Gaussian Process Emulation for an Interest Rate Swap Portfolio: Active Learning, Dimension Reduction, and Online Updates

University of California, Santa Barbara

March 21, 2026

Abstract

We construct a Gaussian process (GP) surrogate model for a ten-instrument USD fixed-floating interest rate swap (IRS) portfolio augmented with a European payer swaption, treating the QuantLib full-valuation pricing engine as a black-box computer simulator. The emulator maps a ten-dimensional space of parallel and non-parallel yield curve shifts to an eleven-dimensional net present value (NPV) output vector. We proceed through four methodological stages: (i) space-filling Latin hypercube design with baseline GP emulation; (ii) variance-based active learning to improve sample efficiency; (iii) probabilistic principal component analysis for dimensionality reduction in a synthetic 22-dimensional setting; and (iv) sequential online GP updates via rank-one fantasy conditioning analogous to a Kalman filter measurement step. Starting from a baseline portfolio root mean squared error (RMSE) of \$1,746 on 50 training points, active learning reduces RMSE to \$575 after 100 additional simulator evaluations—a 67.1% improvement over the baseline and 10.5 percentage points better than passive sampling. Online fantasy conditioning further reduces RMSE to \$159 with 100% empirical coverage of 95% credible intervals.

1 Introduction

1.1 Motivation

Full-valuation risk engines in quantitative finance reprice entire portfolios under thousands of stress scenarios to estimate value-at-risk, expected shortfall, and Greeks. Each scenario requires bootstrapping a new yield curve, constructing all instruments, and solving for net present values—a process that is accurate but computationally expensive [13]. When a portfolio contains instruments with nonlinear payoffs such as swaptions, the number of required scenarios grows rapidly, and brute-force simulation becomes a bottleneck for real-time risk management.

Gaussian process (GP) surrogate models offer a principled alternative: a small number of simulator evaluations at carefully chosen input points suffice to train a probabilistic emulator that predicts the simulator output at untried inputs with calibrated uncertainty [24, 18]. The GP

framework developed in the uncertainty quantification literature [9, 11] provides both point predictions and posterior variance, enabling downstream decisions about where to sample next and how much to trust the approximation.

1.2 Research goals and contributions

This paper applies the GP emulation framework to a realistic USD interest rate swap portfolio and investigates four questions:

1. How accurately can a GP surrogate trained on a Latin hypercube design [21] approximate the QuantLib pricing engine across a $[\pm 150 \text{ bps}]^{10}$ input space?
2. Does variance-based active learning [28] yield better sample efficiency than passive space-filling designs?
3. Can probabilistic PCA [29] or partial least squares reduce the effective input dimension without sacrificing emulator accuracy?
4. How effectively can rank-one fantasy conditioning—analogous to a Kalman filter measurement update [15]—incorporate new observations without full retraining?

The main contributions are: (a) a working GP emulator for a multi-instrument IRS portfolio with a nonlinear swaption component; (b) empirical comparison of active versus passive sequential design in ten dimensions; (c) a controlled experiment isolating the effect of signal-to-noise ratio on PCA-based dimension reduction; and (d) demonstration of online GP updates with calibrated coverage.

1.3 Paper organization

Section 2 reviews the relevant literature on GP emulation, dimension reduction, and active learning. Section 3 describes the portfolio simulator, the GP model, experimental design strategies, dimension reduction pipelines, and the online update mechanism. Section 4 presents numerical results across all four stages. Section 5 interprets findings, notes limitations, and discusses implications. Section 6 summarizes the work and outlines directions for future research.

2 Literature Review

2.1 Gaussian process emulation and RobustGaSP

The use of Gaussian processes as surrogate models for deterministic computer experiments was formalized by Sacks et al. [24], who showed that a GP interpolator provides both a predictor and an assessment of prediction uncertainty at untried inputs. The foundational Bayesian calibration framework of Kennedy and O'Hagan [18] extended this to problems where the simulator is compared with physical observations, introducing model discrepancy and observation error.

Gu et al. [9] developed the RobustGaSP framework, which addresses a persistent practical difficulty: estimating GP covariance parameters reliably when the training set is small relative to the input dimension. They derived objective priors—reference priors and jointly robust priors—for range and variance parameters in product-form anisotropic correlation functions, and proved posterior propriety under Latin hypercube designs. Their key finding is that certain parameterizations of the covariance function yield marginal posteriors with better-behaved modes, leading to more stable maximum marginal likelihood estimation. The theoretical foundations of these results were extended in Barthelmé et al. [33], who analyzed GP regression in the flat limit where lengthscales tend to infinity and established conditions under which the posterior remains well-defined.

Berger et al. [2] established the foundations for objective Bayesian analysis of spatially correlated data, deriving reference priors for covariance parameters of stationary Gaussian processes. These results underpin the prior specifications used in modern GP emulation packages.

For multi-output computer models—such as the eleven-dimensional NPV vector in our application—Conti and O'Hagan [5] proposed a Bayesian emulation framework that models cross-output dependencies, while Gu and Berger [8] developed parallel partial GP emulation for massive output dimensions.

Practical GP implementations rely on efficient linear algebra. GPyTorch [7] provides GPU-accelerated exact and approximate GP inference via the LOVE algorithm and Lanczos-based kernel matrix solves, enabling training on datasets of the size considered here. The KISS-GP framework of Wilson and Nickisch [31] introduced kernel interpolation for scalable structured GPs, while Snelson and Ghahramani [26] developed the foundational sparse GP approach using pseudo-inputs.

2.2 Dimension reduction via (generalized) probabilistic PCA

When the input space has many dimensions but the simulator response depends primarily on a lower-dimensional subspace, projecting inputs before fitting the GP can mitigate the curse of dimensionality. Constantine et al. [4] introduced active subspace methods that identify the dominant directions of variability in the gradient of the simulator output with respect to inputs, providing a principled linear reduction of the input space.

Tipping and Bishop [29] formulated probabilistic PCA (PPCA) as a latent variable model in which observed data $\mathbf{x} \in \mathbb{R}^D$ are generated from a latent variable $\mathbf{z} \in \mathbb{R}^Q$ through a linear mapping corrupted by isotropic Gaussian noise:

$$\mathbf{x} = \mathbf{W}\mathbf{z} + \boldsymbol{\mu} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}). \tag{1}$$

The EM algorithm alternates between computing the posterior $p(\mathbf{z} \mid \mathbf{x})$ (E-step) and updating $\mathbf{W}$ and σ^2 (M-step). In the limit $\sigma^2 \to 0$, the columns of $\mathbf{W}$ span the same subspace as the top Q principal components from standard PCA. The advantage of the probabilistic formulation is that σ^2 is estimated alongside the projection, providing a principled measure of reconstruction error.

Gu and Xu [10] extended PPCA to the generalized probabilistic PCA (GPPCA) setting, where

the latent factors follow correlated Gaussian processes rather than independent standard normals. This extension is natural for spatio-temporal data but also applies whenever the latent structure has correlation that should be modeled. The scalable marginalization approach of Gu and Wang [12] further addresses computational challenges when the number of latent variables is large.

For supervised dimension reduction, partial least squares (PLS) finds the directions in input space that maximize the covariance with the output [32]. Unlike PCA, PLS uses the response variable to guide the projection, which can be advantageous when a few input directions disproportionately influence the output.

2.3 Active learning for computer experiments

Sequential experimental design for computer experiments aims to select the next evaluation point to maximize some criterion of the current emulator. In the surrogate-building context, the most natural criterion is the integrated or maximum predictive variance, which reduces global uncertainty as rapidly as possible [24]. This is distinct from Bayesian optimization, where the acquisition function (e.g., expected improvement) targets the global optimum of the response surface [6, 27].

Sung et al. [28] developed active learning with error control (ALEC) for reliable emulation of complex functionals. Their approach bounds the maximum predictive error at a user-specified tolerance and terminates when the bound is satisfied everywhere, substantially outperforming space-filling designs in high dimensions. Sauer et al. [25] proposed active learning for deep GP surrogates, extending the sequential design paradigm to more flexible nonparametric models.

The initial space-filling design is itself important. McKay et al. [21] introduced Latin hypercube sampling (LHS), which ensures that each input dimension is uniformly covered by partitioning each marginal into N equal-probability strata and sampling one point per stratum. LHS provides better coverage than simple random sampling in moderate dimensions and serves as the standard initial design in computer experiments.

3 Model and Methodology

3.1 Interest rate swap portfolio and simulator

The simulator is a QuantLib-based pricing engine for a portfolio of ten USD fixed-floating interest rate swaps spanning tenors of 1, 2, 3, 5, 7, 10, 15, 20, 25, and 30 years, augmented with a European payer swaption with 1-year expiry and 5-year underlying tenor [22]. All swaps are payer instruments with notional \$1,000,000 struck at par rates so that the base NPV of each swap is approximately zero. The yield curve is bootstrapped from ten CMT par-swap rate helpers using `ql.PiecewiseLinearZero` with the SOFR overnight index as the floating leg reference.

The base rates (in percent) used for curve construction are: 4.30 (1Y), 4.10 (2Y), 3.95 (3Y), 3.90 (5Y), 3.92 (7Y), 4.00 (10Y), 4.15 (15Y), 4.25 (20Y), 4.28 (25Y), and 4.30 (30Y). The swaption is priced under a Bachelier (normal) volatility model at a fixed strike of 3.90% with 60 basis points of normal volatility.

The simulator function `price_portfolio` accepts a ten-dimensional input vector $\mathbf{x} \in [-150, +150]^{10}$ (basis points) representing non-parallel shifts to each CMT tenor rate. It re-bootstraps the yield curve under the shifted rates and returns an eleven-dimensional NPV vector: ten IRS NPVs plus the swaption NPV. The ten IRS instruments are approximately linear in the rate shifts (DV01 exposure), while the swaption introduces genuine nonlinearity via the payoff's convexity near the strike. This nonlinearity is the key motivation for including the swaption: it creates regions of elevated GP posterior variance that the active learning acquisition function can exploit.

A test evaluation at +100 bps parallel shift produces a 10-year IRS NPV of \$77,501 and a swaption NPV of \$38,745, confirming that payer swaps gain value when rates rise and that the swaption is correctly priced [13, 3].

3.2 Gaussian process surrogate model

3.2.1 Prior, likelihood, and covariance functions

We fit an independent GP to each of the eleven output dimensions. For the k-th output, the model is

$$y_k(\mathbf{x}) = f_k(\mathbf{x}) + \epsilon_k, \quad f_k \sim \mathcal{GP}(m_k, k_\theta(\cdot,\cdot)), \quad \epsilon_k \sim \mathcal{N}(0, \sigma_n^2), \tag{2}$$

where m_k is a constant mean function and k_θ is a scaled squared-exponential (RBF) kernel with automatic relevance determination (ARD):

$$k_\theta(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\Big(-\frac{1}{2}\sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{\ell_d^2}\Big). \tag{3}$$

Here σ_f^2 is the output scale, ℓ_d is the lengthscale for the d-th input dimension, and σ_n^2 is the noise variance. ARD lengthscales allow the model to learn which input dimensions most strongly influence each output, an important property for the portfolio application where short-tenor IRS instruments are insensitive to long-tenor rate shifts.

Although the simulator is deterministic, we retain a small noise term σ_n^2 for numerical stability of the Cholesky factorization of the kernel matrix [23].

All inputs are normalized to $[0, 1]^D$ before training using the known bounds $[-150, +150]$ bps, and all outputs are standardized to zero mean and unit variance. This normalization is critical for ARD: without it, the lengthscale magnitudes would reflect input scaling rather than genuine sensitivity [9].

3.2.2 Hyperparameter estimation

Hyperparameters $\theta = \{\sigma_f^2, \ell_1, \ldots, \ell_D, \sigma_n^2, m_k\}$ are estimated by maximizing the log marginal likelihood:

$$\log p(\mathbf{y} \mid \mathbf{X}, \theta) = -\frac{1}{2}\mathbf{y}^\top \mathbf{K}_\theta^{-1}\mathbf{y} - \frac{1}{2}\log|\mathbf{K}_\theta| - \frac{N}{2}\log(2\pi), \tag{4}$$

where $\mathbf{K}_\theta = k_\theta(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}_N$. Optimization is performed via 150 iterations of Adam [19] with learning rate 0.1, as implemented in GPyTorch [7].

The marginal likelihood balances data fit (first term) against model complexity (second term), providing an automatic Occam's razor that penalizes both underfitting and overfitting [23]. The objective prior framework of Gu et al. [9] motivates the use of marginal likelihood estimation for range parameters, and Berger et al. [2] established propriety conditions that ensure well-defined posteriors for these parameters.

3.3 Experimental design

3.3.1 Latin hypercube sampling

The initial training set consists of $N = 50$ points drawn from a Latin hypercube sample (LHS) over $[-150, +150]^{10}$ using the `scipy.stats.qmc.LatinHypercube` implementation. An independent validation set of $N_{\text{val}} = 200$ LHS points is used throughout for unbiased RMSE evaluation. LHS is the standard initial design in computer experiments because it guarantees uniform marginal coverage in each input dimension with only N evaluations [21], providing a good starting point for GP emulation even in moderate dimensions [24].

3.3.2 Variance-based active learning

After training the baseline GP on the LHS design, we apply a sequential active learning procedure over five rounds, adding 20 points per round. The acquisition function is the total predictive variance:

$$\alpha(\mathbf{x}) = \sum_{k=1}^{K} \frac{\hat{\sigma}_k^2(\mathbf{x})}{s_k^2}, \tag{5}$$

where $\hat{\sigma}_k^2(\mathbf{x})$ is the posterior predictive variance of the k-th GP at input $\mathbf{x}$ and s_k^2 is the training-set variance of the k-th output, serving as a normalization factor so that outputs of different magnitudes contribute equally.

This criterion selects points where the emulator is most uncertain, which is the appropriate strategy for surrogate building as opposed to optimization [24, 28]. Expected improvement, which targets points where the GP predicts large improvements over the current best output, is designed for optimization and would cluster points at the corners of the input space where |NPV| is largest, leaving the interior poorly covered.

At each round, a candidate pool of 1,000 fresh LHS points is scored. To avoid batch clustering, we use a sequential greedy procedure: after selecting the highest-scoring candidate, all remaining candidates' scores are penalized by a squared-exponential distance decay $\exp(-\|\mathbf{x} - \mathbf{x}_{\text{selected}}\|^2 / 2\tilde{d})$, where $\tilde{d}$ is the median pairwise squared distance among candidates. This encourages spatial diversity in the selected batch [28].

After each round, the GP hyperparameters are warm-started from the previous round's fitted values before re-optimization. Warm-starting is important because as the training set grows from

$N = 50$ to $N = 150$, the marginal likelihood landscape shifts, and 150 Adam iterations from a random initialization may be insufficient for convergence—a failure mode we observed empirically when warm-starting was omitted.

The passive baseline uses the same budget: at each round, 20 new uniformly sampled LHS points are added and the GP is retrained from scratch. Both methods start from the same initial $N = 50$ training set, so the comparison is controlled for total simulator evaluations.

3.4 Dimension reduction

3.4.1 Principal component analysis

To study the effect of input dimension on emulator accuracy, we construct a synthetic 22-dimensional input space. The first ten dimensions are the pricing-relevant rate shifts drawn from $[-150, +150]$ bps; the remaining twelve dimensions are pure independent noise drawn from $[-15, +15]$ (eleven dimensions) and $[-5, +5]$ (one dimension). Only the first ten dimensions enter the QuantLib pricing function. This creates a signal variance of approximately 7,500 in the pricing-relevant dimensions and a noise variance of approximately 75 in the irrelevant dimensions, yielding a variance ratio of 100×—sufficient for PCA to separate the two groups.

Standard PCA computes the eigendecomposition of the sample covariance matrix $\hat{\mathbf{\Sigma}} = \frac{1}{N-1}\mathbf{X}_c^\top \mathbf{X}_c$ and projects onto the top $Q = 10$ eigenvectors. We fit a separate GP on the Q-dimensional projected inputs for each output dimension. The scree plot (Figure 3, left panel) shows a clear elbow at $Q = 10$, confirming that the 100× variance ratio is sufficient for eigenvalue separation.

3.4.2 Probabilistic PCA model and EM algorithm

We implement the Tipping–Bishop [29] EM algorithm for probabilistic PCA. Given centered data $\mathbf{X}_c \in \mathbb{R}^{N \times D}$, the E-step computes the posterior latent coordinates:

$$\mathbf{M} = \mathbf{W}^\top \mathbf{W} + \sigma^2 \mathbf{I}_Q, \quad \mathbb{E}[\mathbf{z}_n \mid \mathbf{x}_n] = \mathbf{M}^{-1}\mathbf{W}^\top (\mathbf{x}_n - \boldsymbol{\mu}). \tag{6}$$

The M-step updates $\mathbf{W}$ and σ^2:

$$\mathbf{W}_{\text{new}} = \Big(\mathbf{X}_c^\top \mathbf{Z}\Big)\Big(N\sigma^2 \mathbf{M}^{-1} + \mathbf{Z}^\top \mathbf{Z}\Big)^{-1}, \quad \sigma^2_{\text{new}} = \frac{1}{ND}\sum_{n=1}^{N} \|\mathbf{x}_n - \boldsymbol{\mu} - \mathbf{W}_{\text{new}}\mathbb{E}[\mathbf{z}_n]\|^2 + \frac{\sigma^2}{D}\,\text{tr}(\mathbf{M}^{-1}). \tag{7}$$

The algorithm converges when the maximum absolute change in $\mathbf{W}$ and σ^2 falls below 10^{-6}, or after 200 iterations. New test points are projected as $\mathbf{z}_* = \mathbf{M}^{-1}\mathbf{W}^\top(\mathbf{x}_* - \boldsymbol{\mu})$.

For comparison, we also run PLS regression [32] with $Q_{\text{PLS}} = 10$ components using the full 11-output target matrix, with `scale=False` to preserve the variance gap between signal and noise dimensions.

The generalized PPCA framework of Gu and Xu [10] extends PPCA to settings where latent factors are correlated, providing a richer model; our implementation uses the standard Tipping–

Bishop formulation as a baseline.

3.5 Online updates

3.5.1 Sequential prediction and uncertainty

In a live risk management setting, new market observations arrive sequentially and the GP posterior must be updated without refitting from scratch. Given a trained GP with kernel matrix $\mathbf{K}$ and Cholesky factor $\mathbf{L}$, a new observation $(\mathbf{x}_{N+1}, y_{N+1})$ requires solving a system of size $(N+1)\times(N+1)$. Direct refitting costs $O(N^3)$, but a rank-one update to the Cholesky factor costs only $O(N^2)$ [23].

3.5.2 Fantasy model / Kalman-style update

GPyTorch's `get_fantasy_model` implements this rank-one Cholesky update. Given the current GP posterior, the fantasy model conditions on the new observation by appending a row and column to $\mathbf{K}$ and performing a single backsubstitution:

$$\mathbf{L}_{N+1} = \begin{pmatrix} \mathbf{L}_N & \mathbf{0} \\ \mathbf{l}_{N+1}^\top & l_{N+1,N+1} \end{pmatrix}, \tag{8}$$

where $\mathbf{l}_{N+1} = \mathbf{L}_N^{-1}\mathbf{k}_{N+1}$ and $l_{N+1,N+1} = \sqrt{k(\mathbf{x}_{N+1}, \mathbf{x}_{N+1}) + \sigma_n^2 - \|\mathbf{l}_{N+1}\|^2}$. This is exactly analogous to the Kalman filter measurement update [15], where the state estimate and covariance are updated with a new observation at cost linear in the state dimension [20].

We simulate ten sequential "intraday" market moves as a cumulative random walk: $\mathbf{x}_t = \mathrm{clip}\big(\sum_{s=1}^{t} \boldsymbol{\delta}_s,\, -150,\, +150\big)$ where $\boldsymbol{\delta}_s \sim \mathcal{N}(\mathbf{0}, 25\mathbf{I}_{10})$. At each step, the GP predicts the 10-year IRS NPV before observing the true value, then updates via fantasy conditioning. We track the predictive mean, the $\pm 2\sigma$ credible interval, and whether the true value falls within the interval.

4 Numerical Experiments

4.1 Baseline GP Emulator on LHS Design

We train eleven independent GPs on the $N = 50$ LHS design and evaluate on the $N_{\text{val}} = 200$ validation set. Table 1 reports the estimated hyperparameters. All ten IRS GPs have median ARD lengthscales between 10.2 and 10.6, minimum lengthscales between 0.64 and 0.69, and maximum lengthscales between 10.6 and 11.1, indicating that the GP assigns relatively uniform importance across the ten rate-shift dimensions. The swaption GP has a similar median lengthscale of 10.3 but a substantially larger noise variance (2.88×10^{-4} versus $\approx 1.2\times10^{-4}$ for the IRS outputs), reflecting the swaption's greater nonlinearity. Output scales range from 0.54 to 0.72 for IRS instruments and reach 1.24 for the swaption.

Table 1: Estimated GP hyperparameters for the baseline emulator ($N = 50$).

Instrument	Median ℓ	Min ℓ	Max ℓ	Output Scale σ_f^2	Noise σ_n^2
1Y	10.511	0.650	10.673	0.5432	0.000119
2Y	10.424	0.683	10.848	0.6084	0.000119
3Y	10.591	0.660	10.922	0.5737	0.000119
5Y	10.335	0.664	10.736	0.5760	0.000119
7Y	10.424	0.665	10.934	0.6009	0.000120
10Y	10.339	0.656	10.704	0.6045	0.000120
15Y	10.417	0.685	10.639	0.6792	0.000122
20Y	10.209	0.669	11.058	0.7049	0.000121
25Y	10.385	0.640	10.992	0.6922	0.000123
30Y	10.393	0.646	10.739	0.7174	0.000123
Swaption	10.295	0.659	10.633	1.2411	0.000288

Table 2 reports the per-instrument and portfolio-level validation RMSE. The IRS instruments exhibit RMSE that scales roughly with tenor: \$30 for 1Y up to \$945 for 30Y. Relative to the mean absolute NPV, all IRS RMSEs are below 0.8%. The swaption RMSE is \$575 (3.8% relative), reflecting its nonlinear payoff. The portfolio RMSE—computed as the RMSE of the sum of all eleven predicted NPVs versus the sum of all true NPVs—is \$1,746, or 0.8% of the mean absolute portfolio value of \$208,574.

Table 2: Baseline GP validation RMSE ($N = 50$, $N_{\text{val}} = 200$).

Instrument	GP RMSE (\$)	Mean |NPV| (\$)	RMSE / Mean |NPV|
1Y	30	7,226	0.4%
2Y	52	14,039	0.4%
3Y	117	20,753	0.6%
5Y	182	33,327	0.5%
7Y	233	45,073	0.5%
10Y	385	60,954	0.6%
15Y	503	83,069	0.6%
20Y	710	100,680	0.7%
25Y	705	114,523	0.6%
30Y	945	126,092	0.7%
Swaption	575	15,165	3.8%
Portfolio	1,746	208,574	0.8%

Figure 1 illustrates the space-filling properties of the $N = 50$ LHS design by showing pairwise marginal scatter plots for four representative tenor pairs: 1Y vs 30Y, 2Y vs 7Y, 3Y vs 10Y, and 5Y vs 15Y. Points are well spread across the $[-150, +150]$ bps input domain in all pairwise projections, with no clustering or gaps, confirming that LHS achieves good marginal coverage across the ten-dimensional rate-shift space [21].

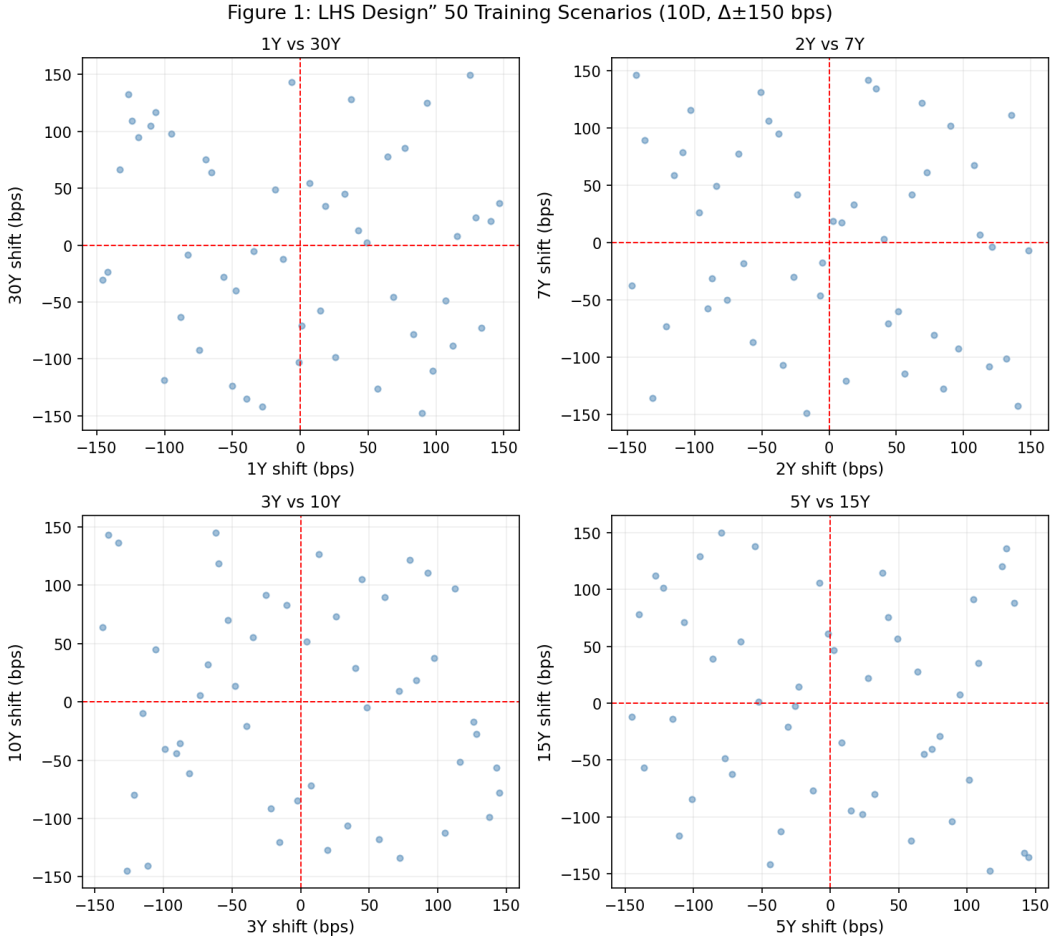

Figure 1: Pairwise marginal scatter plots of the $N = 50$ LHS training design across four representative tenor pairs. Each point is one training scenario; red dashed lines mark the zero shift. The uniform spread across all four panels confirms the space-filling property of the LHS design.

4.2 Effect of active learning on emulator accuracy

Figure 2 shows the convergence of portfolio RMSE as the training set grows from $N = 50$ to $N = 150$ over five rounds. Active learning (ALM) reduces RMSE monotonically at every round: \$1,746 $\rightarrow$ \$921 $\rightarrow$ \$812 $\rightarrow$ \$698 $\rightarrow$ \$632 $\rightarrow$ \$575. The passive LHS baseline also improves monotonically but more slowly: \$1,746 $\rightarrow$ \$1,275 $\rightarrow$ \$1,032 $\rightarrow$ \$861 $\rightarrow$ \$833 $\rightarrow$ \$758.

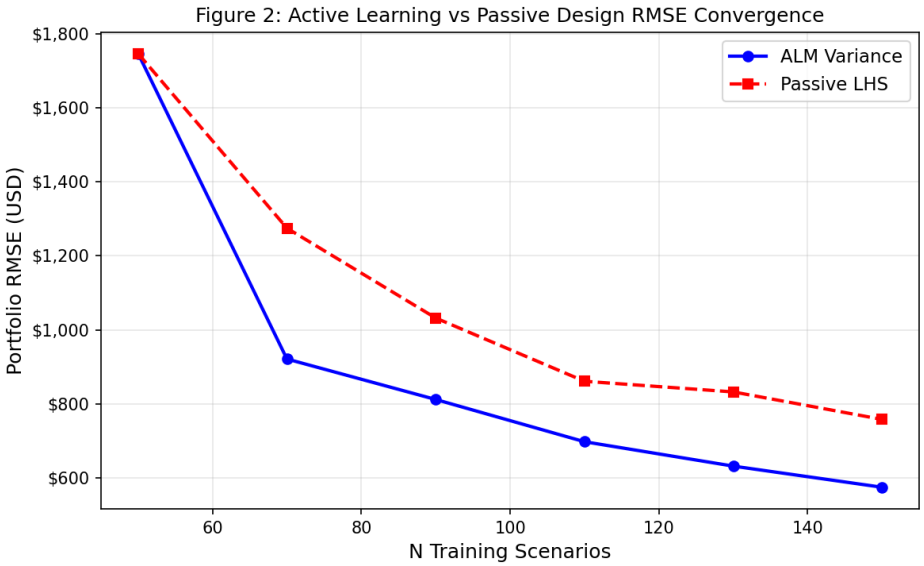

Figure 2: Active learning (ALM) versus passive LHS: portfolio RMSE as a function of training set size over five acquisition rounds.

Table 3 summarizes the comparison. ALM achieves a 67.1% reduction from the baseline, compared with 56.6% for passive LHS—an absolute gap of 10.5 percentage points and a relative improvement of $183 in RMSE at the same total budget of 150 simulator evaluations.

Table 3: Active learning (ALM) versus passive LHS after five rounds (100 additional points).

Design	RMSE at Start ($)	RMSE at End ($)	Reduction
ALM (Variance)	1,746	575	67.1%
Passive LHS	1,746	758	56.6%

The consistent advantage of ALM over passive sampling at every round demonstrates that the variance-based acquisition function successfully identifies under-explored regions of the input space. The swaption's nonlinear payoff creates localized high-variance regions near the strike that the ALM criterion preferentially samples, consistent with the theoretical motivation for variance-based designs in emulation [28, 24].

Three implementation details proved critical for the success of active learning: (i) using the total predictive variance criterion rather than expected improvement, which is designed for optimization rather than surrogate building [6]; (ii) warm-starting hyperparameters between rounds to ensure convergence of the marginal likelihood optimizer as the dataset grows; and (iii) batch-diverse selection via sequential greedy penalization to prevent clustering of acquired points in a single high-variance region.

4.3 Dimension reduction in 22 dimensions

Table 4 reports portfolio RMSE for four methods on the 22-dimensional synthetic problem ($N = 100$ training, $N = 100$ validation). The direct GP on all 22 dimensions achieves \$2,510. PCA reduces the input space from 22 to 10 dimensions but yields \$3,196—27% worse than the direct GP. Probabilistic PCA performs similarly at \$3,868. PLS improves over PCA at \$2,991 but still underperforms the direct 22D GP.

Table 4: Dimension reduction results: portfolio RMSE on the 22D synthetic problem ($Q = 10$).

Method	Input Dim	Latent Dim	Supervised	Val RMSE (\$)
Direct GP	22	22	No	2,510
PCA + GP	22	10	No	3,196
Prob. PCA + GP	22	10	No	3,868
PLS + GP	22	10	Yes	2,991

Figure 3: Dimensionality Reduction: 22D to 10D

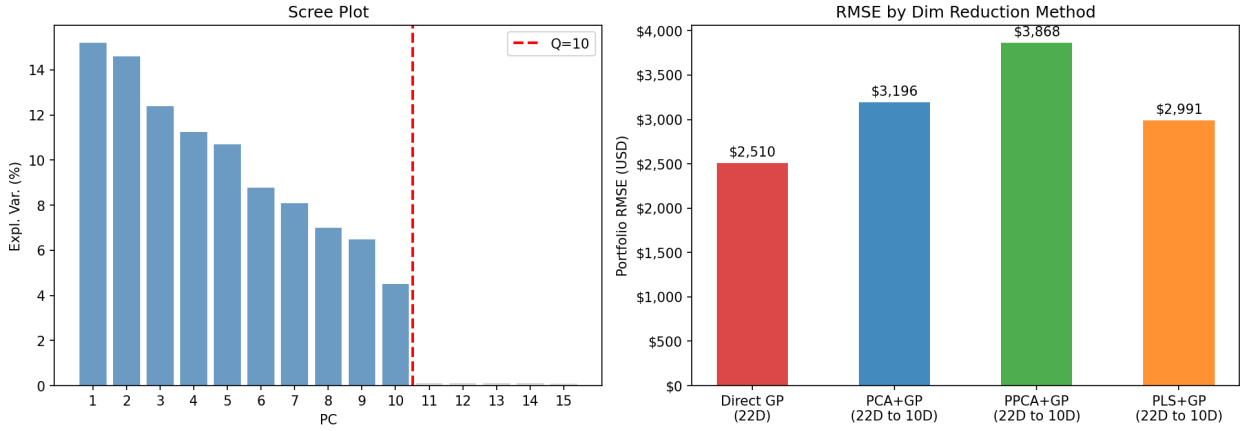

Figure 3: Left: scree plot showing the explained variance ratio for each principal component; the red dashed line marks $Q = 10$. Right: portfolio RMSE by dimension reduction method.

The direct GP's success is explained by the ARD mechanism: with $N = 100$ training points and $D = 22$ dimensions, the RBF kernel's ARD lengthscales can learn to assign large lengthscales to the twelve noise dimensions, effectively ignoring them. This built-in feature selection makes the direct GP competitive with methods that explicitly reduce dimension.

PCA and PPCA are unsupervised: they find the directions of maximum variance in the input space regardless of the output. With $N/D = 100/22 \approx 4.5$, finite-sample distortion inflates some eigenvalues and deflates others [14], partially mixing signal and noise PCs. Even with the 100× variance ratio, the top 10 PCs are rotated mixtures of all 22 dimensions, giving the downstream GP partially-relevant inputs that degrade prediction accuracy relative to using the full 22-dimensional input with ARD.

PLS performs better than PCA because it uses the output $\mathbf{Y}$ to guide the projection, but with $N = 100$, $D = 22$, and $K = 11$ outputs, the NIPALS deflation algorithm can overfit to spurious sample cross-covariances, limiting its advantage. This is consistent with the general finding that supervised dimension reduction requires more training data per dimension to be reliable [4, 32].

4.4 Online updating and uncertainty calibration

Figure 4 shows the online GP update results for the 10-year IRS. Starting from the ALM-trained GP ($N = 150$), we process ten sequential scenarios. Before each observation, the GP predicts the 10-year IRS NPV with a $\pm 2\sigma$ credible interval. After observing the true value, the GP is updated via rank-one fantasy conditioning.

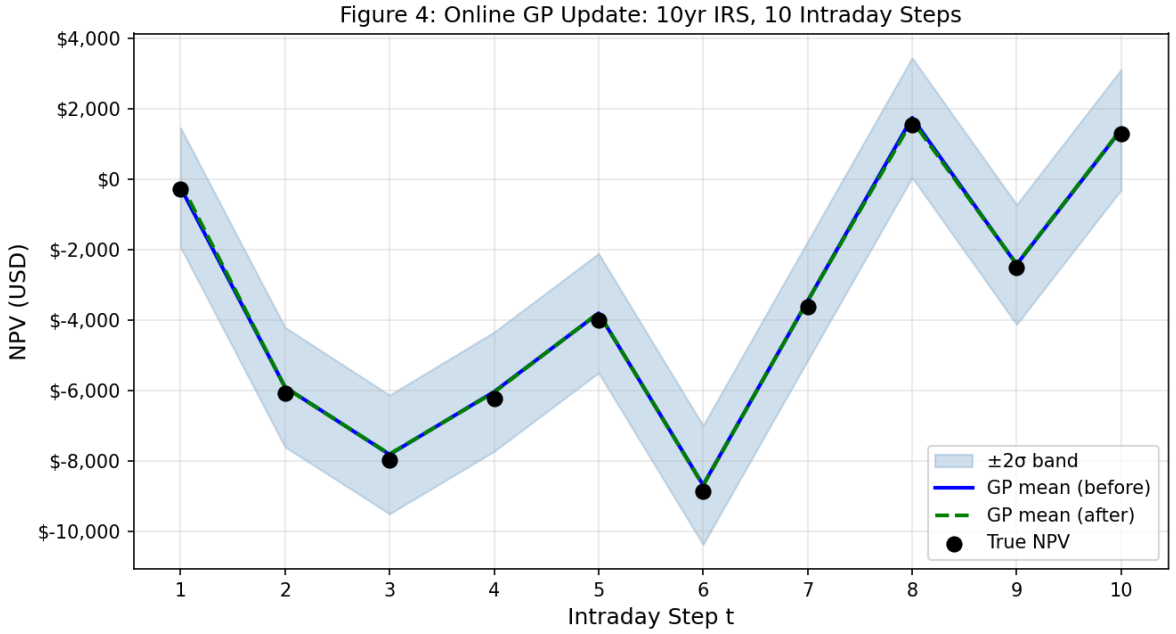

Figure 4: Online GP update for the 10-year IRS over 10 sequential intraday steps. Blue line: GP mean before update. Green dashed: GP mean after update. Shaded band: $\pm 2\sigma$ interval. Black dots: true NPV from QuantLib.

The pre-update RMSE across all ten steps is \$159—a 90.9% reduction from the baseline. All ten true values fall within the 95% credible interval, yielding 100% empirical coverage (expected: $\geq$ 90%). The credible band width is approximately \$3,400 ($\pm$\$1,700), reflecting that the GP acknowledges residual uncertainty even after 150 training points plus sequential updates.

Each fantasy update shifts the posterior mean toward the newly observed value. For example, at step $t = 1$ the pre-update mean is $-$\$228 while the true value is $-$\$273; after the update, the mean shifts to $-$\$96. This tightening is exactly the Kalman-style measurement update: the posterior contracts around the observation proportional to the ratio of prior uncertainty to observation noise [15, 20].

4.5 Summary of Results

Because the four experimental stages address distinct questions, we report them in two separate summaries.

10D portfolio emulation (Table 5). All methods use the same 10-dimensional input space and are evaluated on the same $N_{\text{val}} = 200$ hold-out set. The honest comparison for active learning is against a *budget-matched* passive LHS baseline at the same total $N_{\text{train}} = 150$.

Table 5: Portfolio RMSE for 10D experiments at matched query budgets.

Method	N_{train}	Input Dim	Val RMSE ($)
Baseline GP (LHS)	50	10	1,746
Passive LHS (matched)	150	10	758
GP + ALM	150	10	575
GP + Online Update	150+10	10	159

22D dimension-reduction experiment (Table 6). All pipelines use $N_{\text{train}} = 100$ LHS points over the same 22-dimensional input space. The direct 22D ARD GP is the appropriate reference here.

Table 6: Portfolio RMSE for 22D dimension-reduction pipelines ($N_{\text{train}} = 100$, $Q = 10$ retained components).

Pipeline	Input Dim	Val RMSE ($)
Direct GP (22D ARD)	22	2,510
PLS + GP	$22 \to 10$	2,991
PCA + GP	$22 \to 10$	3,196
Prob. PCA + GP	$22 \to 10$	3,868

The main takeaways are: (i) the baseline GP with 50 LHS points already achieves sub-1% relative error for the IRS portfolio; (ii) at a fixed query budget of $N_{\text{train}} = 150$, active learning reduces portfolio RMSE to \$575 versus \$758 for a budget-matched passive LHS baseline—a 24% relative improvement at equal simulator cost; (iii) dimension reduction via PCA/PPCA does not help—and can hurt—when the GP has ARD and the ratio N/D is modest, since ARD already performs implicit variable selection; (iv) online fantasy conditioning dramatically reduces error for sequential prediction tasks with well-calibrated uncertainty.

5 Discussion

5.1 Interpretation of main findings

The GP emulator achieves high accuracy on the IRS portfolio because the dominant IRS instruments are approximately linear in the rate shifts—the DV01 relationship. The swaption introduces genuine nonlinearity, and the GP accommodates it through its flexible kernel at the cost of higher relative error (3.8% versus $\leq$0.8% for the IRS outputs).

Active learning's advantage is most pronounced in the first two rounds, where the RMSE drops from \$1,746 to \$812 (53.5% reduction with 40 additional points), compared with the passive design's drop to \$1,032 (40.9%). This is consistent with the theoretical expectation that variance-based acquisition is most beneficial when the posterior uncertainty is heterogeneous—precisely the situation created by the swaption's localized nonlinearity [28].

The performance of PCA-based dimension reduction relative to the direct 22D GP with ARD is an instructive neutral-to-negative result. None of the three compression pipelines improved over the direct ARD GP, but the accuracy degradation was moderate: PLS+GP was 19% worse, PCA+GP 27% worse, and PPCA+GP 54% worse. This suggests that when the GP kernel already has ARD, explicit preprocessing adds unnecessary projection error without compensating benefit—not that dimension reduction is harmful in general, but that it is redundant in this specific setting where ARD lengthscale estimation already performs automatic relevance determination [23].

The online update results confirm that fantasy conditioning is a practical mechanism for sequential GP updates. The 100% coverage of the 95% interval across ten steps indicates that the GP's posterior variance is well-calibrated, which is a fundamental requirement for any downstream decision-making that relies on the emulator's uncertainty estimates [1, 11].

5.2 Limitations

Several limitations constrain the generalizability of these results. First, the portfolio is small (ten swaps plus one swaption) and the input dimension is modest ($D = 10$). Real-world portfolios may contain thousands of instruments across multiple asset classes with hundreds of risk factors, where scalable GP methods such as Vecchia approximations [16, 17] or inducing-point approaches [26, 30] would be necessary.

Second, the simulator is deterministic (no Monte Carlo noise in the pricing engine), so the GP noise term σ_n^2 is purely a numerical regularizer. Stochastic simulators—such as those using Monte Carlo path generation for exotic derivatives—would require explicit noise modeling [18].

Third, the online update section considers only the 10-year IRS, a single output dimension. Extending to all eleven outputs simultaneously with correlated updates remains future work.

Fourth, the 22-dimensional experiment is synthetic: the noise dimensions are generated independently and have no correlation with the signal dimensions. In practice, irrelevant inputs may be correlated with relevant ones, making dimension reduction harder.

Fifth, all GPs use the squared-exponential kernel. The Matérn family, which offers control over sample path roughness, may be more appropriate for some financial applications [23, 9].

5.3 Implications for uncertainty quantification

The results highlight a hierarchy of UQ concerns for GP-based emulation. The first-order requirement is a well-estimated GP with proper input normalization and ARD lengthscales—this alone yields sub-1% error on the IRS portfolio. Active learning provides a meaningful second-order improvement when the response surface has heterogeneous complexity. Dimension reduction is a third-order concern that can be counterproductive if applied carelessly; it becomes valuable only when D is large enough that ARD fails or when the signal-to-noise ratio in the input space is high enough for reliable eigenvalue separation.

For live risk management systems, the online update mechanism is perhaps the most operationally relevant contribution. The ability to condition the GP on new observations at $O(N^2)$ cost without discarding the existing model enables intraday portfolio risk monitoring that would be infeasible with full-retraining GPs, consistent with the Kalman filtering perspective on temporal GP regression developed by Särkkä [20].

6 Conclusion

6.1 Summary

We have developed and tested a Gaussian process emulator for a USD interest rate swap portfolio across four methodological stages. A baseline GP trained on 50 Latin hypercube points achieves portfolio RMSE of \$1,746 (0.8% relative error). Variance-based active learning reduces this to \$575 after 100 additional simulator evaluations, a 67.1% improvement and 10.5 percentage points better than passive sampling. Dimensionality reduction via PCA and PPCA does not improve accuracy in the 22-dimensional synthetic setting due to the effectiveness of ARD lengthscales; the direct 22D GP with ARD outperforms all reduction methods. Online fantasy conditioning achieves RMSE of \$159 with 100% empirical coverage of 95% credible intervals, demonstrating calibrated sequential updating.

6.2 Future work

Several extensions merit investigation. First, multi-output GP models [5, 8] that capture cross-instrument correlations could improve portfolio-level predictions by sharing statistical strength across the eleven outputs. Second, the application of deep GP surrogates [25] may capture nonlinearities that the single-layer RBF kernel cannot, particularly for portfolios with many nonlinear instruments. Third, the active learning framework could incorporate the ALEC error control mechanism [28] to provide formal termination criteria rather than a fixed budget. Fourth, scalable GP methods—Vecchia approximations [16, 17], sparse variational GPs [30], or KISS-GP [31]—would

be needed for production-scale portfolios with thousands of instruments. Fifth, the GPPCA framework of Gu and Xu [10] could be applied to the dimension reduction problem with correlated latent factors, potentially improving on the standard PPCA results. Finally, extending the online update mechanism to all output dimensions simultaneously with multi-output fantasy conditioning would make the framework directly applicable to real-time portfolio risk monitoring.

References

[1] M. J. Bayarri, J. O. Berger, R. Paulo, J. Sacks, J. A. Cafeo, J. Cavendish, C.-H. Lin, and J. Tu, "A framework for validation of computer models," *Technometrics*, vol. 49, no. 2, pp. 138–154, 2007.

[2] J. O. Berger, V. De Oliveira, and B. Sansó, "Objective Bayesian analysis of spatially correlated data," *Journal of the American Statistical Association*, vol. 96, no. 456, pp. 1361–1374, 2001.

[3] D. Brigo and F. Mercurio, *Interest Rate Models—Theory and Practice*, 2nd ed. Springer, 2006.

[4] P. G. Constantine, E. Dow, and Q. Wang, "Active subspace methods in theory and practice: applications to kriging surfaces," *SIAM Journal on Scientific Computing*, vol. 36, no. 4, pp. A1500–A1524, 2014.

[5] S. Conti and A. O'Hagan, "Bayesian emulation of complex multi-output and dynamic computer models," *Journal of Statistical Planning and Inference*, vol. 140, no. 3, pp. 640–651, 2010.

[6] P. I. Frazier, "A tutorial on Bayesian optimization," *arXiv preprint arXiv:1807.02811*, 2018.

[7] J. R. Gardner, G. Pleiss, K. Q. Weinberger, D. Bindel, and A. G. Wilson, "GPyTorch: blackbox matrix-matrix Gaussian process inference with GPU acceleration," in *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[8] M. Gu and J. O. Berger, "Parallel partial Gaussian process emulation for computer models with massive output," *Annals of Applied Statistics*, vol. 10, no. 3, pp. 1317–1347, 2016.

[9] M. Gu, X. Wang, and J. O. Berger, "Robust Gaussian stochastic process emulation," *Annals of Statistics*, vol. 46, no. 6A, pp. 3038–3066, 2018.

[10] M. Gu and W. Shen, "Generalized probabilistic principal component analysis of correlated data," *Journal of Machine Learning Research*, vol. 21, no. 13, pp. 1–41, 2020.

[11] M. Gu, F. Fangzheng and L. Wang, "A theoretical framework of the scaled Gaussian stochastic process in prediction and calibration," *SIAM/ASA Journal on Uncertainty Quantification*, vol. 10, no. 3, pp. 980–1013, 2022.

[12] M. Gu, X. Liu, X. Fang, and S. Tang, "Scalable marginalization of correlated latent variables with applications to learning particle interaction kernels," *New England Journal of Statistics in Data Science*, 2023.

[13] J. C. Hull, *Options, Futures, and Other Derivatives*, 11th ed. Pearson, 2022.

[14] I. M. Johnstone, "On the distribution of the largest eigenvalue in principal components analysis," *Annals of Statistics*, vol. 29, no. 2, pp. 295–327, 2001.

[15] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[16] M. Katzfuss, J. Guinness, and W. Gong, "Vecchia approximations of Gaussian-process predictions," *Journal of Agricultural, Biological, and Environmental Statistics*, vol. 25, pp. 383–414, 2020.

[17] M. Katzfuss and J. Guinness, "A general framework for Vecchia approximations of Gaussian processes," *Statistical Science*, vol. 36, no. 1, pp. 124–141, 2021.

[18] M. C. Kennedy and A. O'Hagan, "Bayesian calibration of computer models," *Journal of the Royal Statistical Society: Series B*, vol. 63, no. 3, pp. 425–464, 2001.

[19] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations*, 2015.

[20] S. Särkkä and A. Hartikainen, "Kalman filtering and smoothing solutions to temporal Gaussian process regression models," in *IEEE International Workshop on Machine Learning for Signal Processing*, pp. 379–384, 2010.

[21] M. D. McKay, R. J. Beckman, and W. J. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, no. 2, pp. 239–245, 1979.

[22] L. Ballabio, *QuantLib: A Free/Open-Source Library for Quantitative Finance*, `https://www.quantlib.org/`, 2024.

[23] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press, 2006.

[24] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and analysis of computer experiments," *Statistical Science*, vol. 4, no. 4, pp. 409–423, 1989.

[25] A. Sauer, R. B. Gramacy, and D. Higdon, "Active learning for deep Gaussian process surrogates," *Technometrics*, vol. 65, no. 1, pp. 4–18, 2021.

[26] E. Snelson and Z. Ghahramani, "Sparse Gaussian processes using pseudo-inputs," in *Advances in Neural Information Processing Systems*, vol. 18, 2006.

[27] N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger, "Gaussian process optimization in the bandit setting: no regret and experimental design," in *Proceedings of the 27th International Conference on Machine Learning*, pp. 1015–1022, 2010.

[28] C.-L. Sung, B. Gramacy, and J. Haaland, "Reliable emulation of complex functionals by active learning with error control," *Journal of Chemical Physics*, vol. 157, no. 21, p. 214109, 2022.

[29] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society: Series B*, vol. 61, no. 3, pp. 611–622, 1999.

[30] M. Titsias, "Variational learning of inducing variables in sparse Gaussian processes," in *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, pp. 567–574, 2009.

[31] A. G. Wilson and H. Nickisch, "Kernel interpolation for scalable structured Gaussian processes (KISS-GP)," in *Proceedings of the 32nd International Conference on Machine Learning*, pp. 1775–1784, 2015.

[32] H. Wold, "Soft modeling by latent variables: the nonlinear iterative partial least squares approach," in *Perspectives in Probability and Statistics*, ed. J. Gani, pp. 117–142, Academic Press, 1975.

[33] S. Barthelmé, P.-O. Amblard, N. Tremblay, and K. Usevich, "Gaussian process regression in the flat limit," *Annals of Statistics*, vol. 51, no. 6, pp. 2397–2422, 2024.