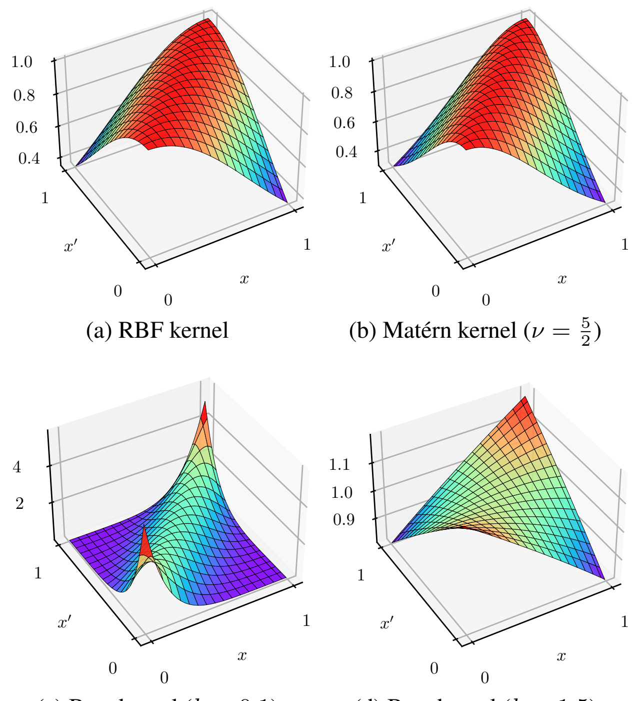
Bayesian Optimization over Bounded Domains with the Beta Product Kernel

Huy Hoang Nguyen[1] **Han Zhou**[2] **Matthew B. Blaschko**[*2] **Aleksei Tiulpin**[*1]

[1]Research Unit of Health Sciences and Technology, Finland, University of Oulu, Oulu, Finland
[2]Department ESAT, Center for Processing Speech and Images, KU Leuven, Leuven, Belgium

Abstract

Bayesian optimization with Gaussian processes (GP) is commonly used to optimize black-box functions. The Matérn and the Radial Basis Function (RBF) covariance functions are used frequently, but they do not make any assumptions about the domain of the function, which may limit their applicability in bounded domains. To address the limitation, we introduce the Beta kernel, a non-stationary kernel induced by a product of Beta distribution density functions. Such a formulation allows our kernel to naturally model functions on bounded domains. We present statistical evidence supporting the hypothesis that the kernel exhibits an exponential eigendecay rate, based on empirical analyses of its spectral properties across different settings. Our experimental results demonstrate the robustness of the Beta kernel in modeling functions with optima located near the faces or vertices of the unit hypercube. The experiments show that our kernel consistently outperforms a wide range of kernels, including the well-known Matérn and RBF, in different problems, including synthetic function optimization and the compression of vision and language models. Our implementation is available at https://github.com/imedslab/BetaKernel.



(a) RBF kernel (b) Matérn kernel ($\nu = \frac{5}{2}$)
(c) Beta kernel ($h = 0.1$) (d) Beta kernel ($h = 1.5$)

Figure 1: Covariance matrices of the Matérn kernel and our Beta kernel in the unit 1D domain. Different from the Matérn kernel, the variation along the diagonal indicates the non-stationarity of our kernel.

1 INTRODUCTION

Bayesian optimization (BO) is a well-founded approach for global optimization of noisy black-box functions, which are often expensive to evaluate. At its core, BO relies on a surrogate model to approximate the objective function and guide the search process efficiently. A Gaussian process (GP) is commonly used as this surrogate due to its flexibility, ability to quantify uncertainty, and capability to incorporate prior knowledge through covariance functions (also called kernels).

A GP is defined by its mean and covariance function. The choice of kernel is critical in encoding prior knowledge about the target function's behavior. Among many available kernels [Oh et al., 2018, Wilson and Adams, 2013, Duvenaud et al., 2011, Jebara et al., 2004], the Matérn and Radial Basis Function (RBF) kernels have been the most extensively studied [Stein, 2012, Williams and Rasmussen, 2006,

[*]Equal last author

Srinivas et al., 2009, Santin and Schaback, 2016, Vakili et al., 2021] and are widely adopted in practical applications [Pedregosa et al., 2011, Head et al., 2018, Gardner et al., 2018] [1] due to their flexibility and smoothness properties. The RBF kernel, with its infinitely differentiable form, is ideal for modeling smooth functions, while the Matérn kernel provides adjustable smoothness through the ν parameter. Formally, the Matérn kernel is defined as [Rasmussen, 2003, Myers, 1994]

$$K_{\text{Matérn}}(r) = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\sqrt{2\nu}\frac{r}{\ell}\right)^{\nu} K_{\nu}\left(\sqrt{2\nu}\frac{r}{\ell}\right), \tag{1}$$

where $r = \|\mathbf{x} - \mathbf{x}'\|_2$, $\nu > 0$ is a smoothness parameter, ℓ is a positive length scale, $\Gamma(\cdot)$ is the Gamma function, and K_ν is a modified Bessel function [Abramowitz and Stegun, 1968]. When $\nu \to \infty$, the Matérn kernel is equivalent to the RBF kernel, formulated as

$$K_{\text{RBF}}(r) = \exp\left(-\frac{r^2}{2\ell^2}\right). \tag{2}$$

RBF and Matérn are defined on unbounded domains. However, in most practical applications, the objective function is specified on bounded ones. The unawareness of the boundary of these kernels may result in either over-exploration (see Figure 3c) or neglecting boundary regions (see Figure 3d).

Both the kernels, together with the Newton-Girard Additive (NGA) [Duvenaud et al., 2011] and Spectral Mixture (SM) [Wilson and Adams, 2013] kernels, are isotropic and stationary since they can be expressed as $K(\mathbf{x}, \mathbf{x}') = K(\|\mathbf{x} - \mathbf{x}'\|), \forall \mathbf{x}, \mathbf{x}'$. However, it has been noted by Oh et al. [2018] that such stationary kernels face challenges with the boundary issue introduced by Swersky [2017], primarily due to their lack of awareness regarding absolute locations. To address this limitation, they proposed BOCK with a cylindrical (CYL) kernel defined in a hypersphere. Nevertheless, besides the CYL kernel, the high performance of BOCK is also attributed to the input wrapping proposed by Snoek et al. [2014], which is based on the cumulative density function (CDF) of the Beta distribution.

In this study, we propose a non-stationary Beta distribution-based kernel, thus the name **Beta kernel**, specifically defined on unit hypercubes. Our kernel is induced from products of multiple Beta distribution density functions, each of which naturally represents a wide range of functions defined on $[0, 1]$. We empirically show that the eigenvalue decay rate of our kernel is exponential, which is similar to RBF's.

We argue that the GP benchmarks are typically performed on synthetic test functions with optima located near the center of the search space [Laguna and Marti, 2005, Molga and Smutnicki, 2005, Back, 1996, Dixon, 1978, Hvarfner et al., 2024]. In a high-dimensional unit hypercube, while the boundary volume becomes significantly larger than the central volume, the volumes near the vertices are significantly small (see the example in Sec. 3.2). Consequently, while locating optima near the boundary is more challenging than in the central region, it is unlikely to sample data points near the vertices. To evaluate this, we modify the test function domains so that the optima are positioned near a face or vertex of the unit hypercube. Our results demonstrate that the proposed kernel is more robust than a wide range of baselines – including RBF and Matérn – under these boundary-focused settings across different test functions.

In addition, we conduct experiments on model compression tasks for deep vision and language models, including Vision Transformer (ViT), Bidirectional encoder representations from transformer (BERT), Generative Pre-trained Transformer 2 (GPT-2), and Decoding-enhanced BERT with disentangled attention (DeBERTa). The results show that our kernel substantially outperforms the baselines across the compression tasks. Furthermore, our experiments also demonstrate that the Beta kernel consistently surpasses the Matérn kernel – the most competitive baseline – when combined with various acquisition functions.

2 BETA KERNEL

2.1 PRELIMINARIES

Gaussian Process. A GP is defined as $f \sim \text{GP}(\mu(\mathbf{x}), K(\mathbf{x}, \mathbf{x}'))$, with its mean function $\mu(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$, and covariance (kernel) function $K(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - \mu(\mathbf{x}))(f(\mathbf{x}') - \mu(\mathbf{x}'))]$. Consider a setting in which we are given a set of observations $\mathbf{X}_t = (\mathbf{x}_1, \ldots, \mathbf{x}_t)$ and corresponding noisy outputs $\mathbf{y}_t = (y_1, \ldots, y_t)^\top$, where $y_t = f(\mathbf{x}_t) + \varepsilon_t$, and $\varepsilon_t \sim \mathcal{N}(0, \sigma^2)$ are i.i.d. Gaussian noise. Then, the posterior over f conditioned on the observations is also a GP whose mean $\mu_t(\mathbf{x})$, covariance $K_t(\mathbf{x}, \mathbf{x}')$, and variance $\sigma_t^2(\mathbf{x})$ are formulated as

$$\begin{aligned}\mu_t(\mathbf{x}) &= \mathbf{K}_t(\mathbf{x})^\top(\mathbf{K}_t^* + \sigma^2\mathbb{I})^{-1}\mathbf{y}_t,\\ K_t(\mathbf{x}, \mathbf{x}') &= K(\mathbf{x}, \mathbf{x}') - \mathbf{K}_t(\mathbf{x})^\top(\mathbf{K}_t^* + \sigma^2\mathbb{I})^{-1}\mathbf{K}_t(\mathbf{x}'),\\ \sigma_t^2(\mathbf{x}) &= K_t(\mathbf{x}, \mathbf{x}),\end{aligned} \tag{3}$$

where $\mathbf{K}_t(\mathbf{x}) = [K(\mathbf{x}_1, \mathbf{x}), \ldots, K(\mathbf{x}_t, \mathbf{x})]^\top$, and $\mathbf{K}_t^* = [K(\mathbf{x}_i, \mathbf{x}_j)]_{i,j=1}^t$ is the positive definite covariance matrix.

Beta Distribution. The probability density function of the Beta distribution $\text{Beta}(\alpha, \beta)$ is defined on $[0, 1]$ as

$$\frac{1}{B(\alpha, \beta)} x^{\alpha-1}(1-x)^{\beta-1}, \tag{4}$$

[1]The Matérn kernel is the default choice in various GP libraries such as Scikit-optimize, GPyTorch, and GPyOpt.

where $x \in [0,1]$, $\alpha, \beta > 0$ and

$$B(\alpha,\beta) = \int_0^1 s^{\alpha-1}(1-s)^{\beta-1} ds = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}, \quad (5)$$

is the beta function with $\Gamma(\cdot)$ denoting the gamma function. In this work, we are interested in the case where $\alpha, \beta > 1$, which allows the mode to be well-defined as $\frac{\alpha-1}{\alpha+\beta-2}$.

Probability product kernels. Let p and p' denote probability distributions on a space $\mathcal{S}$, and let ρ be a positive constant. Assuming that $p^\rho, p'^\rho \in L^2(\mathcal{S})$, the probability product kernel between p and p' is defined by Jebara et al. [2004] as

$$K^{\text{prob}}(p,p') = \int_{\mathcal{S}} p(s)^\rho p'(s)^\rho ds. \quad (6)$$

When $\rho = 1$, the kernel simplifies to the expectation of one distribution with respect to the other. In this case, it becomes equivalent to a kernel defined over two corresponding samples $\mathbf{x}$ and $\mathbf{x}'$ drawn from the distributions p and p', respectively:

$$K(\mathbf{x},\mathbf{x}') = \int p(s)p'(s)ds = \mathbb{E}_p[p'(s)] = \mathbb{E}_{p'}[p(s)]. \quad (7)$$

2.2 KERNEL DERIVATION

Let us first consider the one-dimensional case. Specifically, $\forall x, x' \in [0,1]$, the Beta product kernel (or simply, *Beta kernel*), is constructed as a probability product kernel [Jebara et al., 2004] with respect to the Beta distribution as follows

$$\begin{aligned} K_\beta(x,x') &= K_\beta((\alpha,\beta),(\alpha',\beta')) \\ &= C\int_0^1 s^{\alpha-1}(1-s)^{\beta-1}s^{\alpha'-1}(1-s)^{\beta'-1}\mathrm{d}s \\ &= C\int_0^1 s^{\alpha+\alpha'-2}(1-s)^{\beta+\beta'-2}\mathrm{d}s, \end{aligned}$$

where x and x' represent the modes of the two Beta distributions, respectively, and the normalization term $C = \frac{1}{\mathrm{B}(\alpha,\beta)} \cdot \frac{1}{\mathrm{B}(\alpha',\beta')}$. The shape parameters α and β are connected to the modes via a common smoothing parameter h, such that $\alpha = 1 + \frac{x}{h}$ and $\beta = 1 + \frac{1-x}{h}$. To ensure the unique existence of the modes x and x', we require that $\alpha, \beta > 1$, which implies $h > 0$.

Based on the definition of the Beta function in Eq. (5) and given that $\alpha + \beta = \alpha' + \beta' = 2 + \frac{1}{h}$, it follows that

$$\begin{aligned} K_\beta(x,x') &= \frac{\mathrm{B}(\alpha+\alpha'-1, \beta+\beta'-1)}{\mathrm{B}(\alpha,\beta)\mathrm{B}(\alpha',\beta')} \\ &= \frac{\Gamma(\alpha+\beta)\Gamma(\alpha'+\beta')\Gamma(\alpha+\alpha'-1)\Gamma(\beta+\beta'-1)}{\Gamma(\alpha)\Gamma(\beta)\Gamma(\alpha')\Gamma(\beta')\Gamma(\alpha+\alpha'+\beta+\beta'-2)} \\ &= \frac{\Gamma^2(\frac{1}{h}+2)}{\Gamma(\frac{2}{h}+2)}\frac{\Gamma(\alpha+\alpha'-1)\Gamma(\beta+\beta'-1)}{\Gamma(\alpha)\Gamma(\beta)\Gamma(\alpha')\Gamma(\beta')}. \end{aligned}$$

By assuming that all dimensions are independent of each other, the definition of the Beta kernel in the d-dimensional space can be easily extended as the product over d dimensions. Specifically, $\forall \mathbf{x}, \mathbf{x}' \in [0,1]^d$, we define the Beta kernel as

$$K_\beta(\mathbf{x},\mathbf{x}') = \tilde{C}\prod_{i=1}^{d}\frac{\Gamma(\alpha_i+\alpha_i'-1)\Gamma(\beta_i+\beta_i'-1)}{\Gamma(\alpha_i)\Gamma(\beta_i)\Gamma(\alpha_i')\Gamma(\beta_i')}, \quad (8)$$

where

$$\tilde{C} = \prod_{i=1}^{d}\frac{\Gamma^2\left(\frac{1}{h_i}+2\right)}{\Gamma\left(\frac{2}{h_i}+2\right)}. \quad (9)$$

2.3 PROPERTIES

Validity of Beta product kernel To ensure that the Beta product kernel is a valid kernel function, it must be positive semidefinite. This property guarantees that the resulting covariance matrix is symmetric and all its eigenvalues are non-negative, which is essential for its use in kernel-based methods such as support vector machines and Gaussian processes. Fortunately, this property follows from a more general result concerning probability product kernels. In particular, Theorem 1 guarantees the validity of any kernel constructed as a probability product kernel, including the Beta product kernel.

Theorem 1. $K(\mathbf{x},\mathbf{x}') = \int_{\mathcal{S}} p(s \mid \mathbf{x})^\rho p(s \mid \mathbf{x}')^\rho \mathrm{d}s$ *is positive semidefinite for* $\rho > 0$.

Proof. We follow the definition of a Mercer kernel and consider all possible $\mathbf{x}_1, \dots, \mathbf{x}_m$ and real numbers $c_1, \dots, c_m$. To show that $K(\mathbf{x},\mathbf{x}')$ is a valid kernel, we need to prove that $\sum_{i=1}^m\sum_{j=1}^m c_ic_jK(\mathbf{x}_i,\mathbf{x}_j) \geq 0$. We can see that

$$\sum_{i=1}^{m}\sum_{j=1}^{m} c_ic_jK(\mathbf{x}_i,\mathbf{x}_j) \quad (10)$$

$$= \sum_{i=1}^{m}\sum_{j=1}^{m} c_ic_j\int_{\mathcal{S}} p(s \mid \mathbf{x}_i)^\rho p(s \mid \mathbf{x}_j)^\rho \mathrm{d}s \quad (11)$$

$$= \int_{\mathcal{S}}\left\{\sum_{i=1}^{m}\sum_{j=1}^{m} c_ic_jp(s \mid \mathbf{x}_i)^\rho p(s \mid \mathbf{x}_j)^\rho\right\} ds. \quad (12)$$

Now, we focus on the inner sums under the integral, and express that

$$\sum_{i=1}^{m}\sum_{j=1}^{m} c_ic_jp(s \mid \mathbf{x}_i)^\rho p(s \mid \mathbf{x}_j)^\rho \quad (13)$$

$$= \sum_{i=1}^{m}[c_ip(s \mid \mathbf{x}_i)^\rho]^2 + 2\sum_{i\neq j}^{m}[c_ip(s \mid \mathbf{x}_i)^\rho][c_jp(s \mid \mathbf{x}_j)^\rho] \quad (14)$$

$$= \left[\sum_{i=1}^{m} c_ip(s \mid \mathbf{x}_i)^\rho\right]^2 \geq 0. \quad (15)$$

The last equation (14)–(15) holds due to the identity $\left(\sum_{i=1}^m a_i\right)^2 = \sum_{i=1}^m a_i^2 + 2\sum_{i\neq j}^m a_i a_j$, with $a_i = c_i p(s \mid \mathbf{x}_i)^\rho$. Since the integrand is non-negative for all s, the entire integral is non-negative, which completes the proof. □

Non-stationarity. Due to the formulation in (8), the proposed kernel $K_\beta(\mathbf{x}, \mathbf{x}')$ cannot be expressed solely as a function of $\mathbf{x} - \mathbf{x}'$, which indicates its non-stationarity. To illustrate this, we compare the Beta kernel to the Matérn kernel on the unit 1D domain in Figure 1. Whereas RBF and Matérn have constant diagonals, our Beta kernel's diagonal varies depending on h.

Diagonal. The diagonal of the covariance matrix w.r.t. the Beta kernel is characterized as

$$K_\beta(\mathbf{x}, \mathbf{x}) = \tilde{C}\prod_{i=1}^{d} \frac{\Gamma(2\frac{x_i}{h_i}+1)\Gamma(2\frac{1-x_i}{h_i}+1)}{\Gamma^2(\frac{x_i}{h_i}+1)\Gamma^2(\frac{1-x_i}{h_i}+1)}. \tag{16}$$

We derive an upper bound of $K_\beta(\mathbf{x}, \mathbf{x})$ in Proposition 2.

Proposition 2. *Assume that $h_i = h\ \forall i \in [d]$, $\forall \mathbf{x} \in [0,1]^d$, we can bound $K_\beta(\mathbf{x}, \mathbf{x})$ as follows*

$$K_\beta(\mathbf{x}, \mathbf{x}) \leq 2^{3d-\frac{2d}{h}}\left(\frac{1}{h}+1\right)^d\left(\frac{1}{h\pi}+\frac{3}{2\pi}\right)^{\frac{d}{2}}. \tag{17}$$

Proof. The full proof is provided in Appendix A. □

2.4 NUMERICAL ANALYSES OF EIGENVALUE DECAY

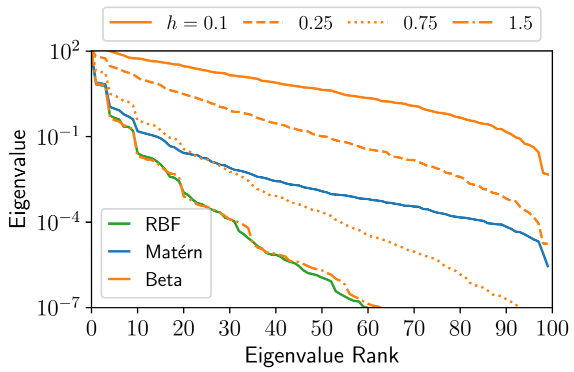


Figure 2: Spectral decay for the RBF ($\ell = 1$), Matérn ($\nu = 2.5$), and Beta kernels on 3D unit hypercube.

In this section, we present numerical analyses to examine the eigenvalue decay rate of the Beta kernel, which is associated with the smoothness of the functions that the GP can model. In addition, the eigendecay rate is linked to the regret bound of the kernel [Srinivas et al., 2009, Vakili et al., 2021]. Our analysis is conducted with reference to the well-established

Table 1: P-values from the statistical analysis of the exponential eigendecay rate of the proposed kernel across different dimensions and bandwidths.

h	$d = 5$	$d = 10$	$d = 20$	$d = 50$
0.1	7.7×10^{-36}	1.4×10^{-31}	3.6×10^{-32}	1.0×10^{-32}
0.25	1.9×10^{-43}	5.6×10^{-36}	2.4×10^{-33}	5.6×10^{-34}
0.5	1.3×10^{-40}	1.1×10^{-31}	1.0×10^{-33}	9.7×10^{-35}
0.75	2.3×10^{-37}	9.7×10^{-28}	1.1×10^{-25}	5.0×10^{-35}
1	2.5×10^{-35}	2.9×10^{-26}	1.5×10^{-21}	2.4×10^{-33}
1.5	7.1×10^{-33}	5.0×10^{-25}	1.4×10^{-19}	2.3×10^{-17}

eigendecay rates of the Matérn and RBF kernels, which are $O(m^{-\frac{2\nu+d}{d}})$ and $O(\exp(-m^{1/d}))$, respectively [Santin and Schaback, 2016, Belkin, 2018].

We compute the expected spectrum decay using the following procedure:

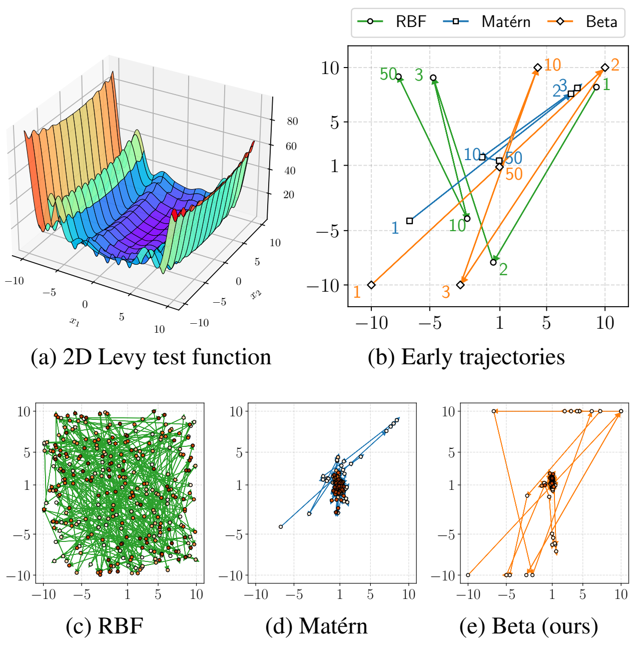
1. We generated 300 random data matrices X_i of size $100 \times d$.
2. For every X_i, we generated the corresponding kernel matrix $K_i \in \mathbb{R}^{d\times d}$ by computing $k_\ell(\cdot, \cdot)$ between all the rows of X_i.
3. We computed the sorted sets of eigenvalues for all K_i and then averaged them, resulting in the corresponding expected spectrum for the most important kernels, several hyperparameter settings, and problem dimensions.

In Figure 2, we analyze the spectral decay of the RBF, Matérn, and Beta kernels on the 3D unit hypercube. Accordingly, there is a strong correlation between the bandwidth parameter h and the eigenvalue decay rate of the Beta kernel. With $h \leq 0.25$, the Beta kernel shows a slower eigenvalue decay rate compared to the Matérn kernel with $\nu = 2.5$. Additionally, in the logarithmic scale of Figure 2, the eigendecay of the Beta kernel appears approximately linear, which suggests a potential exponential decay rate. When $h = 1.5$, its eigendecay closely matches the exponential rate of the RBF kernel.

We further conduct statistical analyses to assess whether the eigendecay of the Beta kernel follows an exponential trend. Specifically, we consider various settings with $d \in \{5, 10, 20, 50\}$ and bandwidth $h \in \{0.1, 0.25, 0.5, 0.75, 1, 1.5\}$. After computing the expected spectrum for each setting, we fit a linear regression model to examine the relationship between $\log \lambda_j$ and j, where λ_j denotes the eigenvalue and j is its index. If the eigendecay is exponential, this relationship should be statistically significant. As shown in Table 1, the p-values across various settings are significantly low, providing strong evidence in support of our hypothesis.

Table 2: Models and datasets for the compression task.

Dataset	Task Description	Model	Params	d
ImageNet	Visual object classification	ViT	87M	72
SQuAD	Question-answering	GPT-2	124M	48
SQuAD	Question-answering	BERT	109M	72
MNLI/ RTE	Natural language inference	DeBERTa	184M	14
QNLI	Sentence pair classification			
MRPC	Similarity and paraphrase			



(a) 2D Levy test function (b) Early trajectories

(c) RBF (d) Matérn (e) Beta (ours)

Figure 3: Global optimization on the 2D Levy test function, where the global minimum is located at $(1, 1)$. (b) The colored numbers represent optimization iterations corresponding to different kernel functions. (c-e) Convergence behavior over 300 iterations: (c) The RBF kernel shows over-exploration, (d) the Matérn kernel tends to neglect boundary regions, and (e) our Beta kernel achieves a more balanced trade-off between exploration and exploitation.

3 EXPERIMENTS

Our main focus of this section is to demonstrate fair and direct comparisons between the proposed Beta kernel and two widely-used stationary kernels – RBF and Matérn – in BO using GP. For that purpose, we conducted our experiments on both synthetic data and real-world vision and natural language data. Additionally, we investigated the compatibility of the mentioned kernels with a wide range of acquisition functions such as UCB, PI [Kushner, 1964], EI [Jones et al., 1998], corrected PI (PI_C) [Ma et al., 2019], and corrected EI (EI_C) [Zhou et al., 2024]. Furthermore, we intuitively illustrated behavioral differences of our kernel compared to RBF and Matérn.

3.1 EXPERIMENTAL SETUP

Optimization Tasks. For synthetic data, we performed the global optimization on the Levy test function with $d \in \{2, 4, 8\}$ [Laguna and Marti, 2005]. For vision model compression, we aimed to compress the Vision Transformer (ViT) architecture [Dosovitskiy, 2020] on the 1k-ImageNet dataset [Deng et al., 2009]. For language model compression, we performed compression on the BERT (large) model [Devlin, 2018], and the GPT-2 model [Radford et al., 2019] on the SQuAD v1 dataset [Rajpurkar et al., 2016]. In addition, we compressed on DeBERTa-v3 [He et al., 2020] on four datasets – namely MNLI [Williams et al., 2017], RTE [Dagan et al., 2005, Bar-Haim et al., 2006, Giampiccolo et al., 2007, Bentivogli et al., 2009], QNLI [Rajpurkar et al., 2016, Wang, 2018], and MRPC [Dolan and Brockett, 2005] – in the GLUE benchmark [Wang, 2018]. The detailed description is presented in Table 2.

Implementation Details. Our implementation was based on the following open-source libraries: PyTorch [Paszke et al., 2019], HuggingFace, BoTorch [Balandat et al., 2020], and GPyTorch [Gardner et al., 2018]. The training processes were run on NVIDIA V100 GPUs. Each experiment was repeated 10 times with different random seeds, then its mean and standard error were reported. The lengthscales of the RBF and Matérn kernels, along with the bandwidth of the Beta kernel, were learned through maximum marginal likelihood during training.

Baselines. Besides the two well-known kernels RBF and Matérn, we compared our kernel to Spectral Mixture (SM) [Wilson and Adams, 2013], Newton-Girard Additive (NGA) [Duvenaud et al., 2011], and Cylindrical (CYL) [Oh et al., 2018] kernels.

3.2 GLOBAL OPTIMIZATION ON SYNTHETIC TEST FUNCTIONS

Behavioral Intuition on Levy Function. We utilized the GP-UCB algorithm for the minimization. We initially selected $3 \cdot d$ data points using the Sobol's algorithm [Sobol', 1967, Owen, 1998], and performed the optimization in 300 iterations.

We graphically depict the 2-dimensional Levy test function with the global minimum at $(1, 1)$ in Figure 3. The convergence behaviors for different kernels are shown in Figure 3b. Due to the Beta kernel's characteristic as shown in Figure 1, our kernel prioritized exploring the two corner points $(-10, -10)$ and $(10, 10)$ as well as the boundaries of the domain. Detailed convergence trajectories are further illustrated in Figures 3c, 3d, and 3e. Whereas the RBF kernel exhibited a tendency for over-exploration across the entire

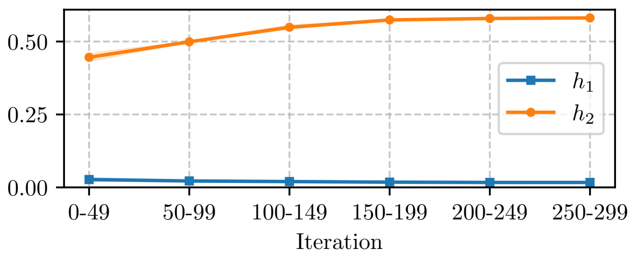
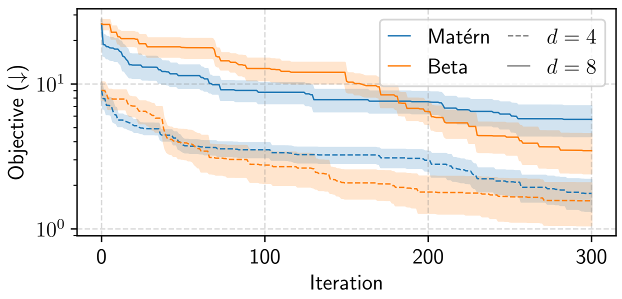
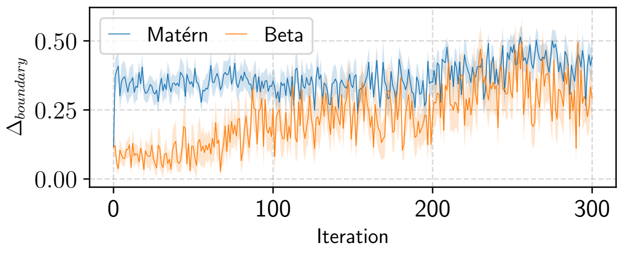


Figure 4: Convergence of the smoothing parameters h in the 2D Levy function optimization



(a) Convergence



(b) Normalized distance to boundary ($d = 8$)

Figure 5: Comparison between GP using the Matérn kernel ($\nu = 2.5$), and our Beta kernel on the Levy test function.

domain, the Matérn kernel favored the central region, albeit largely neglecting the domain boundaries. Different from the two references, our kernel with its awareness of domain boundaries maintained a reasonable ratio of exploration and exploitation. In Figure 4, we illustrate the convergence of the smoothing parameters h_1 and h_2. By maximizing the marginal likelihood, these parameters were automatically learned, converging to distinct values.

In Figure 5, we investigated the convergences of the Beta and Matérn kernels on d-dimensional Levy test function with $d \in \{4, 8\}$. To demonstrate the awareness of boundary regions, we computed the L^∞-based normalized distance to the boundary, formulated as $\Delta_{boundary} = 1 - 2\|\tilde{\mathbf{x}} - \mathbf{c}\|_\infty$, where $\tilde{\mathbf{x}} = \left(\frac{x_1 - m_1}{M_1 - m_1}, \dots, \frac{x_d - m_d}{M_d - m_d}\right)$ is a normalized version of $\mathbf{x}$, m_i and M_i represent the lower and upper bounds of the i-th dimension, and $\mathbf{c} = (\frac{1}{2}, \cdots, \frac{1}{2})$ is the center of the unit hypercube. Since the optimal solution of the Levy function lies at $(1, 1, \dots, 1)^\top$, near the domain center, Matérn demonstrated an advantage in the early iterations.

Table 3: Performance comparison across different settings ($d = 20$). Setting 1: global solutions near the center. Setting 2: first global solution near the margin. Setting 3: first global solution near a vertex. † denotes experiments performed with data warping technique from [Snoek et al., 2014].

Kernel	Griewank	Ackley	Branin	Hartmann	Levy
Setting 1: Global optimum near the center					
RBF	342.0±22.1	20.4±0.1	31.5±0.0	-0.8±0.0	141.0±15.0
SM	277.0±10.3	19.8±0.1	16.5±1.1	-1.4±0.1	79.4±8.8
NGA	216.7±59.3	20.2±0.2	18.0±1.5	-0.9±0.1	55.1±6.6
CYL	296.3±33.8	19.2±1.2	24.2±1.7	-1.2±0.0	131.0±11.8
Matérn	**22.7±1.6**	**17.4±0.3**	**5.5±0.3**	**-2.4±0.1**	10.6±1.4
Beta	122.3±11.1	20.7±0.0	7.0±0.6	-2.2±0.1	**9.2±0.3**
Setting 2: Global optimum near a face					
RBF	326.5±14.9	20.3±0.1	22.3±2.0	-1.1±0.0	104.9±8.0
SM	279.2±7.7	19.5±0.2	18.3±1.1	-1.4±0.1	81.6±5.6
NGA	216.0±36.5	19.4±0.4	24.4±3.9	-1.2±0.1	70.6±10.6
CYL	275.9±28.0	20.3±0.1	26.2±3.0	-1.1±0.1	101.8±6.7
Matérn	24.7±2.3	17.3±0.3	**4.1±0.3**	-1.6±0.0	11.2±1.7
Beta	**20.4±0.2**	**10.1±0.0**	15.0±1.1	**-2.0±0.1**	**2.2±0.1**
Setting 3: Global optimum near a vertex					
RBF	309.9±16.5	19.9±0.2	58.6±5.5	-0.8±0.1	87.2±12.0
RBF†	289.5±27.7	19.9±0.1	52.1±6.2	-0.8±0.1	83.2 ±8.4
SM	189.1±36.5	20.0±0.1	48.9±6.0	-0.9±0.0	69.4±3.6
NGA	112.3±34.7	19.6±0.3	8.5±1.7	-1.2±0.1	68.4±9.6
CYL	278.5±29.2	20.3±0.1	64.0±5.4	-	69.3±3.3
Matérn	21.1±1.8	17.0±0.3	5.9±0.9	-2.1±0.0	24.6±3.6
Matérn†	44.2±4.2	16.9±0.2	10.1±1.2	-2.1±0.1	19.5±2.3
Beta	**20.3±6.4**	**13.9±0.7**	**5.4±1.6**	**-2.2±0.1**	**6.9±0.8**

The changes of $\Delta_{boundary}$ during the training shown in Figure 5b quantitatively indicate the awareness of the boundary regions of our kernel, which is consistent with the observation in Figure 3 and Table 3. The higher dimensional the domain was, the more iterations were needed to explore the domain boundaries. After exploring the hypercube's boundaries the Beta kernel regained its performance in the later stages.

Impact of Optima Location on GP Performance. We considered the unit hypercube with $d = 20$. Let ε be a margin, we could split the unit hypercube into three partitions: (i) central volume, denoted by V_c with $|V_c| = (1 - 2\varepsilon)^d$, (ii) ε-size 2^d sub-hypercubes near vertices, represented by V_v with $|V_v| = (2\varepsilon)^d$, (iii) the remaining volumes near the faces, denoted by V_f with $|V_f| = 1 - (1 - 2\varepsilon)^d - (2\varepsilon)^d$. By fixing $\varepsilon = 0.05$, for instance, then we had $|V_c| \approx 0.1216$, $|V_v| = 10^{-20}$, and $|V_f| \approx 0.8784$. Originally, the synthetic test functions – Griewank, Ackley, repeated Branin, repeated Hartmann6, and Levy – have optimal solutions near the center of the unit hypercube (setting 1). To validate the impact of optima location on the optimization, we cropped the domains of those functions to enforce one of

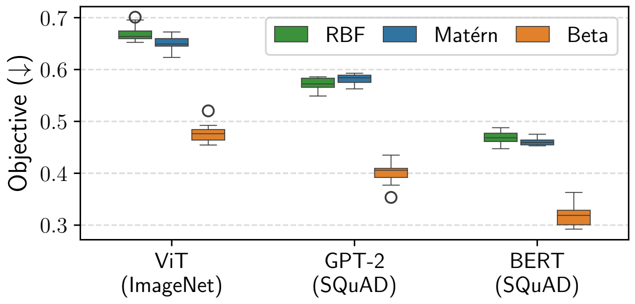


Figure 6: Model compression comparisons between GP using the RBF, Matérn ($\nu = 2.5$), and Beta kernels on the ImageNet and SQuAD datasets. The common acquisition function is UCB.

Table 4: Objective comparison of different kernels across the model compression tasks. Each value represents the combined error rate and compression rate ($w = 1$), as defined in Eq. (18).

Kernel	ViT	BERT	GPT-2	DeBERTa
	ImageNet	SQuAD	SQuAD	MNLI
RBF	$0.671_{\pm 0.005}$	$0.468_{\pm 0.005}$	$0.571_{\pm 0.004}$	$0.257_{\pm 0.006}$
SM	$0.665_{\pm 0.004}$	$0.496_{\pm 0.005}$	$0.560_{\pm 0.003}$	$0.294_{\pm 0.004}$
NGA	$0.680_{\pm 0.002}$	$0.483_{\pm 0.006}$	$0.581_{\pm 0.001}$	–
CYL	$0.684_{\pm 0.008}$	$0.515_{\pm 0.012}$	$0.570_{\pm 0.005}$	$0.304_{\pm 0.004}$
Matérn	$0.651_{\pm 0.005}$	$0.461_{\pm 0.002}$	$0.582_{\pm 0.003}$	$0.246_{\pm 0.008}$
Beta	$\mathbf{0.478}_{\pm 0.006}$	$\mathbf{0.319}_{\pm 0.007}$	$\mathbf{0.401}_{\pm 0.008}$	$\mathbf{0.124}_{\pm 0.001}$

the optima close to a face (setting 2) or a vertex (setting 3) of the unit hypercube. Specifically, we shifted $x_1^{\min}$ such that $x_1^* - x_1^{\min} = \varepsilon(x_1^{\max} - x_1^{\min})x^*$ for setting 2. For setting 3, we ensured that $x_i^* - x_i^{\min} = \varepsilon(x_i^{\max} - x_i^{\min}), \forall i \in [d]$, where x^* is the first optimum.

We present the results of the three settings in Table 3. Of all settings, the Matérn kernel was the most competitive baseline. In setting 1, Matérn achieved the highest objective values in 4 out of 5 test functions, while our kernel was the second-best in most of the cases. The proposed kernel notably showed its strength in settings 2 and 3 by outperforming all the baselines in 9 out of 10 test functions.

3.3 VISION AND LANGUAGE MODEL COMPRESSION

Compression Task Description. We utilized the LoSparse method [Li et al., 2023], which was originally designed to compress linear layers in transformer-based language models using low-rank and sparse approximation. A linear layer is expressed as $\mathbf{Y} = \mathbf{X}\mathbf{W}$ where $\mathbf{W} \in \mathbb{R}^{d_1 \times d_2}$ is learnable parameters, $\mathbf{X} \in \mathbb{R}^{1 \times d_1}$ is input and output features, and $\mathbf{Y} \in \mathbb{R}^{1 \times d_2}$ is the output. LoSparse performs both low-rank decomposition and sparse approximation on $\mathbf{W}$, formulated as $\mathbf{W} = \mathbf{U}\mathbf{V} + \mathbf{S}$ with $\mathbf{U} \in \mathbb{R}^{d_1 \times r}$, $\mathbf{V} \in \mathbb{R}^{r \times d_2}$, and $\mathbf{S} \in \mathbb{R}^{d_1 \times d_2}$. In [Li et al., 2023], the low ranks r of all linear layers were identical and independent of layer indices. In our work, we utilized BO with GP to search for the global optimal set of ranks for all linear layers. Precisely, the model compression is formulated as a multiple objective optimization as follows

$$\mathbf{x}^* = \arg\min_{\mathbf{x} \in [0,1]^d} \left[w \cdot \mathcal{R}(\mathbf{x}) + \mathcal{L}(\mathbf{x}) \right], \tag{18}$$

where $\mathbf{x}$ represents the d-dimensional vector of low ranks, $\mathcal{R}(\cdot)$ is the compression rate compared to the original model, $\mathcal{L}(\cdot)$ is the error rate of the compressed model, and w is a positive coefficient.

In practice, we constrained $x_i \in [0.05, 0.95], \forall i \in [d]$, and set $w = 1$. We initially sampled 5 data points using the Sobol's algorithm [Sobol', 1967, Owen, 1998], and performed the optimization in 30 iterations. After obtaining each compressed version of a model, we fine-tuned it for a varying number of iterations based on the evaluation cost (see Table S1).

Table 5: Performance of different kernels in terms of FLOPs, F1 score, and accuracy. FLOPs saving is calculated as (original FLOPs - compressed FLOPs) / original FLOPs.

(a) FLOPs and F1 score comparison

Kernel	FLOPs	FLOPs saving (%)	F1 (%)
Original	1.5T	0	
RBF	638G	57.5	86.06
Matern	626G	58.3	86.47
Beta	349G	76.7	86.44

(b) FLOPs and accuracy comparison

Kernel	FLOPs	FLOPs saving (%)	Accuracy (%)
Original	1.08T	0	
RBF	450G	58.4	74.26
Matern	444G	58.9	74.34
Beta	261G	75.8	74.29

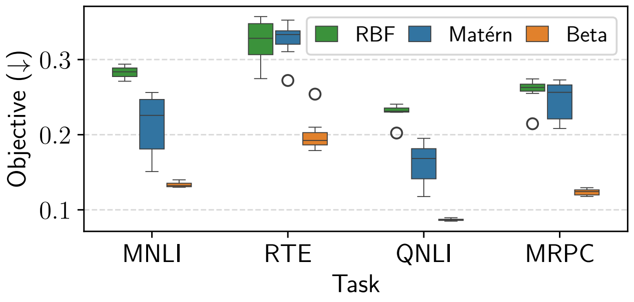


Figure 7: DeBERTa-v3 compression comparison between GP using the RBF kernel, the Matérn kernel ($\nu = 2.5$), and our Beta kernel on different tasks in the GLUE benchmark ($d = 14$). The common acquisition function is UCB.

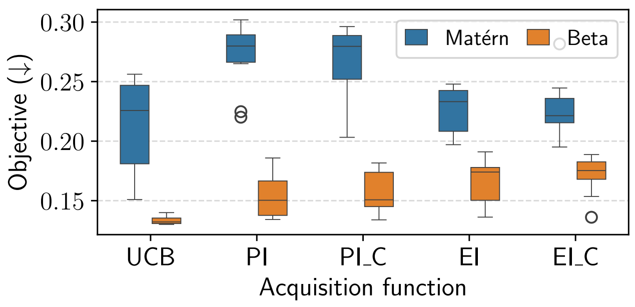


Figure 8: DeBERTa-v3 compression ($d = 14$) comparison between GP using the Matérn kernel ($\nu = 2.5$) and our Beta kernel on the MNLI task of GLUE.

ViT, GPT-2, and BERT Compression. For this set of experiments, we employed UCB as the common acquisition function. In Figure 6 and Table 4, we present the compression results on the ViT, GPT-2, and BERT models. Overall, Matérn was the most competitive baseline across the tasks (see Table 4). In Figure 6, while the difference between RBF and Matérn was insignificant, the Beta kernel enabled BO with GP to achieve substantially better compression objectives compared to the two well-known baselines across the three settings. Specifically, when compressing ViT on ImageNet, utilizing the Beta kernel yielded an objective of 0.478 ± 0.006, which was 0.193 and 0.172 better than RBF and Matérn, respectively. For the GPT-2 compression task, our Beta kernel achieved an objective of 0.401 ± 0.008, outperforming RBF and Matérn with margins of 0.170 and 0.181, respectively. For BERT compression, our method reached an objective of 0.319 ± 0.007, substantially surpassing RBF and Matérn by 0.147 and 0.142, respectively.

In Table 5, we further analyzed individual metrics in the optimization objective. In both tasks, the advantage of the Beta kernel primarily came from the compression rate improvement. Compared to Matérn, our kernel maintained insignificant trade-offs of performance for the gains of 17.3% and 14.09% in compression rate on ViT and BERT, respectively. Our compressed models were 14.79 and 43.68 times more computationally efficient than the original ViT and BERT, respectively.

DeBERTa-v3 Compression. The results of compressing DeBERTa-v3 on the four tasks from the GLUE benchmark are presented in Figure 7 and part of Table 4. On the MNLI dataset, the Matérn kernel was the strongest baseline competitor. The proposed Beta kernel achieved objective scores of 0.124 ± 0.001, 0.198 ± 0.007, 0.087 ± 0.000, and 0.133 ± 0.001 on MNLI, RTE, QNLI, and MRPC, respectively, which consistently outperformed both RBF and Matérn kernels. The objective gaps between our kernel and the Matérn kernel were 0.08, 0.127, 0.075, and 0.123 on MNLI, RTE, QNLI, and MRPC, respectively.

We further examined the combinations of the Matérn and Beta kernels against various acquisition functions in the MNLI task (see Figure 8). Overall, the Beta kernel achieved substantially better objective scores than the baseline across all five acquisition functions. While UCB and corrected EI were the most compatible acquisition functions for the Matérn kernel, our kernel demonstrated a clear advantage when paired with UCB. Specifically, the Beta kernel with UCB obtained an objective score of 0.133, outperforming its combinations with PI, corrected PI, EI, and corrected PI by 0.021, 0.023, 0.033, and 0.038, respectively. When using UCB, the Beta kernel outperformed the Matérn kernel by 0.08.

Learned Compression Strategy Analysis. In Figure 9, we graphically demonstrate the kernel density estimates (KDEs) of $\{x_i\}_{i\in[d]}$ during training. To properly address the boundary issue of KDE on these bounded domains, we employed the Beta KDE [Chen, 1999]. On both the vision and language models, we observed that the two reference kernels tended to moderately compress all linear layers. As a result, the mean of each estimated density was centered around 0.5. In contrast, the Beta kernel selected a small set of relevant layers, applying slight compression to them while strongly suppressing the irrelevant ones (see Figures 9g and 9h). Given that our kernel's compression outperformed the two baseline kernels in Table 5, it implies that the optimal solution to model compression may lie near the boundaries of the unit hypercube, which played to the strengths of the Beta kernel.

4 RELATED WORK

RBF and Matérn Kernels. Various studies have been conducted to derive the regret bounds of the RBF and Matérn kernels [Srinivas et al., 2009, Scarlett et al., 2017, Scarlett, 2018, Belkin, 2018, Santin and Schaback, 2016, Cai and Scarlett, 2021, Vakili et al., 2021]. The key distinction between the two kernels lies in their rates of eigenvalue decay (termed eigendecay). The RBF kernel exhibits an exponential decay of eigenvalues, while the Matérn kernel's eigenvalues decay at a polynomial rate [Vakili et al., 2021] (see Figure 2). In this work, we numerically show that the proposed Beta kernel's eigendecay rate is exponential, which is similar to RBF's.

Non-stationary Kernels. Higdon et al. [1999] introduced a spatially evolving family of smoothing kernels on $\mathbb{R}^2$. Paciorek and Schervish [2003] extended the Matérn kernel into a non-stationary version on $\mathbb{R}^d$. Remes et al. [2017] proposed the non-stationary generalized spectral mixture kernel with input-dependent GP frequency surfaces. A common characteristic of these non-stationary kernels is that they are all defined on unbounded domains. In contrast, our Beta kernel is constructed from products of Beta distributions, which allows it to naturally capture a wide variety of

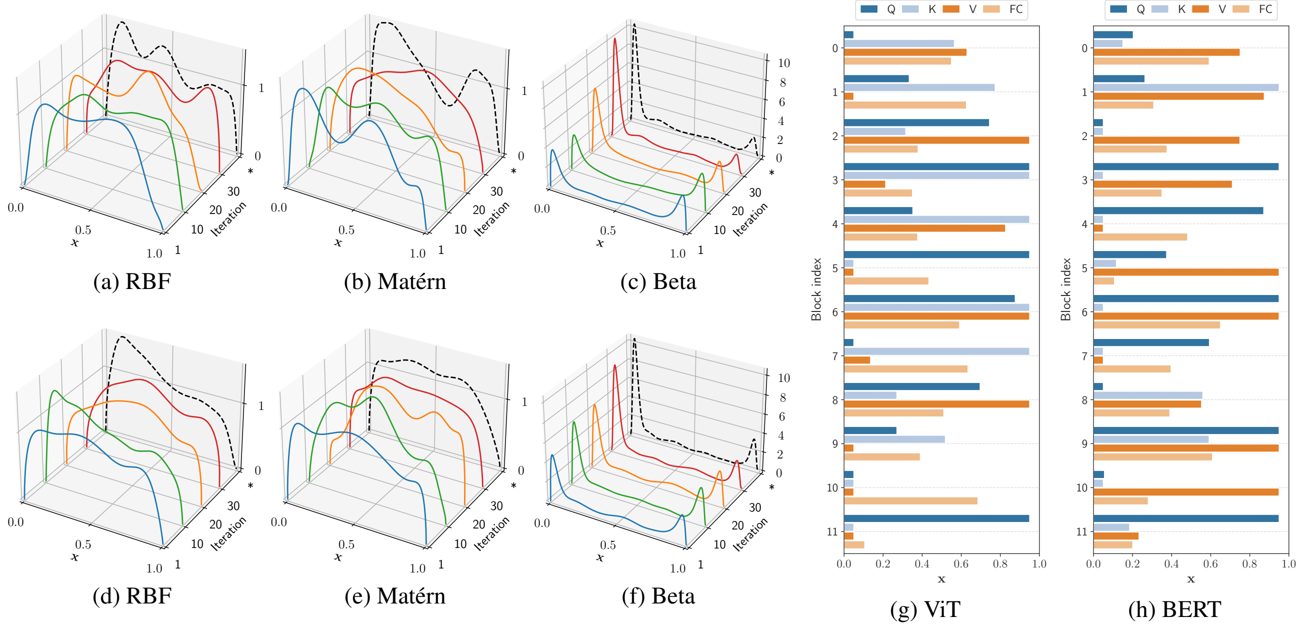


Figure 9: (a-f) Convergence comparison between different kernels. The rows (a-c) and (d-f) correspond to the ViT, and BERT compression tasks, respectively. The blue, green, orange, and red curves indicate the density estimates at iterations 1, 10, 20, and 30, respectively. $*$ represents $\arg\min_{t\in[T]} f(\mathbf{x}_t)$ with $T = 30$, and the dashed curves are the corresponding density estimates. (g-h) Detailed compression results using the Beta kernel. Q, K, V, and FC indicate "query", "key", "value", and fully connected layers, respectively.

smooth functions within bounded unit hypercubes. Such a formulation gives the Beta kernel a distinct advantage in being more sensitive to boundary regions.

5 CONCLUSION

We present a novel non-stationary kernel constructed from products of Beta probability density functions, whose close form is a product of Gamma functions. We provide empirical evidence indicating that the proposed kernel's eigendecay rate is exponential. Such an eigendecay rate is similar to RBF's, which was proved to have sub-linear regret bound by Vakili et al. [2021]. However, a primary limitation of our study is the absence of a formal regret bound for the proposed kernel. Future work should prioritize deriving these theoretical guarantees to deepen our understanding of the algorithm's performance and enhance its robustness. Our experiments indicate that the Beta kernel is robust in modeling functions with optima near faces or vertices of the unit hypercube. We show that our kernel substantially outperforms the two well-known kernels – RBF and Matern – on synthetic data as well as the model compression tasks. Our codebase is made publically available at https://github.com/imedslab/BetaKernel.

ACKNOWLEDGMENTS

The authors wish to acknowledge CSC—IT Center for Science, Finland, for generous computational resources. H.Z. and M.B.B. acknowledge support from the Flemish Government (AI Research Program) and the Research Foundation - Flanders (FWO) through project number G0G2921N. H.Z. is supported by the China Scholarship Council. A.T. and H.H.N. were supported by the Research Council of Finland (Profi6 336449 funding program), Sigrid Juselius foundation, University of Oulu strategic funds, and the European Union Horizon Programs (STAGE project, decision 101137146, and UltraSense project, Grant Agreement number 101130192). Finally, we gratefully acknowledge Prof. Simo Särkkä for insightful discussions on regret bounds and probability product kernels.

References

Milton Abramowitz and Irene A Stegun. *Handbook of mathematical functions with formulas, graphs, and mathematical tables*, volume 55. US Government printing office, 1968.

Thomas Back. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford university press, 1996.

Maximilian Balandat, Brian Karrer, Daniel R. Jiang, Samuel

Daulton, Benjamin Letham, Andrew Gordon Wilson, and Eytan Bakshy. BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization. In *Advances in Neural Information Processing Systems 33*, 2020. URL http://arxiv.org/abs/1910.06403.

Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. The second pascal recognising textual entailment challenge. In *Proceedings of the second PASCAL challenges workshop on recognising textual entailment*, volume 1. Citeseer, 2006.

Mikhail Belkin. Approximation beats concentration? an approximation view on inference with smooth radial kernels. In *Conference On Learning Theory*, pages 1348–1361. PMLR, 2018.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. The fifth pascal recognizing textual entailment challenge. *TAC*, 7(8):1, 2009.

Xu Cai and Jonathan Scarlett. On lower bounds for standard and robust gaussian process bandit optimization. In *International Conference on Machine Learning*, pages 1216–1226. PMLR, 2021.

Song Xi Chen. Beta kernel estimators for density functions. *Computational Statistics & Data Analysis*, 31(2):131–145, 1999.

Ido Dagan, Oren Glickman, and Bernardo Magnini. The pascal recognising textual entailment challenge. In *Machine learning challenges workshop*, pages 177–190. Springer, 2005.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Laurence Charles Ward Dixon. The global optimization problem: an introduction. *Towards Global Optimiation 2*, pages 1–15, 1978.

Bill Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Third international workshop on paraphrasing (IWP2005)*, 2005.

Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

David K Duvenaud, Hannes Nickisch, and Carl Rasmussen. Additive gaussian processes. *Advances in neural information processing systems*, 24, 2011.

Jacob Gardner, Geoff Pleiss, Kilian Q Weinberger, David Bindel, and Andrew G Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. *Advances in neural information processing systems*, 31, 2018.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and William B Dolan. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9, 2007.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*, 2020.

Tim Head, Gilles Louppe MechCoder, Iaroslav Shcherbatyi, et al. scikit-optimize/scikit-optimize: v0. 5.2. *Version v0*, 5, 2018.

D Higdon, J Swall, and J Kern. Non-stationary spatial modeling. bayesian statistics 6, eds. j. bernardo, j. berger, a. dawid, and a. smith, 1999.

Carl Hvarfner, Erik Orm Hellsten, and Luigi Nardi. Vanilla Bayesian optimization performs great in high dimensions. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 20793–20817. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/hvarfner24a.html.

Tony Jebara, Risi Kondor, and Andrew Howard. Probability product kernels. *The Journal of Machine Learning Research*, 5:819–844, 2004.

Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13:455–492, 1998.

Harold J Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. 1964.

Manuel Laguna and Rafael Marti. Experimental testing of advanced scatter search designs for global optimization of multimodal functions. *Journal of Global Optimization*, 33:235–255, 2005.

Yixiao Li, Yifan Yu, Qingru Zhang, Chen Liang, Pengcheng He, Weizhu Chen, and Tuo Zhao. Losparse: Structured compression of large language models based on low-rank and sparse approximation. In *International Conference on Machine Learning*, pages 20336–20350. PMLR, 2023.

Xingchen Ma, Amal Rannen Triki, Maxim Berman, Christos Sagonas, Jacques Cali, and Matthew B Blaschko. A bayesian optimization framework for neural network compression. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10274–10283, 2019.

Marcin Molga and Czesław Smutnicki. Test functions for optimization needs. *Test functions for optimization needs*, 101:48, 2005.

Donald E Myers. Spatial interpolation: an overview. *Geoderma*, 62(1-3):17–28, 1994.

ChangYong Oh, Efstratios Gavves, and Max Welling. Bock: Bayesian optimization with cylindrical kernels. In *International Conference on Machine Learning*, pages 3868–3877. PMLR, 2018.

Art B Owen. Scrambling sobol'and niederreiter–xing points. *Journal of complexity*, 14(4):466–489, 1998.

Christopher Paciorek and Mark Schervish. Nonstationary covariance functions for gaussian process regression. *Advances in neural information processing systems*, 16, 2003.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003.

Sami Remes, Markus Heinonen, and Samuel Kaski. Nonstationary spectral kernels. *Advances in neural information processing systems*, 30, 2017.

Gabriele Santin and Robert Schaback. Approximation of eigenfunctions in kernel-based spaces. *Advances in Computational Mathematics*, 42(4):973–993, 2016.

Jonathan Scarlett. Tight regret bounds for bayesian optimization in one dimension. In *International Conference on Machine Learning*, pages 4500–4508. PMLR, 2018.

Jonathan Scarlett, Ilija Bogunovic, and Volkan Cevher. Lower bounds on regret for noisy gaussian process bandit optimization. In *Conference on Learning Theory*, pages 1723–1742. PMLR, 2017.

Jasper Snoek, Kevin Swersky, Rich Zemel, and Ryan Adams. Input warping for bayesian optimization of non-stationary functions. In *International conference on machine learning*, pages 1674–1682. PMLR, 2014.

Il'ya Meerovich Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, 7(4): 784–802, 1967.

Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

Michael L Stein. *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media, 2012.

Kevin Swersky. *Improving Bayesian optimization for machine learning using expert priors*. University of Toronto (Canada), 2017.

Sattar Vakili, Kia Khezeli, and Victor Picheny. On information gain and regret bounds in gaussian process bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 82–90. PMLR, 2021.

Alex Wang. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Adina Williams, Nikita Nangia, and Samuel R Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.

Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

Andrew Wilson and Ryan Adams. Gaussian process kernels for pattern discovery and extrapolation. In *International conference on machine learning*, pages 1067–1075. PMLR, 2013.

Han Zhou, Xingchen Ma, and Matthew B Blaschko. A corrected expected improvement acquisition function under noisy observations. In *Asian Conference on Machine Learning*, pages 1747–1762. PMLR, 2024.

Bayesian Optimization over Bounded Domains with the Beta Product Kernel (Supplementary Material)

Huy Hoang Nguyen[1] Han Zhou[2] Matthew B. Blaschko[*2] Aleksei Tiulpin[†1]

[1]Research Unit of Health Sciences and Technology, Finland, University of Oulu, Oulu, Finland
[2]Department ESAT, Center for Processing Speech and Images, KU Leuven, Leuven, Belgium

Table S1: Training details of different models.

Model	Training Steps	Batch Size	Time per Evaluation	Num of Training Samples
GPT-2	512	25	55 mins	12,800
BERT	256	100	40 mins	25,600
DeBERTa-v3	256	100	30 mins	25,600
ViT	1 epoch	200	7 mins	Full training set

A PROOF OF UPPER BOUND OF $K_\beta(\mathbf{x}, \mathbf{x})$

A.1 PROOF OF LEMMA 1

Lemma 1. *For gamma function $\Gamma(\cdot)$, we have that*

$$\frac{\Gamma(2x+1)}{\Gamma^2(x+1)} = \frac{2^{2x}\Gamma(x+\frac{1}{2})}{\sqrt{\pi}\Gamma(x+1)} \tag{1}$$

Proof. We start by utilizing the duplication formula for the Gamma function, which states:

$$\Gamma(z)\Gamma\left(z+\frac{1}{2}\right) = 2^{1-2z}\sqrt{\pi}\Gamma(2z)$$

Applying the duplication formula to our specific case by setting $z = x + \frac{1}{2}$, we get:

$$\Gamma\left(x+\frac{1}{2}\right)\Gamma(x+1) = 2^{-2x}\sqrt{\pi}\Gamma(2x+1).$$

Dividing both sides by $\Gamma^2(x+1)$, we obtain:

$$\frac{\Gamma(2x+1)}{\Gamma^2(x+1)} = \frac{2^{2x}\Gamma\left(x+\frac{1}{2}\right)}{\sqrt{\pi}\Gamma(x+1)}.$$

which completes the proof. □

[*]Equal last author
[†]Equal last author

A.2 PROOF OF LEMMA 2

Lemma 2. *For $x \geq 0$ and $0 < s < 1$, it holds that:*

$$\left(\frac{2}{2x+1}\right)^{\frac{1}{2}} \leq \frac{\Gamma(x+\frac{1}{2})}{\Gamma(x+1)} \leq 2.$$

Proof. The Wendel's inequality is stated as

$$\left(\frac{z}{z+s}\right)^{1-s} \leq \frac{1}{z^s} \cdot \frac{\Gamma(z+s)}{\Gamma(z)} \leq 1, \tag{2}$$

where $z > 0$ and $s \in (0,1)$. When $s = \frac{1}{2}$, it is equivalent to

$$1 \leq z^{\frac{1}{2}} \frac{\Gamma(z)}{\Gamma(z+\frac{1}{2})} \leq \left(\frac{z+\frac{1}{2}}{z}\right)^{\frac{1}{2}} \tag{3}$$

$$z^{-\frac{1}{2}} \leq \frac{\Gamma(z)}{\Gamma(z+\frac{1}{2})} \leq z^{-\frac{1}{2}}\left(\frac{1}{2z}+1\right)^{\frac{1}{2}}. \tag{4}$$

Apply the inequality with $z = x + \frac{1}{2}$ and $s = \frac{1}{2}$, we have that

$$\left(\frac{2}{2x+1}\right)^{\frac{1}{2}} \leq \frac{\Gamma\left(x+\frac{1}{2}\right)}{\Gamma\left(x+1\right)} \leq \left(x+\frac{1}{2}\right)^{-\frac{1}{2}}\left(\frac{1}{2x+1}+1\right)^{\frac{1}{2}} \tag{5}$$

$$= \left[\frac{2}{2x+1}\left(\frac{1}{2x+1}+1\right)\right]^{\frac{1}{2}} =: g(x). \tag{6}$$

As $g(x)$ is a decreasing function on $[0, \infty)$, thus $g(0) = \max_{x \geq 0} g(x) = 2$, which concludes the proof. ☐

A.3 PROOF OF PROPOSITION 1

Proposition 3. $K_\beta(\mathbf{x}, \mathbf{x}) = \mathcal{O}(2^{2d - \frac{2d}{h}} h^{-\frac{3d}{2}}), \forall \mathbf{x} \in [0,1]^d$.

Proof. For any $\mathbf{x} = (x_1, \ldots, x_d) \in [0,1]^d$, $K_\beta(\mathbf{x}, \mathbf{x})$ is expressed as

$$K_\beta(\mathbf{x}, \mathbf{x}) = C \prod_{i=1}^{d} \frac{\Gamma(2\frac{x_i}{h_i}+1)\Gamma(2\frac{1-x_i}{h_i}+1)}{\Gamma^2(\frac{x_i}{h_i}+1)\Gamma^2(\frac{1-x_i}{h_i}+1)},$$

where

$$C = \frac{\Gamma^2(\frac{1}{h_i}+2)}{\Gamma(\frac{2}{h_i}+2)}. \tag{7}$$

By utilizing Lemma 1 with $\frac{x_i}{h_i}$ and $\frac{1-x_i}{h_i}$, we have that

$$K_\beta(\mathbf{x}, \mathbf{x}) = C \prod_{i=1}^{d} \frac{2^{2\frac{x_i}{h_i}+2(1-\frac{x_i}{h_i})}\Gamma(\frac{x_i}{h_i}+\frac{1}{2})\Gamma(\frac{1-x_i}{h_i}+\frac{1}{2})}{\pi\Gamma(\frac{x_i}{h_i}+1)\Gamma(\frac{1-x_i}{h_i}+1)} \tag{8}$$

$$= C \prod_{i=1}^{d} \frac{4}{\pi} \frac{\Gamma(\frac{x_i}{h_i}+\frac{1}{2})\Gamma(\frac{1-x_i}{h_i}+\frac{1}{2})}{\Gamma(\frac{x_i}{h_i}+1)\Gamma(\frac{1-x_i}{h_i}+1)} \tag{9}$$

When applying Lemma 2, we have that

$$\frac{\Gamma(\frac{x_i}{h_i}+\frac{1}{2})\Gamma(\frac{1-x_i}{h_i}+\frac{1}{2})}{\Gamma(\frac{x_i}{h_i}+1)\Gamma(\frac{1-x_i}{h_i}+1)} \leq 4 \tag{10}$$

Therefore, we can derive that

$$
\begin{aligned}
K_\beta(\mathbf{x},\mathbf{x}) &\le \frac{16^d}{\pi^d}\prod_{i=1}^{d}\frac{\Gamma^2(\frac{1}{h_i}+2)}{\Gamma(\frac{2}{h_i}+2)} && && (11)\\
&= \frac{16^d}{\pi^d}\prod_{i=1}^{d}\frac{(\frac{2}{h_i}+2)\Gamma^2(\frac{1}{h_i}+2)}{\Gamma(\frac{2}{h_i}+2+1)} && \text{by } \Gamma(x+1)=x\Gamma(x) && (12)\\
&= \frac{16^d}{\pi^d}\prod_{i=1}^{d}\frac{(\frac{2}{h_i}+2)\sqrt{\pi}\Gamma(\frac{1}{h_i}+2)}{2^{2(\frac{1}{h_i}+1)}\Gamma(\frac{1}{h_i}+\frac{3}{2})} && \text{by Lemma 1 with } x=\frac{1}{h_i}+1 && (13)\\
&= \frac{2^{2d}}{\pi^{\frac{d}{2}}}\prod_{i=1}^{d}\frac{(\frac{2}{h_i}+2)\Gamma(\frac{1}{h_i}+2)}{2^{\frac{2}{h_i}}\Gamma(\frac{1}{h_i}+\frac{3}{2})} && && (14)\\
&\le \frac{2^{3d}}{\pi^{\frac{d}{2}}}\prod_{i=1}^{d}\frac{(\frac{1}{h_i}+1)(\frac{1}{h_i}+\frac{3}{2})^{\frac{1}{2}}}{2^{\frac{2}{h_i}}} && \text{by Lemma 2 } \frac{\Gamma(\frac{1}{h_i}+2)}{\Gamma(\frac{1}{h_i}+\frac{3}{2})}\le\left(\frac{1}{h_i}+\frac{3}{2}\right)^{\frac{1}{2}} && (15)\\
&= 2^{3d-\frac{2d}{h}}\left(\frac{1}{h}+1\right)^d\left(\frac{1}{h\pi}+\frac{3}{2\pi}\right)^{\frac{d}{2}} && && (16)
\end{aligned}
$$

Therefore, we conclude the proof. □