Fast Automatic Bayesian Cubature using Matching Kernels and Designs

Jagadeeswaran R.
Adviser: Prof. Fred J Hickernell

Department of Applied Mathematics, Illinois Institute of Technology
jrathin1@iit.edu

Thesis Defense • Oct 22, 2019

Contents

1 Introduction
   - Automatic cubature algorithm
2 Bayesian Cubature
   - Posterior Error
   - Parameter estimation
   - Example with Matérn kernel
3 Faster
   - Fast transform kernel
   - Shape param θ_{EB}
   - Error Bound
4 Lattice
   - Shift invariant covariance kernels
   - Rank-1 Lattice points
   - Periodization transforms
5 Sobol'
   - Sobol' Nets and Walsh Kernels
6 Demonstrate
   - Cancellation error
   - MVN
   - Keister integral
   - Option pricing
7 Conclusion

Outline

1 Introduction

2 Bayesian Cubature

3 Faster

4 Lattice

5 Sobol'

6 Demonstrate

7 Conclusion

Numerical Integration

A fundamental problem in various fields, including finance, machine learning and statistics,

$$\mu = \int_{\mathbb{R}^d} g(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_{[0,1]^d} f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \mathbb{E}[f(\boldsymbol{X})], \quad \text{where} \quad \boldsymbol{X} \sim \mathcal{U}[0,1]^d \qquad (1)$$

by a cubature rule $\hat{\mu}_n := w_0 + \sum_{j=1}^{n} f(\boldsymbol{x}_j) w_j$

using points $\{\boldsymbol{x}_j\}_{j=1}^n$ and associated weights w_j.

The goal of this work is to

- Develop an automatic algorithm for integration
- Assume f is drawn from a Gaussian process
  - Need to estimate the mean and Covariance kernel
- Parameter estimation (MLE, Cross validation) is expensive in general
  - Use points and kernel for which it is cheap
- Use an extensible point-set and an algorithm that allows extending points
- Determine n such that, given ϵ, $|\mu - \hat{\mu}_n| \leqslant \epsilon$

procedure AUTOBAYESCUBATURE(f,ϵ)

Require: a generator for the sequence $x_1, x_2, \ldots$; a black-box function, f; an absolute error tolerance, $\varepsilon > 0$; the positive initial sample size, n_0; the maximum sample size n_{max}

```
    n ← n_0, n′ ← 0, err_n ← ∞
    while err_n > ε and n ⩽ n_max do
        Generate {x_i}_{i=n′+1}^n and sample {f(x_i)}_{i=n′+1}^n,
        Compute parameters, compute error bound err_CI
        n′ ← n, n ← 2 × n′
    end while
    Sample size to compute μ̂, n ← n′
    Compute approximate μ̂_n, the approximate integral
    return μ̂_n                                   ▷ Integral estimate μ̂_n
end procedure
```

Problem:

- How to choose $\{x_i\}_{i=1}^n$, and $\{w_i\}_{i=1}^n$ to make $|\mu - \hat{\mu}_n|$ small? what is err_n? (Bayesian posterior error)
- How to find n such that $|\mu - \hat{\mu}_n| \leqslant \text{err}_\text{CI} \leqslant \epsilon$? (automatic cubature)

Outline

1 Introduction

2 Bayesian Cubature

3 Faster

4 Lattice

5 Sobol'

6 Demonstrate

7 Conclusion

Bayesian posterior error

Assume random $f \sim \mathcal{GP}(m, s^2 C_\theta)$, a Gaussian process with mean m and covariance kernel, $s^2 C_\theta$, $C_\theta : [0,1] \times [0,1] \to \mathbb{R}$.

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

$$\boldsymbol{c} = \left(\int_{[0,1]} C_\theta(\boldsymbol{x}_i, \boldsymbol{t}) \mathrm{d}\boldsymbol{t} \right)_{i=1}^n, \quad \mathsf{C} = \Big(C_\theta(\boldsymbol{x}_i, \boldsymbol{x}_j) \Big)_{i,j=1}^n$$

$$\mu - \hat{\mu}_n \big| \boldsymbol{y} \sim \mathcal{N}\Big(-w_0 + m(1 - \boldsymbol{1}^T \mathsf{C}^{-1}\boldsymbol{c}) + \boldsymbol{y}^T(\mathsf{C}^{-1}\boldsymbol{c} - \boldsymbol{w}), \quad s^2(c_0 - \boldsymbol{c}^T \mathsf{C}^{-1}\boldsymbol{c}) \Big)$$

where $\boldsymbol{y} = \big(f(\boldsymbol{x}_i) \big)_{i=1}^n$. Moreover m, s and θ needs to be inferred.

$$\hat{\mu}_n = w_0 + \sum_{i=1}^n w_i f(\boldsymbol{x}_i) = w_0 + \boldsymbol{w}^T \boldsymbol{y}$$

$$\text{Choosing} \quad w_0 = m(1 - \boldsymbol{1}^T \mathsf{C}^{-1}\boldsymbol{c}), \; \boldsymbol{w} = \mathsf{C}^{-1}\boldsymbol{c}, \quad \text{makes error unbiased}$$

Diaconis (1988), O'Hagan (1991), Ritter (2000), Rasmussen (2003), Briol et al. (2018+), Traub et al. (1988) and others

Parameter estimation - Empirical Bayes

The log-likelihood of the parameters given the data $\boldsymbol{y} = (f(\boldsymbol{x}_i))_{i=1}^n$ is :

$$l(s, \theta|y) = \log \left[\frac{1}{\sqrt{(2\pi)^n \det(s^2 \mathsf{C})}} \exp\left(-\frac{1}{2} s^{-2} (\boldsymbol{y} - m\mathbf{1})^T \mathsf{C}^{-1} (\boldsymbol{y} - m\mathbf{1}) \right) \right]$$

Maximising w.r.t m and then s^2, further with θ :

$$m_{\text{EB}} = \frac{\mathbf{1}^T \mathsf{C}^{-1} \boldsymbol{y}}{\mathbf{1}^T \mathsf{C}^{-1} \mathbf{1}}, \quad s^2_{\text{EB}} = \frac{1}{n} (\boldsymbol{y} - m_{\text{EB}}\mathbf{1})^T \mathsf{C}^{-1} (\boldsymbol{y} - m_{\text{EB}}\mathbf{1}), \qquad \text{(Explicit)}$$

$$\theta_{\text{EB}} = \operatorname*{argmin}_{\theta} \log \left(\frac{1}{2n} \log(\det \mathsf{C}) + \log(s_{\text{EB}}) \right) \qquad \text{(numeric)}$$

$$\hat{\mu}_{\text{EB}} = \left(\frac{(1 - \mathbf{1}^T \mathsf{C}^{-1} \boldsymbol{c})\mathbf{1}}{\mathbf{1}^T \mathsf{C}^{-1} \mathbf{1}} + \boldsymbol{c} \right)^T \mathsf{C}^{-1} \boldsymbol{y}, \qquad \text{(Explicit)}$$

Why do we need θ_{EB}? The sample space spanned by C_θ customized to have the integrand f in the middle.

Parameter estimation - Full Bayes

Treat m and s as hyper-parameters with a non-informative, conjugate prior, namely $\rho_{m,s^2}(\xi,\lambda) \propto 1/\lambda$. Then the posterior density for the integral μ given the data is :

$$\rho_\mu(z|\boldsymbol{f}=\boldsymbol{y}) \propto \int_0^\infty \int_{-\infty}^\infty \rho_\mu(z|\boldsymbol{f}=\boldsymbol{y}, m=\xi, s^2=\lambda)\rho_{\boldsymbol{f}}(\boldsymbol{y}|\xi,\lambda)\rho_{m,s^2}(\xi,\lambda)\,\mathrm{d}\xi\mathrm{d}\lambda$$

$$\propto \left(1+\frac{1}{n-1}\frac{(z-\mu_{\text{full}})^2}{\hat{\sigma}^2_{\text{full}}}\right)^{-n/2}$$

Where :

$$\mu_{\text{full}} = \mu_{\text{EB}}$$

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

$$\mathbb{P}_f\left[|\mu - \hat{\mu}_{\text{full}}| \leqslant \text{err}_{\text{full}}\right] = 99\%,$$

$$\text{err}_{\text{full}} := t_{n_j-1,0.995}\hat{\sigma}_{\text{full}} > \text{err}_{\text{EB}}$$

Parameter estimation - Generalized Cross validation

Let $\tilde{y}_i = \mathbb{E}[f(\boldsymbol{x}_i) | \boldsymbol{f}_{-i} = \boldsymbol{y}_{-i}]$. The cross-validation criterion, which is to be minimized, is sum of squares of the difference between these conditional expectations and the observed values: :

$$\mathsf{CV} = \sum_{i=1}^{n} (y_i - \tilde{y}_i)^2 = \sum_{i=1}^{n} \left(\frac{\zeta_i}{a_{ii}} \right)^2, \quad \text{where } \zeta = \mathsf{C}^{-1}(\boldsymbol{y} - m\mathbf{1}),$$

$$a_{ii} \text{ are diagonal elems of } \mathsf{C}^{-1} = \begin{pmatrix} a_{ii} & \boldsymbol{A}^T_{-i,i} \\ \boldsymbol{A}_{-i,i} & \mathsf{A}_{-i,-i} \end{pmatrix}$$

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

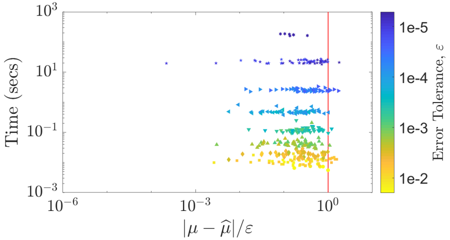
$$\theta_{\mathsf{GCV}} = \underset{\theta}{\mathsf{argmin}} \left\{ \log\left(\boldsymbol{y}^T \left[\mathsf{C}^{-2} - \frac{\mathsf{C}^{-2}\mathbf{1}\mathbf{1}^T\mathsf{C}^{-2}}{\mathbf{1}^T\mathsf{C}^{-2}\mathbf{1}} \right] \boldsymbol{y} \right) - 2\log\left(\mathsf{trace}(\mathsf{C}^{-1})\right) \right\}$$

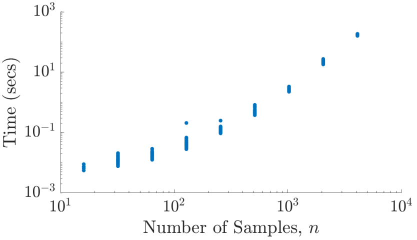
$$s^2_{\mathsf{GCV}} := \boldsymbol{y}^T \left[\mathsf{C}^{-2} - \frac{\mathsf{C}^{-2}\mathbf{1}\mathbf{1}^T\mathsf{C}^{-2}}{\mathbf{1}^T\mathsf{C}^{-2}\mathbf{1}} \right] \boldsymbol{y} \left[\mathsf{trace}(\mathsf{C}^{-1})\right]^{-1}, \quad m_{\mathsf{GCV}} := \frac{\mathbf{1}^T\mathsf{C}^{-2}\boldsymbol{y}}{\mathbf{1}^T\mathsf{C}^{-2}\mathbf{1}}.$$

Multivariate Gaussian integration with Matérn kernel

$$\text{Matérn covariance kernel: } C_\theta(\boldsymbol{x}, \boldsymbol{t}) = \prod_{\ell=1}^{d} \exp(-\theta|x_\ell - t_\ell|)(1 + \theta|x_\ell - t_\ell|) \quad (2)$$

$$\text{Multivariate Gaussian: } \mu = \int_{(\boldsymbol{a},\boldsymbol{b})} \frac{\exp\left(-\frac{1}{2}\boldsymbol{t}^T\Sigma^{-1}\boldsymbol{t}\right)}{\sqrt{(2\pi)^d \det(\Sigma)}} \, \mathrm{d}\boldsymbol{t}. \quad (3)$$





Problem: Computation time (in seconds) increases rapidly, so it's not practical to use more than 4000 points in the cubature.

Outline

1 Introduction

2 Bayesian Cubature

3 Faster

4 Lattice

5 Sobol'

6 Demonstrate

7 Conclusion

Fast transform kernel

Choose the kernel C_θ and $\{\boldsymbol{x}_i\}_{i=1}^n$, so the Gram matrix $\mathsf{C} = \big(C_\theta(\boldsymbol{x}_i, \boldsymbol{x}_j)\big)_{i,j=1}^n$ has:

$$\mathsf{C} = (\boldsymbol{C}_1, ..., \boldsymbol{C}_n) = \frac{1}{n}\mathsf{V}\Lambda\mathsf{V}^H, \quad \mathsf{V} = (\boldsymbol{v}_1, ..., \boldsymbol{v}_n)^T = (\boldsymbol{V}_1, ..., \boldsymbol{V}_n), \boldsymbol{V}_1 = \boldsymbol{v}_1 = \boldsymbol{1},$$

$$\Lambda = \mathrm{diag}(\boldsymbol{\lambda}), \quad \boldsymbol{\lambda} = (\lambda_1, ..., \lambda_n),$$

$$\text{Then,} \qquad \mathsf{V}^H\boldsymbol{C}_1 = \mathsf{V}^H\left(\frac{1}{n}\mathsf{V}\Lambda\boldsymbol{v}_1^*\right) = \Lambda\boldsymbol{1} = \big(\lambda_1, \ldots, \lambda_n\big)^T = \boldsymbol{\lambda}$$

C_θ is a fast transform kernel, if

$$\mathsf{V} \text{ may be identified analytically,} \tag{4a}$$

$$\boldsymbol{v}_1 = \boldsymbol{V}_1 = \boldsymbol{1}, \tag{4b}$$

$$\text{Computing } \mathsf{V}^H\boldsymbol{b} \text{ requires only } \mathcal{O}(n\log n) \text{ operations } \forall \boldsymbol{b}. \tag{4c}$$

The covariance kernel may also be normalized

$$\int_{[0,1]^d} C(\boldsymbol{t}, \boldsymbol{x})\,\mathrm{d}\boldsymbol{t} = 1 \qquad \forall \boldsymbol{x} \in [0,1]^d, \text{ leading to } c_0 = 1 \text{ and } \boldsymbol{c} = \boldsymbol{1}. \tag{5}$$

Using the fast transform, $\boldsymbol{a}^T\mathsf{C}^p\boldsymbol{b} = \dfrac{1}{n}\boldsymbol{a}^T\mathsf{V}\Lambda^p\mathsf{V}^H\boldsymbol{b} = \dfrac{1}{n}\widetilde{\boldsymbol{a}}^H\Lambda^p\widetilde{\boldsymbol{b}} = \dfrac{1}{n}\sum_{i=1}^n \lambda_i^p \widetilde{a}_i^* \widetilde{b}_i.$

Faster parameters estimation

MLE and GCV estimates of θ made faster by using the properties of the fast transform kernel:

$$\theta_{\text{EB}} = \underset{\theta}{\operatorname{argmin}} \left[\log \left(\sum_{i=2}^{n} \frac{|\widetilde{y}_i|^2}{\lambda_i} \right) + \frac{1}{n} \sum_{i=1}^{n} \log(\lambda_i) \right], \tag{6a}$$

$$\theta_{\text{GCV}} = \underset{\theta}{\operatorname{argmin}} \left[\log \left(\sum_{i=2}^{n} \frac{|\widetilde{y}_i|^2}{\lambda_i^2} \right) - 2 \log \left(\sum_{i=1}^{n} \frac{1}{\lambda_i} \right) \right], \tag{6b}$$

Also,

$$m_{\text{EB}} = m_{\text{GCV}} = \frac{1}{n} \sum_{i=1}^{n} y_i, \quad s_{\text{EB}}^2 = \frac{1}{n} \sum_{i=2}^{n} \frac{|\widetilde{y}_i|^2}{\lambda_i}, \quad s_{\text{GCV}}^2 = \frac{1}{n} \sum_{i=2}^{n} \frac{|\widetilde{y}_i|^2}{\lambda_i^2} \left[\sum_{i=1}^{n} \frac{1}{\lambda_i} \right]^{-1}$$

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

$$\tilde{\boldsymbol{y}} = (\widetilde{y}_i)_{i=1}^n = \mathsf{V}^T \boldsymbol{y}, \qquad \boldsymbol{\lambda} = (\lambda_i)_{i=1}^n = \mathsf{V}^T \boldsymbol{C}_1, \quad \text{where } \boldsymbol{C}_1 = \big(C(\boldsymbol{x}_i, \boldsymbol{x}_1)\big)_{i=1}^n$$

$\mathcal{O}(n \log n)$ operations to compute $\tilde{\boldsymbol{y}}$ and $\hat{\boldsymbol{\lambda}}$, So the θ_{EB}

Computing the error bound $\mathrm{err}_{\mathrm{CI}}$ and $\hat{\mu}$ faster

Using the properties of the fast transform kernel, the error bound $\mathrm{err}_{\mathrm{n}}$ can be computed faster

$$\mathrm{err}_{\mathrm{EB}} = \frac{2.58}{n}\left\{\sum_{i=2}^{n}\frac{|\tilde{y}_i|^2}{\lambda_i}\left(1-\frac{n}{\lambda_1}\right)\right\}^{1/2} \tag{7a}$$

$$\mathrm{err}_{\mathrm{full}} = t_{n_j-1,0.995}\left\{\frac{1}{n(n-1)}\sum_{i=2}^{n}\frac{|\tilde{y}_i|^2}{\lambda_i}\left(\frac{\lambda_1}{n}-1\right)\right\}^{1/2}, \tag{7b}$$

$$\mathrm{err}_{\mathrm{GCV}} = \frac{2.58}{n}\left\{\sum_{i=2}^{n}\frac{|\tilde{y}_i|^2}{\lambda_i^2}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{\lambda_i}\right]^{-1}\times\left(1-\frac{n}{\lambda_1}\right)\right\}^{1/2}. \tag{7c}$$

Similarly, $\hat{\mu}$ can be computed faster

$$\hat{\mu}_{\mathrm{EB}} = \hat{\mu}_{\mathrm{full}} = \hat{\mu}_{\mathrm{GCV}} = \boldsymbol{w}^T\boldsymbol{y} = \boxed{\sum_{i=1}^{n}\frac{y_i}{n}},$$

where

$$\tilde{\boldsymbol{y}} = \mathsf{V}^T\boldsymbol{y}, \quad \boldsymbol{\lambda} = \mathsf{V}^T\boldsymbol{C}_1, \text{ where } \boldsymbol{C}_1 = (C(\boldsymbol{x}_i,\boldsymbol{x}_1))_{i=1}^{n}$$

$\mathcal{O}(n\log n)$ operations to compute the err. $\mathcal{O}(n)$ operations to compute the $\hat{\mu}$

Outline

1 Introduction

2 Bayesian Cubature

3 Faster

4 Lattice

5 Sobol'

6 Demonstrate

7 Conclusion

Lattice nodes and shift invariant covariance kernels

Theorem

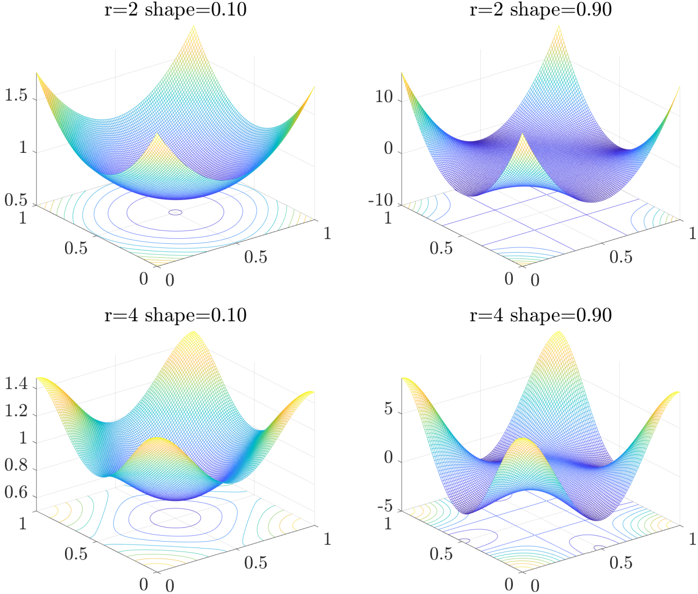
Let C_θ be any symmetric, positive definite, shift-invariant covariance kernel of the form $C_\theta(\boldsymbol{x},\boldsymbol{t}) = K_\theta(\boldsymbol{x}-\boldsymbol{t} \bmod \boldsymbol{1})$, where K_θ has period one in every variable. Furthermore, let K_θ be scaled to satisfy (5). *When matched with rank-1 lattice data-sites, C_θ must satisfy assumptions* (4). *The cubature, $\hat{\mu}$, is just the sample mean. The fast Fourier transform (FFT) can be used to expedite the estimates of θ in* (6) *and the credible interval widths* (7) *in $\mathcal{O}(n \log n)$ operations.*

$$C_\theta(\boldsymbol{x},\boldsymbol{t}) = \prod_{\ell=1}^{d}\left[1 - \eta_\ell \frac{(2\pi\sqrt{-1})^r}{r!} B_r(|x_\ell - t_\ell|)\right], \quad r \in 2\mathbb{N}, \quad \eta_\ell > 0, \quad \theta = (\boldsymbol{\eta}, r)$$

where B_r is Bernoulli polynomial of order r (Olver et al., 2013). We call C_θ, Fourier kernel. Also this kernel has:

$$c_0 = \int_{[0,1]^2} C_\theta(\boldsymbol{x},\boldsymbol{t})\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{t} = 1, \qquad \boldsymbol{c} = \left(\int_{[0,1]} C_\theta(\boldsymbol{x}_i,\boldsymbol{t})\mathrm{d}\boldsymbol{t}\right)_{i=1}^{n} = \boldsymbol{1}.$$

Fourier kernel



Rank-1 Lattice rules : low discrepancy point set

Given the "generating vector" $\boldsymbol{h}$, the construction of n - Rank-1 lattice points (Dick and Pillichshammer, 2010) is given by

$$\mathsf{L}_{n,\boldsymbol{h}} := \{\boldsymbol{x}_i := \boldsymbol{h}\phi(i-1) \ \textbf{mod}\ 1; \quad i = 1, \ldots, n\} \tag{8}$$

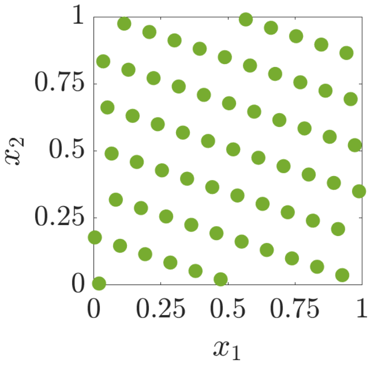
where $\boldsymbol{h}$ is a *generalized Mahler integer* (∞ digit expression) (Hickernell and Niederreiter, 2003) also called generating vector. $\phi(i)$ is the Van der Corput sequence in base 2. Then the Lattice rule approximation is

$$\frac{1}{n}\sum_{k=1}^{n} f\left(\left\{\frac{k\boldsymbol{h}}{n} + \boldsymbol{\Delta}\right\}_1\right)$$

where $\{.\}_1$ the fractional part, i.e, *modulo 1* operator and $\boldsymbol{\Delta}$ a random shift.

Extensible integration lattices : The number of points in the node set can be increased while retaining the existing points.(Hickernell and Niederreiter, 2003)

Rank-1 Lattice points in $d = 2$



Shift invariant kernel + Lattice points = '*Symmetric circulant kernel*' matrix

The shift invariant kernel with rank-1 Lattice points

- Satisfies all the requirements to be a fast transform kernel
- Fast Bayesian transform = fast Fourier transform
- Complexity of fast Fourier transform is $\mathcal{O}(n \log n)$
- No need to compute the kernel matrix C explicitly, so $\mathcal{O}(n^2)$ memory not required
- There are no matrix inversions, no matrix multiplications
- Factorization of matrix C does not need any computations.

  where V is just the Fourier coefficient matrix: $\mathsf{V} = \left(e^{2\pi n\sqrt{-1}(i-1)(j-1)}\right)_{i=1}^{n}$

Periodization transforms

Suppose the original integral is

$$\mu := \int_{(\boldsymbol{a},\boldsymbol{b})^d} g(\boldsymbol{t})\,\mathrm{d}\boldsymbol{t}, \quad \text{where } g \text{ is smooth, not periodic.}$$

The Baker's transform, the tent transform,

$$\boldsymbol{\Psi} : \boldsymbol{x} \mapsto (\Psi(x_1), \ldots, \Psi(x_d)), \quad \Psi(x) = 1 - 2\,|x - 1/2|\,, \quad f(\boldsymbol{x}) = g(\boldsymbol{\Psi}(\boldsymbol{x})).$$

A family of smoother variable transforms:

$$\boldsymbol{\Psi} : \boldsymbol{x} \mapsto (\Psi(x_1), \ldots, \Psi(x_d)), \quad \Psi : [0,1] \mapsto [0,1], \;\; f(\boldsymbol{x}) = g(\boldsymbol{\Psi}(\boldsymbol{x})) \prod_{\ell=1}^{d} \Psi'(x_\ell).$$

Example:

$$C^1 : \Psi(x) = x^3(10 - 15x + 6x^2), \Psi'(x) = 30x^2(1-x)^2,$$

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

when it holds $\Psi \in C^{r+1}[0,1]$, $\lim_{x \downarrow 0} x^{-r-1}\Psi'(x) = \lim_{x \uparrow 1}(1-x)^{-r-1}\Psi'(x) = 0$, and $g \in C^{(r,\ldots,r)}[0,1]^d$, for $r \in \mathbb{N}_0$.

Outline

1 Introduction
2 Bayesian Cubature
3 Faster
4 Lattice
5 Sobol'
6 Demonstrate
7 Conclusion

Sobol' Nets and Walsh Kernels

Definition ((t, m, d) – net)

Let $\mathcal{A}$ be the set of all elementary intervals $\mathcal{A} \subset [0, 1)^d$ where $\mathcal{A} = \prod_{\ell=1}^{d} [\alpha_\ell b^{-\gamma_\ell}, (\alpha_\ell + 1) b^{-\gamma_\ell})$, with $d, b, \gamma_\ell \in \mathbb{N}, b \geqslant 2$ and $b^{\gamma_\ell} > \alpha_\ell \geqslant 0$. For $m, t \in \mathbb{N}, m \geqslant t \geqslant 0$, the point set $\mathcal{P}_m \in [0, 1)^d$ with $n = b^m$ points is a (t, m, d) – net in base b if every $\mathcal{A}$ with volume b^{t-m} contains b^t points of $\mathcal{P}_m$.

Digital (t, m, d)-nets are a special case of (t, m, d)-nets, constructed using matrix-vector multiplications over finite fields.

Digital Sequence

Digital sequences are infinite length digital nets, i.e., the first $n = b^m$ points of a digital sequence comprise a digital net for all integer $m \in \mathbb{N}_0$.

Definition

For any non-negative integer $i = \ldots i_3 i_2 i_1 (\text{base}\, b)$, define the $\infty \times 1$ vector $\vec{\imath}$ as the vector of its digits, that is, $\vec{\imath} = (i_1, i_2, \ldots)^T$. For any point $z = 0.z_1 z_2 \ldots (\text{base}\, b) \in [0, 1)$, define the $\infty \times 1$ vector of the digits of z, that is, $\vec{z} = (z_1, z_2, \ldots)^T$. Let $\mathsf{G}_1, \ldots, \mathsf{G}_d$ denote predetermined $\infty \times \infty$ generator matrices. The digital sequence in base b is $\{\boldsymbol{z}_0, \boldsymbol{z}_1, \boldsymbol{z}_2, \ldots\}$, where each $\boldsymbol{z}_i = (z_{i1}, \ldots, z_{id})^T \in [0, 1)^d$ is defined by

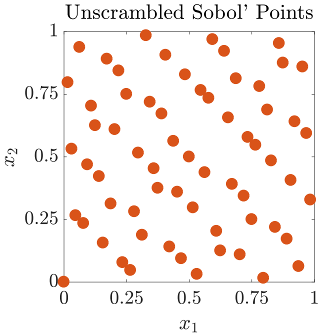
$$\vec{z}_{i\ell} = \mathsf{G}_\ell \, \vec{\imath}, \quad \ell = 1, \ldots, d, \quad i = 0, 1, \ldots .$$

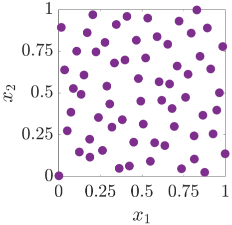
The value of t as mentioned in Definition ((t, m, d) – net) depends on the choice of G_ℓ.

Sobol' nets (Sobol', 1976) are a special case of (t, m, d)-nets when base $b = 2$.

Sobol' nets

An example of $n = 64$ Sobol' nets in $d = 2$ is given below:





Walsh kernels

Consider the covariance kernels of the form,

$$C_{\theta}(\boldsymbol{x}, \boldsymbol{t}) = K_{\theta}(\boldsymbol{x} \ominus \boldsymbol{t}) \tag{9}$$

The Walsh kernels are of the form,

$$K_{\theta}(\boldsymbol{x} \ominus \boldsymbol{t}) = \prod_{\ell=1}^{d} 1 + \eta_{\ell}\omega_{r}(x_{\ell} \ominus t_{\ell}), \quad \boldsymbol{\eta} = (\eta_1, \cdots, \eta_d), \quad \theta = (r, \boldsymbol{\eta}) \tag{10}$$

where r is the kernel order, $\boldsymbol{\eta}$ is the kernel shape parameter, and

$$\omega_r(x) = \sum_{k=1}^{\infty} \frac{\mathrm{wal}_{2,k}(x)}{2^{2r\lfloor \log_2 k \rfloor}}.$$

Explicit expression is available for ω_r in the case of order $r = 1$ (?Nuyens2013),

$$\omega_1(x) = 6\left(\frac{1}{6} - 2^{\lfloor \log_2 x \rfloor - 1}\right). \tag{11}$$

Walsh kernels

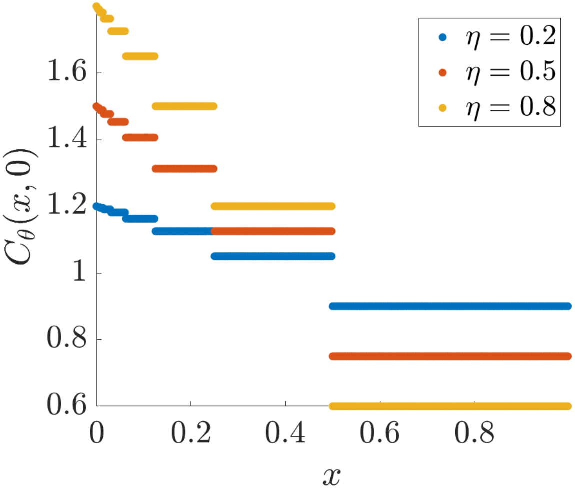


Figure: Walsh kernel of order $r = 1$ in dimension $d = 1$.

Sobol' Nets and Walsh Kernels

Theorem

Any symmetric, positive definite, digital shift-invariant covariance kernel of the form (10) *scaled to satisfy* (5), *when matched with digital net data-sites, satisfies assumptions* (4). *The fast Walsh-Hadamard transform (FWHT) can be used to expedite the estimates of* θ *in* (6) *and the credible interval widths* (7) *in* $\mathcal{O}(n \log n)$ *operations. The cubature,* $\widehat{\mu}$, *is just the sample mean.*

Walsh kernels + digital nets = '*Block-Toeplitz*' matrix

Walsh transform

The WHT involves multiplications by $2^m \times 2^m$ Walsh-Hadamard matrices, which is constructed recursively, starting with $\mathsf{H}^{(0)} = 1$,

$$\begin{aligned}
\mathsf{H}^{(1)} &= \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \\
\mathsf{H}^{(2)} &= \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}, \\
&\vdots \\
\mathsf{H}^{(m)} &= \begin{pmatrix} \mathsf{H}^{(m-1)} & \mathsf{H}^{(m-1)} \\ \mathsf{H}^{(m-1)} & -\mathsf{H}^{(m-1)} \end{pmatrix} = \underbrace{\mathsf{H}^{(1)} \bigotimes \cdots \bigotimes \mathsf{H}^{(1)}}_{m \text{ times}} = \mathsf{H}^{(1)} \bigotimes \mathsf{H}^{(m-1)} \qquad (12)
\end{aligned}$$

where $\bigotimes$ is Kronecker product.

Eigenvectors of C

The columns of Walsh-Hadamard matrix are the eigenvectors of C, i.e., V := H

Theorem

Let $(\boldsymbol{x}_i)_{i=0}^{n-1}$ be digitally shifted Sobol' nodes and K be any function, then the Gram matrix,

$$\mathsf{C}_{\theta} = \big(C(\boldsymbol{x}_i, \boldsymbol{x}_j)\big)_{i,j=0}^{n-1} = \big(K(\boldsymbol{x}_i \ominus \boldsymbol{x}_j)\big)_{i,j=0}^{n-1},$$

where $\quad n = 2^m, \quad C(\boldsymbol{x}, \boldsymbol{t}) = K(\boldsymbol{x} \ominus \boldsymbol{t}), \quad \boldsymbol{x}, \boldsymbol{t} \in [0, 1)^d,$ *is a 2×2 block-Toeplitz matrix and all the sub-blocks and their sub-sub-blocks, etc. are also 2×2 block-Toeplitz.*

Fast Bayesian transform

Theorem

The Walsh-Hadamard matrix $\mathsf{H}^{(m)}$ *factorizes* $\mathsf{C}_{\theta}^{(m)}$*, so that the columns of Walsh-Hadamard matrix are the eigenvectors of* $\mathsf{C}_{\theta}^{(m)}$*, i.e.,*

$$\mathsf{H}^{(m)}\mathsf{C}_{\theta}^{(m)} = \Lambda^{(m)}\mathsf{H}^{(m)}, \quad m \in \mathbb{N},$$

where (m) *denotes the size of the matrix is* $2^m \times 2^m$*.*

By using these two theorems

$$\mathsf{C}^{(m)} = \frac{1}{n}\mathsf{H}^{(m)}\Lambda^{(m)}\mathsf{H}^{(m)}, \quad \text{where} \quad \mathsf{H}^{(m)} = \underbrace{\mathsf{H}^{(1)} \bigotimes \cdots \bigotimes \mathsf{H}^{(1)}}_{m \text{ times}}. \tag{13}$$

Iterative Computation of Walsh Transform

Let $\tilde{\boldsymbol{y}} = \mathsf{H}^{(m+1)}\boldsymbol{y}$ for some arbitrary $\boldsymbol{y} \in \mathbb{R}^{2n}$, $n = 2^m$. Define,

$$\boldsymbol{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_{2n} \end{pmatrix}, \quad \boldsymbol{y}^{(1)} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}, \quad \boldsymbol{y}^{(2)} = \begin{pmatrix} y_{n+1} \\ \vdots \\ y_{2n} \end{pmatrix},$$

$$\tilde{\boldsymbol{y}}^{(1)} = \mathsf{H}^{(m)}\boldsymbol{y}^{(1)} = \begin{pmatrix} \tilde{y}_1^{(1)} \\ \tilde{y}_2^{(1)} \\ \vdots \\ \tilde{y}_n^{(1)} \end{pmatrix}, \quad \tilde{\boldsymbol{y}}^{(2)} = \mathsf{H}^{(m)}\boldsymbol{y}^{(2)} = \begin{pmatrix} \tilde{y}_1^{(2)} \\ \tilde{y}_2^{(2)} \\ \vdots \\ \tilde{y}_n^{(2)} \end{pmatrix}.$$

Then,

$$\tilde{\boldsymbol{y}} = \mathsf{H}^{(m+1)}\boldsymbol{y} = \begin{pmatrix} \mathsf{H}^{(m)} & \mathsf{H}^{(m)} \\ \mathsf{H}^{(m)} & -\mathsf{H}^{(m)} \end{pmatrix} \begin{pmatrix} \boldsymbol{y}^{(1)} \\ \boldsymbol{y}^{(2)} \end{pmatrix}, \qquad \text{by (12)}$$

$$= \left(\mathsf{H}^{(m)}\boldsymbol{y}^{(1)} + \mathsf{H}^{(m)}\boldsymbol{y}^{(2)}\mathsf{H}^{(m)}\boldsymbol{y}^{(1)} - \mathsf{H}^{(m)}\boldsymbol{y}^{(2)}\right) = \begin{pmatrix} \tilde{\boldsymbol{y}}^{(1)} + \tilde{\boldsymbol{y}}^{(2)} \\ \tilde{\boldsymbol{y}}^{(1)} - \tilde{\boldsymbol{y}}^{(2)} \end{pmatrix} =: \tilde{\boldsymbol{y}}$$

Outline

1 Introduction

2 Bayesian Cubature

3 Faster

4 Lattice

5 Sobol'

6 Demonstrate

7 Conclusion

Shape parameter search using gradient descent

Steepest descent search is defined as:

$$\eta_\ell^{(j+1)} = \eta_\ell^{(j)} - \nu \frac{\partial}{\partial \eta_\ell} \mathcal{L}_{\mathsf{x}}(\theta|\boldsymbol{y}), \quad j = 0, 1, \cdots, \quad \ell = 1, \cdots, d$$

$$\mathsf{x} \in \{\mathsf{EB}, \mathsf{GCV}\}$$

where ν is the step size for the gradient descent, j is the iteration index, and $\dfrac{\partial}{\partial \eta_\ell} \mathcal{L}(\theta|\boldsymbol{y})$ is either (14) or (15) depending on the choice of the hyperparameter search method. The parameter η_ℓ is usually searched in the whole $\mathbb{R}$ by using the simple domain transformation.

Computing the derivative of $\mathcal{L}_{\text{EB}}(\theta|\boldsymbol{y})$

Taking derivative with respect to θ_ℓ, for $\ell = 1, \cdots, d$

$$\mathcal{L}_{\text{EB}}(\theta|\boldsymbol{y}) = \log\left((\boldsymbol{y} - m_{\text{EB}}\mathbf{1})^T \mathsf{C}^{-1}(\boldsymbol{y} - m_{\text{EB}}\mathbf{1})\right) + \frac{1}{n}\log(\det(\mathsf{C}_\theta)),$$

$$\frac{\partial}{\partial\theta_\ell}\mathcal{L}_{\text{EB}}(\theta|\boldsymbol{y}) = -\frac{\left((\boldsymbol{y} - m_{\text{EB}}\mathbf{1})^T \mathsf{C}^{-1}\right)^T \left(\frac{\partial \mathsf{C}}{\partial\theta_\ell}\right)\left((\boldsymbol{y} - m_{\text{EB}}\mathbf{1})^T \mathsf{C}^{-1}\right)}{(\boldsymbol{y} - m_{\text{EB}}\mathbf{1})^T \mathsf{C}^{-1}(\boldsymbol{y} - m_{\text{EB}}\mathbf{1})}$$
$$+ \frac{1}{n}\operatorname{trace}\left(\mathsf{C}^{-1}\frac{\partial \mathsf{C}}{\partial\theta_\ell}\right), \quad \text{where} \quad m_{\text{EB}} = \frac{\mathbf{1}^T\mathsf{C}_\theta^{-1}\boldsymbol{y}}{\mathbf{1}^T\mathsf{C}_\theta^{-1}\mathbf{1}},$$

where we used some of the results from (?Dong2017a). After using the fast Bayesian transform properties

$$\frac{\partial}{\partial\theta_\ell}\mathcal{L}_{\text{EB}}(\theta|\boldsymbol{y}) = \frac{1}{n}\sum_{i=1}^{n}\frac{\bar{\lambda}_{i(\ell)}}{\lambda_i} - \left(\sum_{i=2}^{n}\frac{\left|\tilde{\boldsymbol{y}}_i\right|^2\bar{\lambda}_{i(\ell)}}{\lambda_i^2}\right)\left(\sum_{i=2}^{n}\frac{\left|\tilde{\boldsymbol{y}}_i\right|^2}{\lambda_\ell}\right)^{-1} \tag{14}$$

Computing the derivative of $\mathcal{L}_{\text{GCV}}(\theta|\boldsymbol{y})$

Similarly for the generalized cross-validation

$$\mathcal{L}_{\text{GCV}}(\theta|\boldsymbol{y}) = \log\left(\boldsymbol{y}^T\left[\mathsf{C}_\theta^{-2} - \frac{\mathsf{C}_\theta^{-2}\mathbf{1}\mathbf{1}^T\mathsf{C}_\theta^{-2}}{\mathbf{1}^T\mathsf{C}_\theta^{-2}\mathbf{1}}\right]\boldsymbol{y}\right) - \log\left(\text{trace}(\mathsf{C}_\theta^{-2})\right),$$

$$\text{where} \quad m_{\text{GCV}} = \frac{\mathbf{1}^T\mathsf{C}_\theta^{-2}\boldsymbol{y}}{\mathbf{1}^T\mathsf{C}_\theta^{-2}\mathbf{1}},$$

After using the fast Bayesian transform properties

$$\frac{\partial}{\partial\theta_\ell}\mathcal{L}_{\text{GCV}}(\theta|\boldsymbol{y}) = -2\left(\sum_{i=2}^{n}\frac{|\widetilde{y}_i|^2}{\lambda_i^2}\right)^{-1}\left(\sum_{i=2}^{n}\frac{|\widetilde{y}_i|^2\,\bar{\lambda}_{i(\ell)}}{\lambda_i^3}\right) + 2\left(\sum_{i=1}^{n}\frac{1}{\lambda_i}\right)^{-1}\left(\sum_{i=1}^{n}\frac{\bar{\lambda}_{i(\ell)}}{\lambda_i^2}\right), \tag{15}$$

where $\bar{\lambda}_{i(\ell)}$ is the derivative of the ith eigenvalue of the Gram matrix, C, in the ℓth variable.

Product Kernels

Product kernels in d dimensions are of the form,

$$C_\theta(\boldsymbol{t}, \boldsymbol{x}) = \prod_{\ell=1}^{d} \left[1 - \eta_\ell \; \mathfrak{C}(x_\ell, t_\ell) \right] \tag{16}$$

where η_ℓ is called shape parameter.

Derivative of the product kernel when $\eta_1 = \cdots = \eta_d = \eta$

$$\frac{\partial}{\partial \eta} C_\theta(\boldsymbol{t}, \boldsymbol{x}) = (d/\eta) C_\theta(\boldsymbol{t}, \boldsymbol{x}) \left(1 - \frac{1}{d} \sum_{\ell=1}^{d} \frac{1}{1 - \eta \mathfrak{C}(x_\ell, t_\ell)} \right).$$

When η_ℓ is different for each $\ell = 1, \cdots, d$

$$\frac{\partial}{\partial \eta_\ell} C_\theta(\boldsymbol{t}, \boldsymbol{x}) = \frac{1}{\eta_\ell} C_\theta(\boldsymbol{t}, \boldsymbol{x}) \left(1 - \frac{1}{1 - \eta_\ell \mathfrak{C}(x_\ell, t_\ell)} \right).$$

To compute $\bar{\lambda}_{i(\ell)}$

If V does not depend on θ then one can fast compute the derivative of Gram matrix C,

$$\frac{\partial \mathsf{C}}{\partial \theta_\ell} = \frac{1}{n}\mathsf{V}\frac{\partial \Lambda}{\partial \theta_\ell}\mathsf{V}^H = \frac{1}{n}\mathsf{V}\bar{\Lambda}_{(\ell)}\mathsf{V}^H, \quad \text{using} \quad \mathsf{C} = \frac{1}{n}\mathsf{V}\Lambda\mathsf{V}^H$$

$$\text{where} \quad \bar{\Lambda}_{(\ell)} = \mathrm{diag}(\bar{\boldsymbol{\lambda}}_{(\ell)}), \quad \text{and}$$

$$\bar{\boldsymbol{\lambda}}_{(\ell)} = \frac{\partial \boldsymbol{\lambda}}{\partial \theta_\ell} = \left(\frac{\partial \lambda_i}{\partial \theta_\ell}\right)_{i=1}^n = \left(\frac{\partial}{\partial \theta_\ell}\mathsf{V}^H \boldsymbol{C}_1\right) = \mathsf{V}^H\left(\frac{\partial}{\partial \theta_\ell} C_{\boldsymbol{\theta}}(\boldsymbol{x}_1, \boldsymbol{x}_i)\right)_{i=1}^n, \tag{17}$$

where we used the fast Bayesian transform property $\boldsymbol{\lambda} = \mathsf{V}^H C_1$.

Cancellation error in err_{CI}

$$\text{err}_{\text{EB}} = 2.58\sqrt{\left(1 - \frac{n}{\lambda_1}\right)\frac{1}{n^2}\sum_{i=2}^{n}\frac{|\widetilde{y}_i|^2}{\lambda_i}},$$ term $1 - \frac{n}{\lambda_1}$ causes cancellation error

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

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

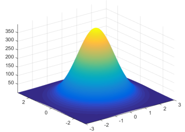
$$\begin{aligned}
\mathring{C}_\theta^{(1)}(\boldsymbol{t},\boldsymbol{x}) &= \mathring{C}_{\theta,1}(t_1,x_1),\\
\mathring{C}_\theta^{(\ell)}(\boldsymbol{t},\boldsymbol{x}) &= \mathring{C}_\theta^{(\ell-1)}[1 + \mathring{C}_{\theta,\ell}(t_\ell,x_\ell)] + \mathring{C}_{\theta,\ell}(t_\ell,x_\ell), \qquad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

Eigenvalues of $\mathring{\mathsf{C}}_\theta$: $(\mathring{\lambda}_i)_{i=1}^n = \mathsf{V}^T\mathring{\boldsymbol{C}}_1, \quad \mathring{\lambda}_1 = \lambda_1 - n, \lambda_2, \ldots, \lambda_n$

$$\text{err}_{\text{EB}} = \frac{2.58}{n}\sqrt{\frac{\mathring{\lambda}_1}{\lambda_1}\sum_{i=2}^{n}\frac{|\widetilde{y}_i|^2}{\lambda_i}}, \qquad \theta_{\text{EB}} = \underset{\theta}{\text{argmin}}\left[\log\left(\sum_{i=2}^{n}\frac{|\widetilde{y}_i|^2}{\lambda_i}\right) + \frac{1}{n}\sum_{i=1}^{n}\log(\lambda_i)\right]$$

Example Integrands

Gaussian probability $= \displaystyle\int_{[\boldsymbol{a},\boldsymbol{b}]} \frac{\mathrm{e}^{-\boldsymbol{x}^T\Sigma^{-1}\boldsymbol{x}/2}}{(2\pi)^{d/2}\,|\Sigma|^{1/2}}\,\mathrm{d}\boldsymbol{x},$ (Genz, 1993)



$$\text{Option pricing} = \int_{\mathbb{R}^d} \text{payoff}(\boldsymbol{x}) \underbrace{\frac{\mathrm{e}^{-\boldsymbol{x}^T\Sigma^{-1}\boldsymbol{x}/2}}{(2\pi)^{d/2}\,|\Sigma|^{1/2}}}_{\text{PDF of Brownian motion at } d \text{ times}} \mathrm{d}\boldsymbol{x}, \quad \text{(Glasserman, 2004)}$$

$$\text{where} \quad \text{payoff}(\boldsymbol{x}) = \mathrm{e}^{-rT} \max\left(\frac{1}{d}\sum_{k=1}^{d} S_k(x_k) - K, 0\right)$$

$$S_j(x_j) = S_0\mathrm{e}^{(r-\sigma^2/2)t_j+\sigma x_j} = \text{stock price at time } t_j = jT/d;$$

$$\text{Keister integral} = \int_{\mathbb{R}^d} \cos(\|\boldsymbol{x}\|)\exp(-\|\boldsymbol{x}\|^2)\,\mathrm{d}\boldsymbol{x}, \quad d = 1, 2, \ldots \quad \text{(Keister, 1996)}$$

Multivariate normal probability: Lattice

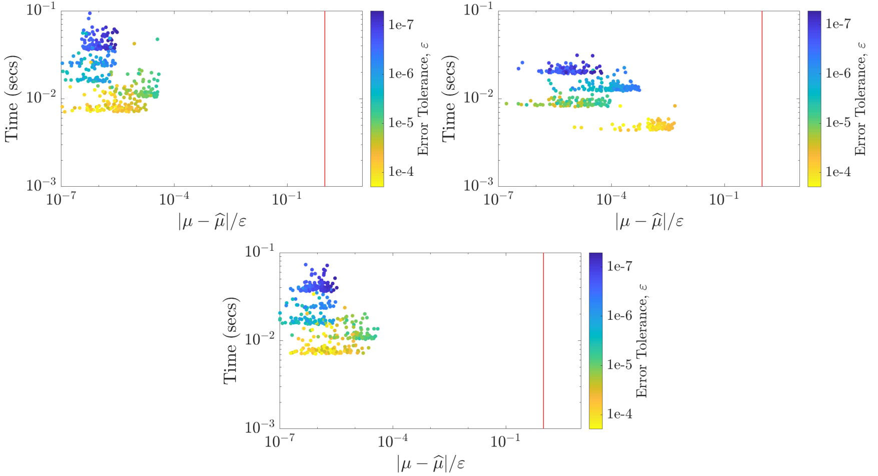


Figure: Multivariate normal probability example using 1) Empirical Bayes, 2) GCV, 3) Full Bayes stopping criterion

Keister Integral: Lattice

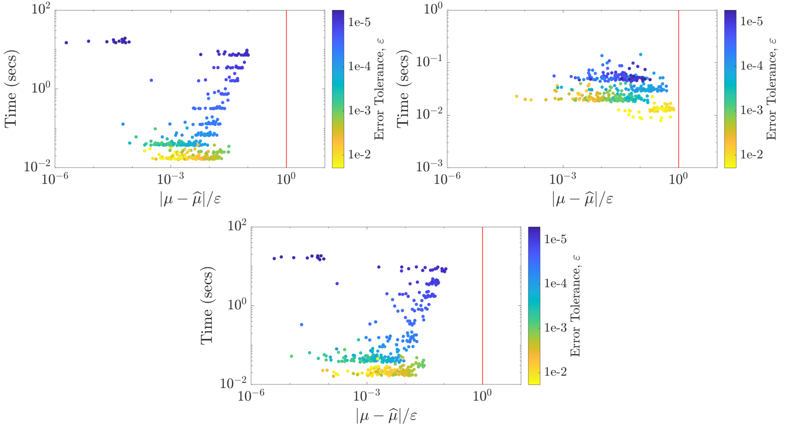


Figure: Integrating Keister function using 1) Empirical Bayes, 2) GCV, 3) Full Bayes stopping criterion

Option pricing: Lattice

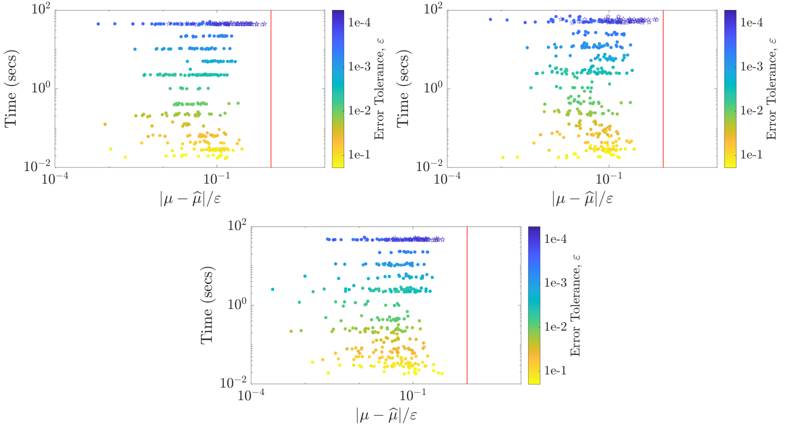


Figure: Option pricing using 1) Empirical Bayes, 2) GCV, 3) Full Bayes stopping criterion

Outline

1 Introduction

2 Bayesian Cubature

3 Faster

4 Lattice

5 Sobol'

6 Demonstrate

7 Conclusion

Summary

- Developed a *technique* for a Fast Bayesian transform
- Developed a fast automatic Bayesian cubature with $\mathcal{O}(n \log n)$ complexity
- Having the advantages of a kernel method and the low computation cost of Quasi Monte carlo
- Scalable based on the complexity of the Integrand
  i.e, Kernel order and Lattice-points can be chosen to suit the smoothness of the integrand
- Conditioning problem if the kernel C is very smooth
- Source code : `https://github.com/GailGithub/GAIL_Dev/tree/feature/BayesianCubature`.
- This work is to be published as a paper "Fast Automatic Bayesian Cubature Using Lattice Sampling by R. Jagadeeswaran, Fred J. Hickernell" (`https://arxiv.org/abs/1809.09803`)

Future work

- Choosing the kernel order r and periodization transform automatically
- Use gradient descent to find optimal θ
- Diagnostics for Gaussian process assumption
- Broaden the choice of numerical examples
- Better handling of conditioning problem and numerical errors
- Sobol pointset and Fast Walsh Transform with smooth kernels (More details next) ...

Future work : Higher order nets and fast Walsh transform

- Higher order nets and Walsh kernels could be used to achieve higher order or accuracy.

Future work : More applications

- Control variates : We would like to approximate a function of the form $(f - \beta_1 g_1 -, ..., -\beta_p g_p)$, then

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

- Function approximation : consider approximating a function of the form

$$\int_{[0,1]^d} \underbrace{f(\boldsymbol{\phi}(\boldsymbol{t})).\left|\frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{t}}\right|}_{g(\boldsymbol{t})} \mathrm{d}\boldsymbol{t}, \quad \text{where } \left|\frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{t}}\right| \text{ is Jacobian, then}$$

$$g(\boldsymbol{\psi}(\boldsymbol{x})) = f(\underbrace{\boldsymbol{\phi}(\boldsymbol{\psi}(\boldsymbol{x}))}_{\boldsymbol{x}}).\left|\frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{t}}\right|(\boldsymbol{\psi}(\boldsymbol{x})), \quad f(\boldsymbol{x}) = g(\boldsymbol{\psi}(\boldsymbol{x})).\frac{1}{\left|\frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{t}}\right|(\boldsymbol{\psi}(\boldsymbol{x}))}$$

Finally, the function approximation is

$$\begin{aligned}\tilde{f}(\boldsymbol{x}) &= \tilde{g}(\boldsymbol{\psi}(\boldsymbol{x})) \\ &= \sum w_i C(.,.)\end{aligned}$$

Thank you!

References I

Briol, F-X, C. J. Oates, M. Griolami, M. A. Osborne, and D. Sejdinovic. 2018+. *Probabilistic integration: A role in statistical computation?*, Statist. Sci. to appear.

Diaconis, P. 1988. *Bayesian numerical analysis*, Statistical decision theory and related topics iv, papers from the 4th purdue symp., west lafayette/indiana 1986, pp. 163–175.

Dick, J. and F. Pillichshammer. 2010. *Digital nets and sequences: Discrepancy theory and quasi-Monte Carlo integration*, Cambridge University Press, Cambridge.

Genz, A. 1993. *Comparison of methods for the computation of multivariate normal probabilities*, Computing Science and Statistics **25**, 400–405.

Glasserman, P. 2004. *Monte Carlo methods in financial engineering*, Applications of Mathematics, vol. 53, Springer-Verlag, New York.

Hickernell, F. J. and H. Niederreiter. 2003. *The existence of good extensible rank-1 lattices*, J. Complexity **19**, 286–300.

Keister, B. D. 1996. *Multidimensional quadrature algorithms*, Computers in Physics **10**, 119–122.

O'Hagan, A. 1991. *Bayes-Hermite quadrature*, J. Statist. Plann. Inference **29**, 245–260.

Olver, F. W. J., D. W. Lozier, R. F. Boisvert, C. W. Clark, and A. B. O. Dalhuis. 2013. *Digital library of mathematical functions*.

References II

Rasmussen, C. E. 2003. *Bayesian Monte Carlo*, Advances in Neural Information Processing Systems, pp. 489–496.

Ritter, K. 2000. *Average-case analysis of numerical problems*, Lecture Notes in Mathematics, vol. 1733, Springer-Verlag, Berlin.

Sobol', I. M. 1976. *Uniformly distributed sequences with an additional uniformity property*, Zh. Vychisl. Mat. i Mat. Fiz. **16**, 1332–1337 (Russian).

Traub, J. F., G. W. Wasilkowski, and H. Woźniakowski. 1988. *Information-based complexity*, Academic Press, Boston.