faster results than the other stopping criteria, similarly to the multivariate normal case.

Option Pricing. The price of financial derivatives can often be modeled by high dimensional integrals. If the underlying asset is described in terms of a discretized geometric Brownian motion, then the fair price of the option is:

$$\mu = \int_{\mathbb{R}^d} \text{payoff}(\boldsymbol{z}) \frac{\exp(\frac{1}{2}\boldsymbol{z}^T\Sigma^{-1}\boldsymbol{z})}{\sqrt{(2\pi)^d \det(\Sigma)}}\, \mathrm{d}\boldsymbol{z} = \int_{[0,1]^d} f(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x},$$

where payoff$(\cdot)$ defines the discounted payoff of the option,

$$\Sigma = (T/d)\big(\min(j,k)\big)_{j,k=1}^{d} = \mathsf{L}\mathsf{L}^T,$$

$$f(\boldsymbol{x}) = \text{payoff}\left(\mathsf{L}\begin{pmatrix}\Phi^{-1}(x_1)\\ \vdots \\ \Phi^{-1}(x_d)\end{pmatrix}\right).$$

The Asian arithmetic mean call option has a payoff of the form

$$\text{payoff}(\boldsymbol{z}) = \max\left(\frac{1}{d}\sum_{j=1}^{d} S_j(\boldsymbol{z}) - K, 0\right) \mathrm{e}^{-rT},$$

where $S_j(\boldsymbol{z}) = S_0 \exp\big((r - \sigma^2/2)jT/d + \sigma\sqrt{T/d}z_j\big)$. Here, T denotes the time to maturity of the option, d the number of time steps, S_0 the initial price of the stock, r the interest rate, σ the volatility, and K the strike price.

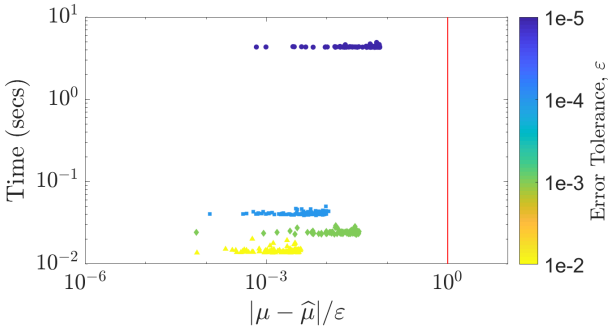

Fig. 8: Keister example using the empirical Bayes stopping criterion.

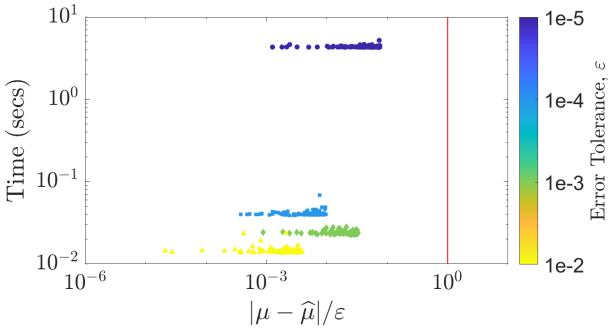

Fig. 9: Keister example using the full Bayes stopping criterion.

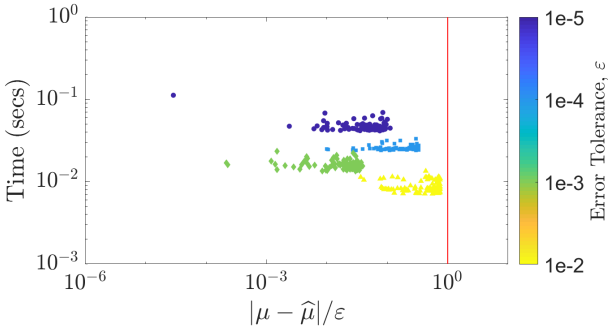

Fig. 10: Keister example using the GCV stopping criterion.

Figures 11, 12 and 13 summarize the numerical results for this example using $T = 1/4$, $d = 13$, $S_0 = 100$, $r = 0.05$, $\sigma = 0.5$, $K = 100$. Moreover, L is chosen to be the matrix of eigenvectors of Σ times the square root of the diagonal matrix of eigenvalues of Σ. Because the integrand has a kink caused by the max function, it does not help to use a periodizing transform that is very smooth. We choose the Baker's transform (43) and $r = 1$.

In summary, the Bayesian cubature algorithm computes the integral within the user-specified threshold in nearly all of the test cases. The rare exceptions occurred in the option pricing example for $\varepsilon = 10^{-4}$. Our

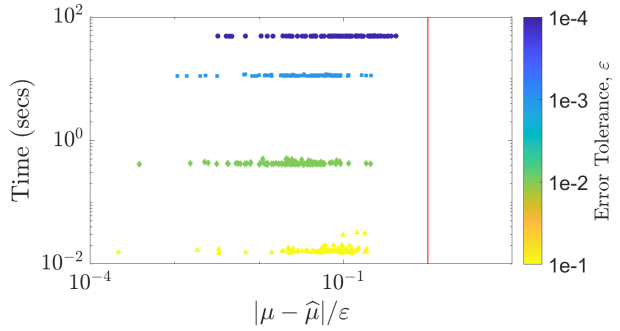

Fig. 11: Option pricing using the empirical Bayes stopping criterion.

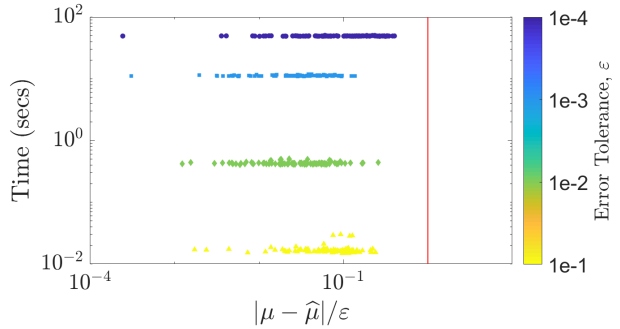

Fig. 12: Option pricing using the full Bayes stopping criterion.

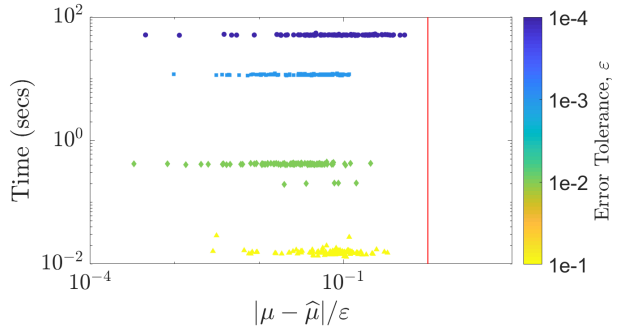

Fig. 13: Option pricing using the GCV stopping criterion.

algorithm used the maximum allowed sample size and still did not reach the stopping criterion $\mathrm{err}_{\mathrm{CI}} \le \varepsilon$, due to the complexity and high dimension of the integrand.

A noticeable aspect from the plots is how much the error bounds differ from the true error. For option pricing example, the error bound is not as conservative as it is for the multivariate normal and Keister examples. A possible reason is that the latter integrands are significantly smoother than the covariance kernel assumed. This is a matter for further investigation.

6 Discussion and Further Work

We have developed a fast, automatic Bayesian cubature that estimates a multidimensional definite integral within a user defined error tolerance. The stopping criteria arise from assuming the integrand to be a Gaussian process. There are three approaches: empirical Bayes, full Bayes, and generalized cross-validation. The computational cost of the automatic Bayesian cubature can be dramatically reduced if the covariance kernel matches the nodes. One such match in practice is rank-1 lattice nodes and shift-invariant kernels. The matrix-vector multiplications can be accomplished using the fast Fourier Transform. The performance of our automatic Bayesian cubature are illustrated using three integration problems.

Digital sequences and digital shift and/or scramble invariant kernels have the potential of being another match that satisfies the conditions in Section 3. The fast transform would correspond to a fast Walsh transform.

One should be able to adapt our Bayesian cubature to control variates, i.e., assuming

$$f = \mathcal{GP}\left(\beta_0 + \beta_1 g_1 + \cdots + \beta_p g_p, s^2 C\right),$$

for some choice of $g_1, \ldots, g_p$ whose integrals are known, and some parameters $\beta_0, \ldots, \beta_p$ in addition to s and C. The efficacy of this approach has not yet been explored.

Acknowledgements This research was supported in part by the National Science Foundation grants DMS-1522687 and DMS-1638521 (SAMSI). The authors would like to thank the organizers of the SAMSI-Lloyds-Turing Workshop on Probabilistic Numerical Methods, where a preliminary version of this work was discussed. The authors also thank Chris Oates and Sou-Cheng Choi for valuable comments.

References

1. Briol, F.X., Oates, C.J., Griolami, M., Osborne, M.A., Sejdinovic, D.: Probabilistic integration: A role in statistical computation? Statist. Sci. (2018+). To appear
2. Choi, S.C.T., Ding, Y., Hickernell, F.J., Jiang, L., Jiménez Rugama, Ll.A., Li, D., Jagadeeswaran, R., Tong, X., Zhang, K., Zhang, Y., Zhou, X.: GAIL: Guaranteed Automatic Integration Library (versions 1.0–2.2). MATLAB software (2013–2017). URL `http://gailgithub.github.io/GAIL_Dev/`
3. Cools, R., Nuyens, D. (eds.): Monte Carlo and Quasi-Monte Carlo Methods: MCQMC, Leuven, Belgium, April 2014, *Springer Proceedings in Mathematics and Statistics*, vol. 163. Springer-Verlag, Berlin (2016)
4. Craven, P., Wahba, G.: Smoothing noisy data with spline functions: Estimating the correct degree of smoothing by the method of generalized cross-validation. Numer. Math. **31**, 307–403 (1979)
5. Diaconis, P.: Bayesian numerical analysis. In: S.S. Gupta, J.O. Berger (eds.) Statistical Decision Theory and Related Topics IV, Papers from the 4th Purdue

Symp., West Lafayette, Indiana 1986, vol. 1, pp. 163–175. Springer-Verlag, New York (1988)
6. Dick, J., Kuo, F., Sloan, I.H.: High dimensional integration — the Quasi-Monte Carlo way. Acta Numer. **22**, 133–288 (2013). DOI 10.1017/S0962492913000044
7. Dick, J., Pillichshammer, F.: Digital Nets and Sequences: Discrepancy Theory and Quasi-Monte Carlo Integration. Cambridge University Press, Cambridge (2010)
8. Genz, A.: Comparison of methods for the computation of multivariate normal probabilities. Computing Science and Statistics **25**, 400–405 (1993)
9. Golub, G.H., Heath, M., Wahba, G.: Generalized cross-validation as a method for choosing a good ridge parameter. Technometrics **21**, 215–223 (1979)
10. Hickernell, F.J.: Quadrature error bounds with applications to lattice rules. SIAM J. Numer. Anal. **33**, 1995–2016 (1996). Corrected printing of Sections 3-6 in ibid., **34** (1997), 853–866
11. Hickernell, F.J.: The trio identity for quasi-Monte Carlo error analysis. In: P. Glynn, A. Owen (eds.) Monte Carlo and Quasi-Monte Carlo Methods: MCQMC, Stanford, USA, August 2016, Springer Proceedings in Mathematics and Statistics, pp. 13–37. Springer-Verlag, Berlin (2018). ArXiv:1702.01487
12. Hickernell, F.J., Jiménez Rugama, Ll.A.: Reliable adaptive cubature using digital sequences. In: Cools and Nuyens [3], pp. 367–383. ArXiv:1410.8615 [math.NA]
13. Hickernell, F.J., Jiménez Rugama, Ll.A., Li, D.: Adaptive quasi-Monte Carlo methods for cubature. In: J. Dick, F.Y. Kuo, H. Woźniakowski (eds.) Contemporary Computational Mathematics — a celebration of the 80th birthday of Ian Sloan, pp. 597–619. Springer-Verlag (2018). DOI 10.1007/978-3-319-72456-0
14. Hickernell, F.J., Niederreiter, H.: The existence of good extensible rank-1 lattices. J. Complexity **19**, 286–300 (2003)
15. Jiménez Rugama, Ll.A., Hickernell, F.J.: Adaptive multidimensional integration based on rank-1 lattices. In: Cools and Nuyens [3], pp. 407–422. ArXiv:1411.1966
16. Keister, B.D.: Multidimensional quadrature algorithms. Computers in Physics **10**, 119–122 (1996). DOI 10.1063/1.168565
17. Nuyens, D.: URL `https://people.cs.kuleuven.be/~dirk.nuyens/qmc-generators/genvecs/exod2_base2_m20.txt`
18. O'Hagan, A.: Bayes-Hermite quadrature. J. Statist. Plann. Inference **29**, 245–260 (1991). DOI 10.1016/0378-3758(91)90002-V
19. Olver, F.W.J., Lozier, D.W., Boisvert, R.F., Clark, C.W., Dalhuis, A.B.O.: Digital library of mathematical functions (2018). URL `http://dlmf.nist.gov/`
20. Rasmussen, C.E., Ghahramani, Z.: Bayesian Monte Carlo. In: S. Thrun, L.K. Saul, K. Obermayer (eds.) Advances in Neural Information Processing Systems, vol. 15, pp. 489–496. MIT Press (2003)
21. Rasmussen, C.E., Williams, C.: Gaussian Processes for Machine Learning. MIT Press, Cambridge, Massachusetts (2006). (online version at `http://www.gaussianprocess.org/gpml/`)
22. Ritter, K.: Average-Case Analysis of Numerical Problems, *Lecture Notes in Mathematics*, vol. 1733. Springer-Verlag, Berlin (2000)
23. Sidi, A.: Further extension of a class of periodizing variable transformations for numerical integration. J. Comput. Appl. Math. **221**, 132–149 (2008)
24. Sloan, I.H., Joe, S.: Lattice Methods for Multiple Integration. Oxford University Press, Oxford (1994)
25. Wahba, G.: Spline Models for Observational Data, *CBMS-NSF Regional Conference Series in Applied Mathematics*, vol. 59. SIAM, Philadelphia (1990)