A Fully Bayesian Framework for Built-in Input Dimension Reduction and Gaussian Process Modeling

Eric Herrison Gyamfi
Joint work with Emily L. Kang and Alex Konomi

University of Cincinnati

Joint Statistical Meetings
Aug 03, 2025

Outline of Presentation

- Introduction
- Methodology
- Numerical Results
- Conclusion and Discussion

Introduction: Motivation

- **Gaussian Processes (GPs)** model complex systems due to **predictions** with **uncertainty quantification** (UQ).

Introduction: Motivation

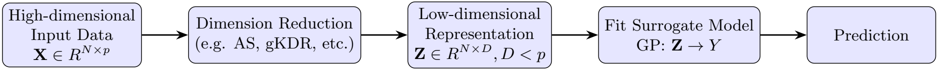
- **Gaussian Processes (GPs)** model complex systems due to **predictions** with **uncertainty quantification** (UQ).
- **Limitation:** GPs scale poorly in high dimensions – accuracy drops and computation becomes costly (**curse of dimensionality**).

Introduction: Motivation

- **Gaussian Processes (GPs)** model complex systems due to **predictions** with **uncertainty quantification** (UQ).
- **Limitation:** GPs scale poorly in high dimensions – accuracy drops and computation becomes costly (**curse of dimensionality**).
- **Conventional approach:** Dimensionality reduction $\rightarrow$ GP modeling (two-stage pipeline) [1, 4, 5, 9, 11, 13, 17].

Introduction: Motivation

- **Gaussian Processes (GPs)** model complex systems due to **predictions** with **uncertainty quantification** (UQ).
- **Limitation:** GPs scale poorly in high dimensions – accuracy drops and computation becomes costly (**curse of dimensionality**).
- **Conventional approach:** Dimensionality reduction $\rightarrow$ GP modeling (two-stage pipeline) [1, 4, 5, 9, 11, 13, 17].



Introduction: Motivation

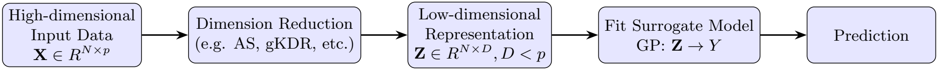
- **Gaussian Processes (GPs)** model complex systems due to **predictions** with **uncertainty quantification** (UQ).
- **Limitation:** GPs scale poorly in high dimensions – accuracy drops and computation becomes costly (**curse of dimensionality**).
- **Conventional approach:** Dimensionality reduction $\rightarrow$ GP modeling (two-stage pipeline) [1, 4, 5, 9, 11, 13, 17].

High-dimensional Input Data $\mathbf{X} \in R^{N \times p}$ → Dimension Reduction (e.g. AS, gKDR, etc.) → Low-dimensional Representation $\mathbf{Z} \in R^{N \times D}, D < p$ → Fit Surrogate Model GP: $\mathbf{Z} \rightarrow Y$ → Prediction

- **Recent works:** Emerging gradient-free methods exist but often lack a *fully Bayesian* framework [6, 14].

Introduction: Motivation

- **Gaussian Processes (GPs)** model complex systems due to **predictions** with **uncertainty quantification** (UQ).
- **Limitation:** GPs scale poorly in high dimensions – accuracy drops and computation becomes costly (**curse of dimensionality**).
- **Conventional approach:** Dimensionality reduction $\rightarrow$ GP modeling (two-stage pipeline) [1, 4, 5, 9, 11, 13, 17].



- **Recent works:** Emerging gradient-free methods exist but often lack a *fully Bayesian* framework [6, 14].
- **Our contribution:** A unified, fully Bayesian GP with integrated dimension reduction.

Key Contributions

- Proposes a fully Bayesian framework that unifies dimensionality reduction and Gaussian process modeling.

Key Contributions

- Proposes a fully Bayesian framework that unifies dimensionality reduction and Gaussian process modeling.
- Enforces orthonormal projection matrices via prior on the Stiefel manifold and HMC with geodesic flows.

Key Contributions

- Proposes a fully Bayesian framework that unifies dimensionality reduction and Gaussian process modeling.
- Enforces orthonormal projection matrices via prior on the Stiefel manifold and HMC with geodesic flows.
- Extends the model to Deep GP (DGP) for handling complex, high-dimensional inputs.

Methodology: GP with Built-in Dimension Reduction

- Let $\mathbf{x} \in \mathbb{R}^p$ be $p-$ high-dimensional inputs and $y = f(\mathbf{x}) : \mathcal{R}^p \to \mathcal{R}$ the response.

Methodology: GP with Built-in Dimension Reduction

- Let $\mathbf{x} \in \mathbb{R}^p$ be $p-$ high-dimensional inputs and $y = f(\mathbf{x}) : \mathcal{R}^p \rightarrow \mathcal{R}$ the response.
- Let $\mathbf{z} = W^T\mathbf{x} \in \mathbb{R}^D$ be $D-$ low-dimensional inputs via projection matrix, $W \in \mathbb{R}^{p\times D}$ and $g(\mathbf{z}) : \mathcal{R}^D \rightarrow \mathcal{R}$ link function.
- Assume $\mathbf{W}$ defined on Stiefel manifold, $\mathcal{V}_{p,D}$

$$\mathcal{V}_{p,D} = \{W \in \mathcal{R}^{p\times D} : W^TW = \mathcal{I}_D\}, \quad \mathcal{I}_D = \text{identity matrix}$$

Methodology: GP with Built-in Dimension Reduction

- Let $\mathbf{x} \in \mathbb{R}^p$ be $p-$ high-dimensional inputs and $y = f(\mathbf{x}) : \mathcal{R}^p \to \mathcal{R}$ the response.
- Let $\mathbf{z} = W^T\mathbf{x} \in \mathbb{R}^D$ be $D-$ low-dimensional inputs via projection matrix, $W \in \mathbb{R}^{p\times D}$ and $g(\mathbf{z}) : \mathcal{R}^D \to \mathcal{R}$ link function.
- Assume $\mathbf{W}$ defined on Stiefel manifold, $\mathcal{V}_{p,D}$

$$\mathcal{V}_{p,D} = \{W \in \mathcal{R}^{p\times D} : W^TW = \mathcal{I}_D\}, \quad \mathcal{I}_D = \text{identity matrix}$$

- Goal: replace costly $f(\mathbf{x})$ with $g(\mathbf{z})$, assuming $f(\mathbf{x}) \approx g(\mathbf{z})$.
- W maps $\mathbb{R}^p \to \mathbb{R}^D$ where $D < p$, enabling dimension reduction (DR) in GP modeling.

Methodology: GP with Built-in Dimension Reduction

- Let $\mathbf{x} \in \mathbb{R}^p$ be $p-$ high-dimensional inputs and $y = f(\mathbf{x}) : \mathcal{R}^p \to \mathcal{R}$ the response.
- Let $\mathbf{z} = W^T\mathbf{x} \in \mathbb{R}^D$ be $D-$ low-dimensional inputs via projection matrix, $W \in \mathbb{R}^{p\times D}$ and $g(\mathbf{z}) : \mathcal{R}^D \to \mathcal{R}$ link function.
- Assume $\mathbf{W}$ defined on Stiefel manifold, $\mathcal{V}_{p,D}$

$$\mathcal{V}_{p,D} = \{W \in \mathcal{R}^{p\times D} : W^T W = \mathcal{I}_D\}, \quad \mathcal{I}_D = \text{identity matrix}$$

- Goal: replace costly $f(\mathbf{x})$ with $g(\mathbf{z})$, assuming $f(\mathbf{x}) \approx g(\mathbf{z})$.
- W maps $\mathbb{R}^p \to \mathbb{R}^D$ where $D < p$, enabling dimension reduction (DR) in GP modeling.
- For n input points and $Y = (y_1, \cdots, y_n)^T \in \mathbb{R}^n$, GP model is defined on $Z = (\mathbf{z_1}, \cdots, \mathbf{z_n})^T \in \mathbb{R}^{n\times D}$:

$$Y \sim \text{GP}(\mu_Y, \Sigma(Z)), \quad \Sigma(Z) = \tau^2[C(Z; \theta_D, W) + g]$$

- τ^2 is process variance, g is nugget, $C(Z; \theta_D, W)$ is an isotropic kernel with lengthscale θ_D and $\mu_Y = 0$.

Why Deep Gaussian Processes (DGPs)?

Limitations of Standard GPs:

- Assume uniform statistical behavior across input space(stationarity).
- Struggle with input-output relationship changing patterns .

Why Deep Gaussian Processes (DGPs)?

Limitations of Standard GPs:

- Assume uniform statistical behavior across input space(stationarity).
- Struggle with input-output relationship changing patterns .

Benefits of DGPs:

Why Deep Gaussian Processes (DGPs)?

Limitations of Standard GPs:

- Assume uniform statistical behavior across input space(stationarity).
- Struggle with input-output relationship changing patterns .

Benefits of DGPs:

- Adapt to changing relationships across input regions(non-stationary).
- Stack multiple GPs layers to:

Why Deep Gaussian Processes (DGPs)?

Limitations of Standard GPs:

- Assume uniform statistical behavior across input space(stationarity).
- Struggle with input-output relationship changing patterns .

Benefits of DGPs:

- Adapt to changing relationships across input regions(non-stationary).
- Stack multiple GPs layers to:
    - Model varying smoothness.

Why Deep Gaussian Processes (DGPs)?

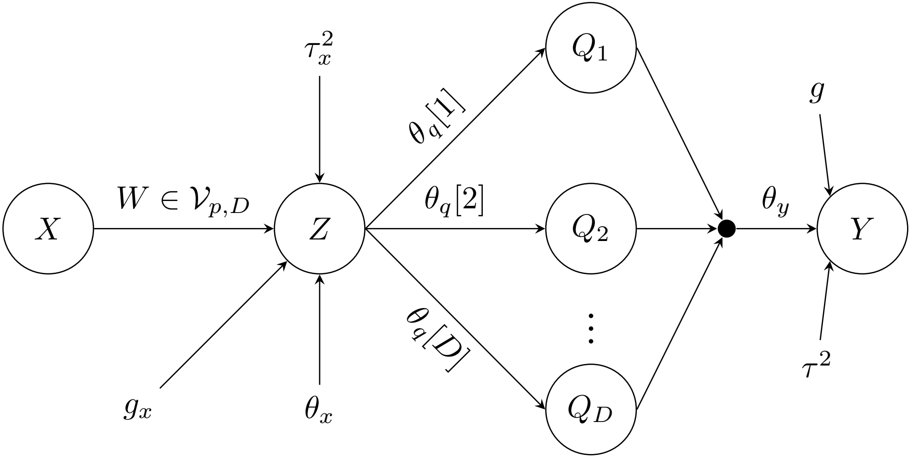
Limitations of Standard GPs:

- Assume uniform statistical behavior across input space(stationarity).
- Struggle with input-output relationship changing patterns .

Benefits of DGPs:

- Adapt to changing relationships across input regions(non-stationary).
- Stack multiple GPs layers to:
  - Model varying smoothness.
- Maintain GP strengths:
  - Accurate predictions.
  - Reliable uncertainty estimates.

Two-layer DGP Model with built-in Dimension Reduction



Two-layer DGP Model with built-in Dimension Reduction

$$Q_j \overset{\text{ind}}{\sim} \mathcal{N}(0, C_{\theta_Q[j],W}(Z)), \quad Q_j \in \mathbb{R}^n, \quad W \in \mathcal{V}_{p,D}, \quad j = 1, \ldots, D.$$

$$\theta_Q = (\theta_Q[1], \cdots, \theta_D[D]) \in \mathbb{R}^D, \quad Q = [Q_1, \cdots, Q_D] \in \mathbb{R}^{n \times D}$$

$$Y \mid Q \sim \mathcal{N}(0, \tau^2[C_{\theta_y}(Q) + g\mathcal{I}_n])$$

$$\mathcal{L}(Y \mid X) \propto \int \mathcal{L}(Y \mid Q)\mathcal{L}(Q \mid Z)\, dW\, dQ$$

Two-layer DGP Model with built-in Dimension Reduction

$$Q_j \stackrel{\text{ind}}{\sim} \mathcal{N}(0, C_{\theta_Q[j],W}(Z)), \quad Q_j \in \mathbb{R}^n, \quad W \in \mathcal{V}_{p,D}, \quad j = 1, \ldots, D.$$

$$\theta_Q = (\theta_Q[1], \cdots, \theta_D[D]) \in \mathbb{R}^D, \quad Q = [Q_1, \cdots, Q_D] \in \mathbb{R}^{n \times D}$$

$$Y \mid Q \sim \mathcal{N}(0, \tau^2[C_{\theta_y}(Q) + g\mathcal{I}_n])$$

$$\mathcal{L}(Y \mid X) \propto \int \mathcal{L}(Y \mid Q)\mathcal{L}(Q \mid Z)\, dW\, dQ$$

- Q: Latent variable and $Q_1, \cdots, Q_D$ be the latent nodes
- $C_\theta(\cdot)$: Covariance function with parameters θ.
- θ_Q, θ_y: Covariance hyperparameters for latent/output layers.
- g, g_x: Nugget terms (noise parameters).
- τ^2, τ_x^2: Variance scale parameters.

Projection Matrix (W) Prior

$W \sim \mathcal{ML}(W; F), W \in \mathcal{V}_{p,D}$ - Matrix Langevin $(\mathcal{ML})$ prior

Projection Matrix (W) Prior

$W \sim \mathcal{ML}(W; F), W \in \mathcal{V}_{p,D}$ - Matrix Langevin ($\mathcal{ML}$) prior

- $\mathcal{ML}-$distribution [3, 8] with respect to Haar measure μ on $\mathcal{V}_{p,D}$:

$$\pi_{\mathcal{ML}}(W; F) = \frac{1}{c(F)} \exp(\mathsf{tr}(F^T W)), \quad c(F) = {}_0\mathcal{F}_1 \left(\frac{d}{2}, \frac{F^T F}{4} \right)$$

Projection Matrix (W) Prior

$$W \sim \mathcal{ML}(W; F), W \in \mathcal{V}_{p,D}$$ - Matrix Langevin ($\mathcal{ML}$) prior

- $\mathcal{ML}-$distribution [3, 8] with respect to Haar measure μ on $\mathcal{V}_{p,D}$:

$$\pi_{\mathcal{ML}}(W; F) = \frac{1}{c(F)} \exp(\text{tr}(F^T W)), \quad c(F) = {}_0\mathcal{F}_1\left(\frac{d}{2}, \frac{F^T F}{4}\right)$$

- ${}_0\mathcal{F}_1\left(\frac{d}{2}, \frac{F^T F}{4}\right)$ is hypergeometric function of order $\frac{d}{2}$ with matrix $F^T F/4$ [2, 3, 10].

Projection Matrix (W) Prior

$$W \sim \mathcal{ML}(W; F), W \in \mathcal{V}_{p,D}$$ - Matrix Langevin ($\mathcal{ML}$) prior

- $\mathcal{ML}-$distribution [3, 8] with respect to Haar measure μ on $\mathcal{V}_{p,D}$:

$$\pi_{\mathcal{ML}}(W; F) = \frac{1}{c(F)} \exp(\mathrm{tr}(F^T W)), \quad c(F) = {}_0\mathcal{F}_1\left(\frac{d}{2}, \frac{F^T F}{4}\right)$$

- ${}_0\mathcal{F}_1\left(\frac{d}{2}, \frac{F^T F}{4}\right)$ is hypergeometric function of order $\frac{d}{2}$ with matrix $F^T F/4$ [2, 3, 10].

Parameterization of F via SVD: (1.5.8) in [3]

$$F = M \Lambda V^T, \quad \Lambda = \mathrm{diag}(\{\lambda_1, \cdots, \lambda_D\})$$

- $V \in \mathcal{V}_{D,D} = \mathcal{O}(D)$ is space of orthogonal matrices of dimension $D \times D$
- $M \in \tilde{\mathcal{V}}_{p,D} = \{W \in \mathcal{V}_{p,D} : W_{1,j} \geq 0, \forall j = 1, 2, \cdots, D\}$
- $\lambda = (\lambda_1, \cdots, \lambda_D) \in \mathcal{S}_D = \{\lambda \in \mathcal{R}_+^D : \infty > \lambda_1 > \cdots > \lambda_D > 0\}$

- $V \in \mathcal{V}_{D,D} = \mathcal{O}(D)$ is space of orthogonal matrices of dimension $D \times D$
- $M \in \tilde{\mathcal{V}}_{p,D} = \{W \in \mathcal{V}_{p,D} : W_{1,j} \geq 0, \forall j = 1, 2, \cdots, D\}$
- $\lambda = (\lambda_1, \cdots, \lambda_D) \in \mathcal{S}_D = \{\lambda \in \mathcal{R}_+^D : \infty > \lambda_1 > \cdots > \lambda_D > 0\}$

$$\pi_{\mathcal{ML}}(W; (V, M, \lambda)) := \frac{1}{c(\Lambda)} exp(tr(V \Lambda M^T W)) \mathcal{I}(W \in \mathcal{V}_{p,D})$$

$$c(\Lambda) = {}_0\mathcal{F}_1(\frac{d}{2}, \frac{\Lambda^2}{4}), V \in \mathcal{V}_{D,D}, M \in \tilde{\mathcal{V}}_{p,D}, \lambda \in \mathcal{S}_D$$

- $V \in \mathcal{V}_{D,D} = \mathcal{O}(D)$ is space of orthogonal matrices of dimension $D \times D$
- $M \in \tilde{\mathcal{V}}_{p,D} = \{W \in \mathcal{V}_{p,D} : W_{1,j} \geq 0, \forall j = 1, 2, \cdots, D\}$
- $\lambda = (\lambda_1, \cdots, \lambda_D) \in \mathcal{S}_D = \{\lambda \in \mathcal{R}_+^D : \infty > \lambda_1 > \cdots > \lambda_D > 0\}$

$$\pi_{\mathcal{ML}}(W; (V, M, \lambda)) := \frac{1}{c(\Lambda)} exp(tr(V \Lambda M^T W)) \mathcal{I}(W \in \mathcal{V}_{p,D})$$

$$c(\Lambda) = {}_0\mathcal{F}_1(\frac{d}{2}, \frac{\Lambda^2}{4}), V \in \mathcal{V}_{D,D}, M \in \tilde{\mathcal{V}}_{p,D}, \lambda \in \mathcal{S}_D$$

Prior for M, V and λ

$$M \sim \mathcal{ML}(F_M), \quad V \sim \mathcal{ML}(F_V)$$

$$\lambda_k \sim^{i.i.d} \Gamma(b_1, b_2), \forall k = 1, \ldots, D$$

Hyperparameter Priors

- Variance scale parameter (τ^2):

$$\tau^2 \sim \mathsf{IG}(\alpha_1, \alpha_2)$$

Hyperparameter Priors

- Variance scale parameter (τ^2):

$$\tau^2 \sim \mathsf{IG}(\alpha_1, \alpha_2)$$

- Nugget (g) and covariance parameters θ

$$\{g, \theta\} \sim^{iid} \Gamma(3/2, b_{[.]})$$

Hyperparameter Priors

- Variance scale parameter (τ^2):

$$\tau^2 \sim \mathsf{IG}(\alpha_1, \alpha_2)$$

- Nugget (g) and covariance parameters θ

$$\{g, \theta\} \sim^{iid} \Gamma(3/2, b_{[.]})$$

- $b_{[\theta_y]} > b_{[\theta_Q]} > b_{[\theta_x]}$: Controls smoothness hierarchy
  - Output layer smoother than latent, which is smoother than input.
  - encoding a prior belief that as layers get deeper they should be less "wiggly"

Hyperparameter Priors

- Variance scale parameter (τ^2):

$$\tau^2 \sim \mathsf{IG}(\alpha_1, \alpha_2)$$

- Nugget (g) and covariance parameters θ

$$\{g, \theta\} \stackrel{iid}{\sim} \Gamma(3/2, b_{[.]})$$

- $b_{[\theta_y]} > b_{[\theta_Q]} > b_{[\theta_x]}$: Controls smoothness hierarchy
  - Output layer smoother than latent, which is smoother than input.
  - encoding a prior belief that as layers get deeper they should be less "wiggly"

Latent Layers:

- All latent layers follow a zero-mean Multivariate Normal (MVN) prior.

Posterior Inference Summary

- Perform fully Bayesian inference for DGP models via MCMC.

Posterior Inference Summary

- Perform fully Bayesian inference for DGP models via MCMC.
- Posterior inference for W, Q and λ is intractable.

Posterior Inference Summary

- Perform fully Bayesian inference for DGP models via MCMC.
- Posterior inference for W, Q and λ is intractable.
- Hybrid MCMC framework prioritizing UQ:

Posterior Inference Summary

- Perform fully Bayesian inference for DGP models via MCMC.
- Posterior inference for W, Q and λ is intractable.
- Hybrid MCMC framework prioritizing UQ:
    - Metropolis-Hastings (MH) for $\{g_x, \theta_x, \theta_Q, g, \theta_y\}$ [7].

Posterior Inference Summary

- Perform fully Bayesian inference for DGP models via MCMC.
- Posterior inference for W, Q and λ is intractable.
- Hybrid MCMC framework prioritizing UQ:
    - Metropolis-Hastings (MH) for $\{g_x, \theta_x, \theta_Q, g, \theta_y\}$ [7].
    - Hamiltonian Monte Carlo (HMC) for W [15, 16]

Posterior Inference Summary

- Perform fully Bayesian inference for DGP models via MCMC.
- Posterior inference for W, Q and λ is intractable.
- Hybrid MCMC framework prioritizing UQ:
  - Metropolis-Hastings (MH) for $\{g_x, \theta_x, \theta_Q, g, \theta_y\}$ [7].
  - Hamiltonian Monte Carlo (HMC) for W [15, 16]
  - Elliptical Slice Sampling (ESS) for $Q_1, \cdots, Q_D$ and λ - requires no tuning as recently employed in [12].

Numerical Experiment 1: Synthetic Data

Data Generation: Employed in [14]

Numerical Experiment 1: Synthetic Data

Data Generation: Employed in [14]

- Inputs $\mathbf{x} \sim \mathcal{N}_d(0, \mathbf{I})$; $d = 10$.

Numerical Experiment 1: Synthetic Data

Data Generation: Employed in [14]

- Inputs $\mathbf{x} \sim \mathcal{N}_d(0, \mathbf{I})$; $d = 10$.
- Output: $Y = \mathbf{a}_0 + \mathbf{a}^T \phi + \phi^T \mathbf{A} \phi + \epsilon$, $\phi = \mathbf{W}^\top \mathbf{x}$, $\epsilon \sim \mathcal{N}(0, 0.01)$.

Numerical Experiment 1: Synthetic Data

Data Generation: Employed in [14]

- Inputs $\mathbf{x} \sim \mathcal{N}_d(0, \mathbf{I})$; $d = 10$.
- Output: $Y = \mathbf{a}_0 + \mathbf{a}^T\phi + \phi^T\mathbf{A}\phi + \epsilon$, $\phi = \mathbf{W}^\top\mathbf{x}$, $\epsilon \sim \mathcal{N}(0, 0.01)$.

Scenario 1: 2D Input subspace

Numerical Experiment 1: Synthetic Data

Data Generation: Employed in [14]

- Inputs $\mathbf{x} \sim \mathcal{N}_d(0, \mathbf{I})$; $d = 10$.
- Output: $Y = \mathbf{a}_0 + \mathbf{a}^T\phi + \phi^T\mathbf{A}\phi + \epsilon$, $\phi = \mathbf{W}^\top\mathbf{x}$, $\epsilon \sim \mathcal{N}(0, 0.01)$.

Scenario 1: 2D Input subspace

$$\mathbf{W} = \begin{pmatrix} 0.008 & -0.184 & 0.343 & -0.053 & 0.081 & 0.066 & -0.412 & 0.654 & 0.485 & 0.040 \\ 0.067 & -0.415 & 0.482 & 0.076 & 0.210 & 0.538 & 0.078 & -0.200 & -0.291 & 0.348 \end{pmatrix}^T$$

$$\mathbf{a}_0 = -0.06976, \quad \mathbf{a} = \big(0.4376, 0.9870\big)^T, \quad \mathbf{A} = \begin{pmatrix} -0.9257 & -0.3840 \\ -0.4174 & -0.6766 \end{pmatrix}$$

Numerical Experiment 1: Synthetic Data

Data Generation: Employed in [14]

- Inputs $\mathbf{x} \sim \mathcal{N}_d(0, \mathbf{I})$; $d = 10$.
- Output: $Y = \mathbf{a}_0 + \mathbf{a}^T\phi + \phi^T\mathbf{A}\phi + \epsilon$, $\phi = \mathbf{W}^\top\mathbf{x}$, $\epsilon \sim \mathcal{N}(0, 0.01)$.

Scenario 1: 2D Input subspace

$$\mathbf{W} = \begin{pmatrix} 0.008 & -0.184 & 0.343 & -0.053 & 0.081 & 0.066 & -0.412 & 0.654 & 0.485 & 0.040 \\ 0.067 & -0.415 & 0.482 & 0.076 & 0.210 & 0.538 & 0.078 & -0.200 & -0.291 & 0.348 \end{pmatrix}^T$$

$$\mathbf{a}_0 = -0.06976, \quad \mathbf{a} = \begin{pmatrix} 0.4376, 0.9870 \end{pmatrix}^T, \quad \mathbf{A} = \begin{pmatrix} -0.9257 & -0.3840 \\ -0.4174 & -0.6766 \end{pmatrix}$$

Experimental Setup:

Numerical Experiment 1: Synthetic Data

Data Generation: Employed in [14]

- Inputs $\mathbf{x} \sim \mathcal{N}_d(0, \mathbf{I})$; $d = 10$.
- Output: $Y = \mathbf{a}_0 + \mathbf{a}^T \phi + \phi^T \mathbf{A} \phi + \epsilon$, $\phi = \mathbf{W}^\top \mathbf{x}$, $\epsilon \sim \mathcal{N}(0, 0.01)$.

Scenario 1: 2D Input subspace

$$\mathbf{W} = \begin{pmatrix} 0.008 & -0.184 & 0.343 & -0.053 & 0.081 & 0.066 & -0.412 & 0.654 & 0.485 & 0.040 \\ 0.067 & -0.415 & 0.482 & 0.076 & 0.210 & 0.538 & 0.078 & -0.200 & -0.291 & 0.348 \end{pmatrix}^T$$

$$\mathbf{a}_0 = -0.06976, \quad \mathbf{a} = \left(0.4376, 0.9870\right)^T, \quad \mathbf{A} = \begin{pmatrix} -0.9257 & -0.3840 \\ -0.4174 & -0.6766 \end{pmatrix}$$

Experimental Setup:

- $n = 600$ samples; training set is 80%, test set 20%.

Numerical Experiment 1: Synthetic Data

Data Generation: Employed in [14]

- Inputs $\mathbf{x} \sim \mathcal{N}_d(0, \mathbf{I})$; $d = 10$.
- Output: $Y = \mathbf{a}_0 + \mathbf{a}^T\phi + \phi^T\mathbf{A}\phi + \epsilon$, $\phi = \mathbf{W}^\top\mathbf{x}$, $\epsilon \sim \mathcal{N}(0, 0.01)$.

Scenario 1: 2D Input subspace

$$\mathbf{W} = \begin{pmatrix} 0.008 & -0.184 & 0.343 & -0.053 & 0.081 & 0.066 & -0.412 & 0.654 & 0.485 & 0.040 \\ 0.067 & -0.415 & 0.482 & 0.076 & 0.210 & 0.538 & 0.078 & -0.200 & -0.291 & 0.348 \end{pmatrix}^T$$

$$\mathbf{a}_0 = -0.06976, \quad \mathbf{a} = \left(0.4376, 0.9870\right)^T, \quad \mathbf{A} = \begin{pmatrix} -0.9257 & -0.3840 \\ -0.4174 & -0.6766 \end{pmatrix}$$

Experimental Setup:

- $n = 600$ samples; training set is 80%, test set 20%.
- Baseline Methods:Active Subspace (AS), gradient kernel dimension reduction (gKDR), Gaussian process maximum likelihood estimate (GP-MLE)

Numerical Experiment 1: Synthetic Data

Data Generation: Employed in [14]

- Inputs $\mathbf{x} \sim \mathcal{N}_d(0, \mathbf{I})$; $d = 10$.
- Output: $Y = \mathbf{a}_0 + \mathbf{a}^T\phi + \phi^T\mathbf{A}\phi + \epsilon$, $\phi = \mathbf{W}^\top\mathbf{x}$, $\epsilon \sim \mathcal{N}(0, 0.01)$.

Scenario 1: 2D Input subspace

$$\mathbf{W} = \begin{pmatrix} 0.008 & -0.184 & 0.343 & -0.053 & 0.081 & 0.066 & -0.412 & 0.654 & 0.485 & 0.040 \\ 0.067 & -0.415 & 0.482 & 0.076 & 0.210 & 0.538 & 0.078 & -0.200 & -0.291 & 0.348 \end{pmatrix}^T$$

$$\mathbf{a}_0 = -0.06976, \quad \mathbf{a} = \left(0.4376, 0.9870\right)^T, \quad \mathbf{A} = \begin{pmatrix} -0.9257 & -0.3840 \\ -0.4174 & -0.6766 \end{pmatrix}$$

Experimental Setup:

- $n = 600$ samples; training set is 80%, test set 20%.
- Baseline Methods:Active Subspace (AS), gradient kernel dimension reduction (gKDR), Gaussian process maximum likelihood estimate (GP-MLE)
- Performance metrics: root mean square prediction error (RMSPE), Nash-Sutcliffe model efficiency coefficient (NSME), Continuous Ranked Probability Score (CRPS), Score, Bayesian Information Criterion (BIC) and mean log pointwise predicted density (MLPPD).

- Baseline methods:
  - **Active Subspace (AS):** Identifies dominant input directions via the second-moment matrix of simulator gradients $\nabla_x f(x)$ [4]. **Requires direct gradient access**, making it impractical for black-box simulators.

- Baseline methods:
  - **Active Subspace (AS):** Identifies dominant input directions via the second-moment matrix of simulator gradients $\nabla_x f(x)$ [4]. **Requires direct gradient access**, making it impractical for black-box simulators.

  - **gKDR:** Builds on the sufficient DR framework using gradients of the *input kernel*, not the simulator [5].

- Baseline methods:
  - **Active Subspace (AS):** Identifies dominant input directions via the second-moment matrix of simulator gradients $\nabla_x f(x)$ [4]. **Requires direct gradient access**, making it impractical for black-box simulators.
  - **gKDR:** Builds on the sufficient DR framework using gradients of the *input kernel*, not the simulator [5].
  - **GP-MLE:** Employs MLE to estimate W and the covariance hyperparameters [14].

- Baseline methods:
    - **Active Subspace (AS):** Identifies dominant input directions via the second-moment matrix of simulator gradients $\nabla_x f(x)$ [4]. **Requires direct gradient access**, making it impractical for black-box simulators.
    - **gKDR:** Builds on the sufficient DR framework using gradients of the *input kernel*, not the simulator [5].
    - **GP-MLE:** Employs MLE to estimate W and the covariance hyperparameters [14].
- DGP $\mathbf{A}$ - layer $(\mathbf{D})$ denotes DGP with $\mathbf{A}$ layer(s) and input subspace $\mathbf{D}$ where $\mathbf{A}, \mathbf{D} = 1, 2, 3$.

- Baseline methods:
  - **Active Subspace (AS):** Identifies dominant input directions via the second-moment matrix of simulator gradients $\nabla_x f(x)$ [4]. **Requires direct gradient access**, making it impractical for black-box simulators.
  - **gKDR:** Builds on the sufficient DR framework using gradients of the *input kernel*, not the simulator [5].
  - **GP-MLE:** Employs MLE to estimate W and the covariance hyperparameters [14].
- DGP $\mathbf{A}$ - layer $(\mathbf{D})$ denotes DGP with $\mathbf{A}$ layer(s) and input subspace $\mathbf{D}$ where $\mathbf{A}, \mathbf{D} = 1, 2, 3$.
- DGP $\mathbf{A}$ - layer $(\mathbf{D})$ W/o represents DGP with $\mathbf{A}$ layer(s) and input subspace $\mathbf{D}$ without DR.
- DGP $\mathbf{A}$ - layer $(\mathbf{D})$ Truth represents DGP with $\mathbf{A}$ layer(s) and input subspace $\mathbf{D}$ with DR but uses the true W.

Method (D)	RMSPE	NSME	CRPS	Score	BIC
AS (2)	0.2596	0.9717	0.7833	−232.3104	245.53
gKDR (2)	0.1849	0.9907	0.5637	281.7241	240.30
GP-MLE (2)	0.0946	0.9976	0.5347	449.8206	246.11
DGP 1-layer (1)	0.1055	0.9945	0.1023	260.5579	248.01
DGP 1-layer (2)	**0.0815**	**0.9982**	**0.0162**	**715.9930**	**249.20**
DGP 1-layer (3)	0.1272	0.9940	0.2499	712.2032	247.52
DGP 2-layer (1)	0.1437	0.9929	0.1046	643.7125	247.00
DGP 2-layer (2)	0.1100	0.9972	0.1442	266.6124	248.33
DGP 2-layer (3)	0.1592	0.9949	0.5855	540.6212	246.07
DGP 3-layer (1)	0.1937	0.9873	0.9335	206.9284	244.02
DGP 3-layer (2)	0.1821	0.9902	0.9890	146.8134	245.06
DGP 3-layer (3)	0.1941	0.9850	0.5073	694.1730	243.90
DGP 1-layer (10) W/o	0.2194	0.9821	0.2376	220.6803	240.00
DGP 2-layer (10) W/o	0.2175	0.9856	0.7510	317.6212	243.45
DGP 3-layer (10) W/o	0.2070	0.9702	0.6450	104.6729	243.98
DGP 1-layer (2) Truth	**0.0706**	**0.9993**	**0.0118**	**324.2333**	**250.08**
DGP 2-layer (2) Truth	0.0981	0.9941	0.1496	317.5155	246.59
DGP 3-layer (2) Truth	0.1629	0.9945	0.2646	331.7272	247.00

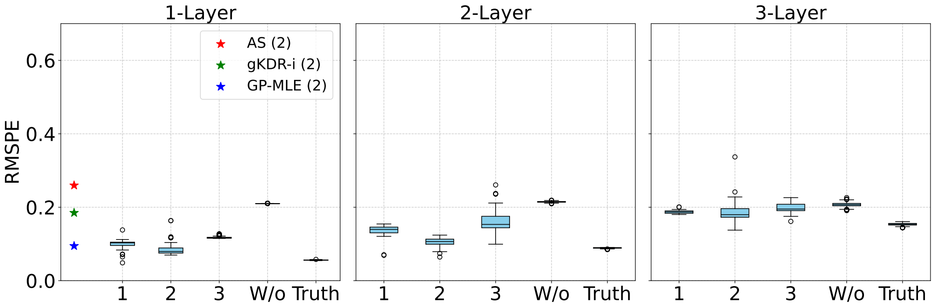


Figure: RMSPE comparisons across different method for train size 480

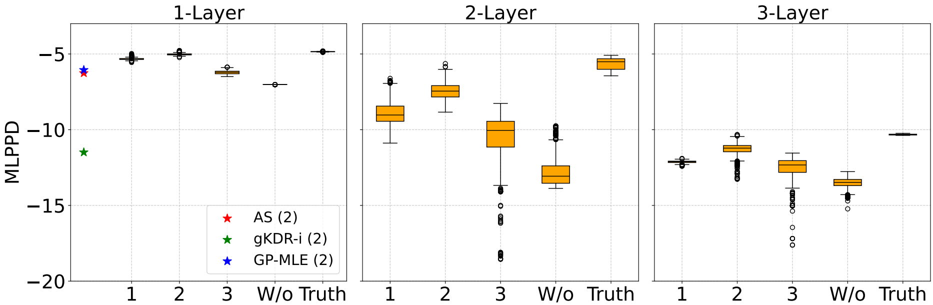


Figure: MLPPD comparisons across different method for train size 480

Numerical Experiment 2

2D Input Subspace: Employed in [12]

Numerical Experiment 2

2D Input Subspace: Employed in [12]

- Inputs $x \in [0, 1]^{10}$ from Latin Hyperpercube Sample

Numerical Experiment 2

2D Input Subspace: Employed in [12]

- Inputs $x \in [0, 1]^{10}$ from Latin Hyperpercube Sample
- Projected by known 10×2 matrix W.

Numerical Experiment 2

2D Input Subspace: Employed in [12]

- Inputs $x \in [0,1]^{10}$ from Latin Hyperpercube Sample
- Projected by known 10×2 matrix W.
- $z = (z_1, z_2) = W^\top x$ with response function given as

Numerical Experiment 2

2D Input Subspace: Employed in [12]

- Inputs $x \in [0,1]^{10}$ from Latin Hyperpercube Sample
- Projected by known 10×2 matrix W.
- $z = (z_1, z_2) = W^\top x$ with response function given as

$$f(z) = 10 z_1 \exp(-z_1^2 - z_2^2); z_j = (z_j - 0.5) \cdot 6 + 1, j = 1, 2.$$

Numerical Experiment 2

2D Input Subspace: Employed in [12]

- Inputs $x \in [0,1]^{10}$ from Latin Hyperpercube Sample
- Projected by known 10×2 matrix W.
- $z = (z_1, z_2) = W^\top x$ with response function given as

$$f(z) = 10 z_1 \exp(-z_1^2 - z_2^2);\, z_j = (z_j - 0.5) \cdot 6 + 1, j = 1, 2.$$

- W is the same as numerical experiment 1

Numerical Experiment 2

2D Input Subspace: Employed in [12]

- Inputs $x \in [0,1]^{10}$ from Latin Hyperpercube Sample
- Projected by known 10×2 matrix W.
- $z = (z_1, z_2) = W^\top x$ with response function given as

$$f(z) = 10 z_1 \exp(-z_1^2 - z_2^2); z_j = (z_j - 0.5) \cdot 6 + 1, j = 1, 2.$$

- W is the same as numerical experiment 1
- $n = 300$ samples; training set is 80%, test set 20%.

Method (D)	RMSPE	NSME	CRPS	Score	BIC
AS (2)	0.6190	0.8329	0.5420	90.2640	601.39
gKDR (2)	0.7791	0.7562	0.6429	88.7980	596.24
GP-MLE (2)	0.6052	0.8400	0.5113	91.0290	604.08
DGP 1-layer (1)	0.4795	0.9035	0.5938	104.6915	615.28
DGP 1-layer (2)	0.5302	0.8873	0.4549	63.3460	612.07
DGP 1-layer (3)	0.5948	0.7691	0.5863	72.6261	608.65
DGP 2-layer (1)	0.4778	0.9079	0.4097	126.6638	616.90
DGP 2-layer (2)	0.4217	0.9192	0.4490	100.6576	618.34
DGP 2-layer (3)	0.5616	0.7897	0.5677	73.6139	610.71
DGP 3-layer (1)	0.4823	0.8937	**0.3705**	87.8325	616.10
DGP 3-layer (2)	**0.4045**	**0.9240**	0.4178	**128.5832**	**619.20**
DGP 3-layer (3)	0.5221	0.8895	0.5377	104.7551	615.37
DGP 1-layer (10) W/o	0.6761	0.7635	0.7710	80.3565	600.03
DGP 2-layer (10) W/o	0.6028	0.8459	0.6382	54.3635	603.22
DGP 3-layer (10) W/o	0.6339	0.8096	0.6329	69.3009	601.48
DGP 1-layer (2) Truth	0.4344	0.9150	**0.4994**	53.8692	617.46
DGP 2-layer (2) Truth	0.4917	0.8929	0.5207	90.0178	616.95
DGP 3-layer (2) Truth	**0.3215**	**0.9371**	0.5308	**113.6072**	**620.88**

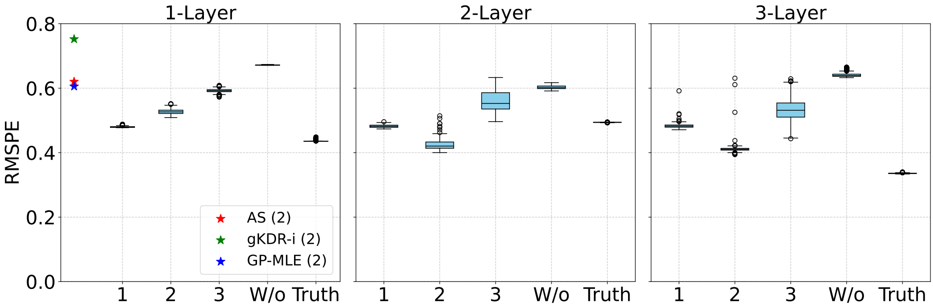


Figure: RMSPE comparisons across different method for train size 240

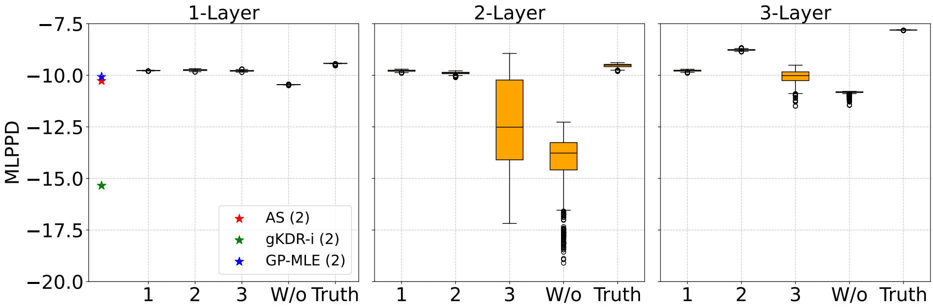


Figure: MLPPD comparisons across different method for train size 240

Discussion and Conclusion

Simulation Results:

- Models with built-in DR consistently outperformed models W/o.
- DGPs with appropriate layer depth adapted well, regardless of the complexity.
- Very deep models (3 layers) sometimes showed diminishing returns or overfitting in low-complexity settings.
- Performance gains from added depth were more apparent when the response surface was complex.

Discussion and Conclusion

Simulation Results:

- Models with built-in DR consistently outperformed models W/o.
- DGPs with appropriate layer depth adapted well, regardless of the complexity.
- Very deep models (3 layers) sometimes showed diminishing returns or overfitting in low-complexity settings.
- Performance gains from added depth were more apparent when the response surface was complex.

Conclusion:

- Model selection is key since data complexity is not known beforehand.
- Start with a moderate number of DGP layers and D, tune for the data at hand.
- Fully Bayesian DGPs with dimension reduction provide flexible modeling and robust performance across a range of complexities.

Thank you

References I

[1] Mohamed Amine Bouhlel et al. "An Improved Approach for Estimating the Hyperparameters of the Kriging Model for High-Dimensional Problems through the Partial Least Squares Method". In: *Mathematical Problems in Engineering* 2016 (2016), pp. 1–11. URL: https://api.semanticscholar.org/CorpusID:55011497.

[2] Ronald W. Butler and Andrew T.A. Wood. "Laplace approximation for Bessel functions of matrix argument". In: *Journal of Computational and Applied Mathematics* 155.2 (2003), pp. 359–382. ISSN: 0377-0427. DOI: https://doi.org/10.1016/S0377-0427(02)00874-9. URL: https://www.sciencedirect.com/science/article/pii/S0377042702008749.

[3] Yasuko Chikuse. "Concentrated matrix Langevin distributions". In: *Journal of Multivariate Analysis* 85.2 (2003), pp. 375–394. ISSN: 0047-259X. DOI: https://doi.org/10.1016/S0047-259X(02)00065-9. URL: https://www.sciencedirect.com/science/article/pii/S0047259X02000659.

[4] Paul G. Constantine, Eric Dow, and Qiqi Wang. "Active Subspace Methods in Theory and Practice: Applications to Kriging Surfaces". In: *SIAM Journal on Scientific Computing* 36.4 (2014), A1500–A1524. DOI: 10.1137/130916138. eprint: https://doi.org/10.1137/130916138. URL: https://doi.org/10.1137/130916138.

[5] Kenji Fukumizu and Chenlei Leng. *Gradient-based kernel dimension reduction for supervised learning*. 2011. arXiv: 1109.0455 [stat.ML]. URL: https://arxiv.org/abs/1109.0455.

[6] Raphael Gautier et al. *A Fully Bayesian Gradient-Free Supervised Dimension Reduction Method using Gaussian Processes*. 2022. DOI: 10.1615/Int.J.UncertaintyQuantification.2021035621. arXiv: 2008.03534 [stat.ML]. URL: https://arxiv.org/abs/2008.03534.

[7] Robert B Gramacy and Herbert K. H Lee. "Bayesian Treed Gaussian Process Models With an Application to Computer Modeling". In: *Journal of the American Statistical Association* 103.483 (2008), pp. 1119–1130. DOI: 10.1198/016214508000000689. eprint: https://doi.org/10.1198/016214508000000689. URL: https://doi.org/10.1198/016214508000000689.

References II

[8] Peter D. Hoff. "Simulation of the Matrix Bingham–von Mises–Fisher Distribution, With Applications to Multivariate and Relational Data". In: *Journal of Computational and Graphical Statistics* 18.2 (2009), pp. 438–456. DOI: 10.1198/jcgs.2009.07177. eprint: https://doi.org/10.1198/jcgs.2009.07177. URL: https://doi.org/10.1198/jcgs.2009.07177.

[9] Dimitrios Kapsoulis et al. "The use of Kernel PCA in evolutionary optimization for computationally demanding engineering applications". In: *2016 IEEE Symposium Series on Computational Intelligence (SSCI)* (2016), pp. 1–8. URL: https://api.semanticscholar.org/CorpusID:14548071.

[10] Plamen Koev and Alan Edelman. *The Efficient Evaluation of the Hypergeometric Function of a Matrix Argument*. 2005. arXiv: math/0505344 [math.PR]. URL: https://arxiv.org/abs/math/0505344.

[11] Xiaoyu Liu and Serge Guillas. "Dimension Reduction for Gaussian Process Emulation: An Application to the Influence of Bathymetry on Tsunami Heights". In: *SIAM/ASA Journal on Uncertainty Quantification* 5.1 (2017), pp. 787–812. DOI: 10.1137/16M1090648. eprint: https://doi.org/10.1137/16M1090648. URL: https://doi.org/10.1137/16M1090648.

[12] Annie Sauer, Robert B. Gramacy, and David Higdon. "Active Learning for Deep Gaussian Process Surrogates". In: *Technometrics* 65.1 (2023), pp. 4–18. URL: https://doi.org/10.1080/00401706.2021.2008505.

[13] Jun Tao et al. "Application of a PCA-DBN-based surrogate model to robust aerodynamic design optimization". In: *Chinese Journal of Aeronautics* (2020). URL: https://api.semanticscholar.org/CorpusID:216240132.

[14] Rohit Tripathy, Ilias Bilionis, and Marcial Gonzalez. "Gaussian processes with built-in dimensionality reduction: Applications to high-dimensional uncertainty propagation". In: *Journal of Computational Physics* 321 (2016), pp. 191–223. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2016.05.039. URL: https://www.sciencedirect.com/science/article/pii/S002199911630184X.

References III

[15] P. Tsilifis and R. G. Ghanem. "Bayesian adaptation of chaos representations using variational inference and sampling on geodesics". In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 474.2217 (2018), p. 20180285. DOI: 10.1098/rspa.2018.0285. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rspa.2018.0285. URL: https://royalsocietypublishing.org/doi/abs/10.1098/rspa.2018.0285.

[16] Panagiotis Tsilifis et al. "Bayesian learning of orthogonal embeddings for multi-fidelity Gaussian Processes". In: *Computer Methods in Applied Mechanics and Engineering* 386 (2021), p. 114147. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2021.114147. URL: https://www.sciencedirect.com/science/article/pii/S0045782521004783.

[17] Tong Zhou and Yong-bo Peng. "Kernel principal component analysis-based Gaussian process regression modelling for high-dimensional reliability analysis". In: *Computers & Structures* (2020). URL: https://api.semanticscholar.org/CorpusID:224947503.

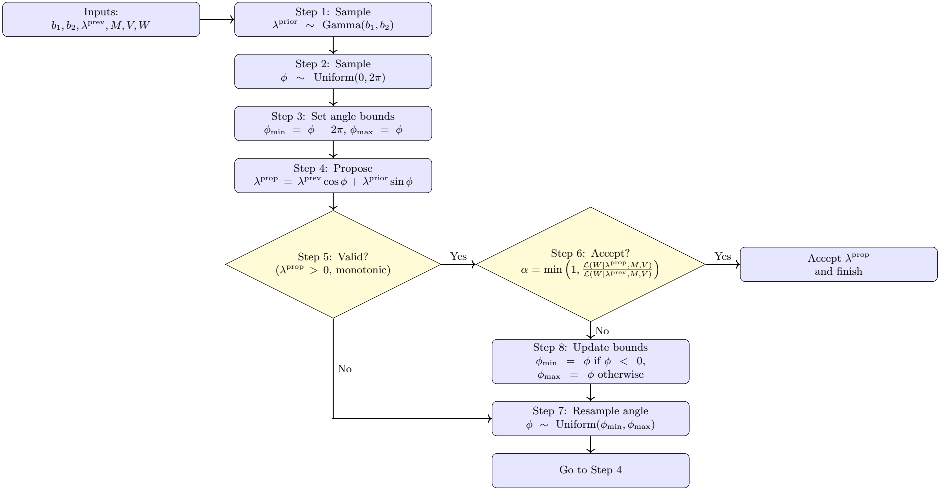


Figure: Elliptical slice sampling procedure for the concentration parameter (λ)

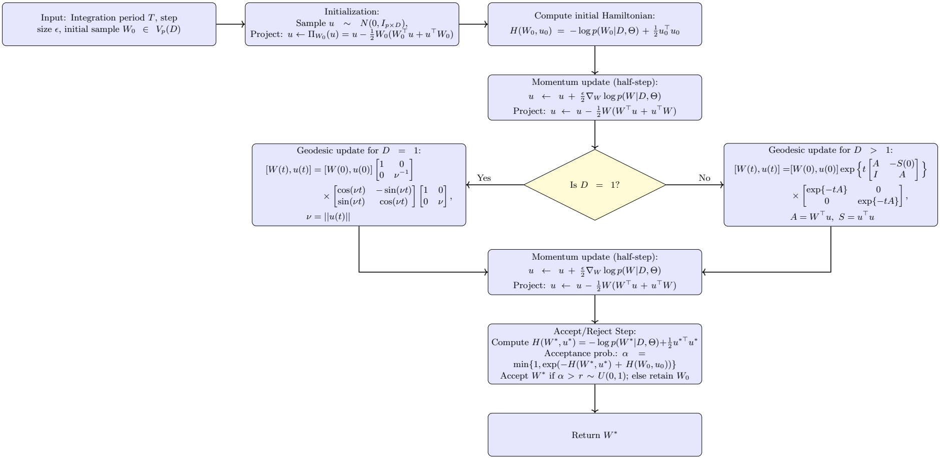


Figure: Geodesic Monte Carlo Sampling procedure on the Stiefel Manifold$(V_{p,D})$

Algorithm 1 : Geodesic Monte Carlo Algorithm on the Stiefel Manifold

Input: Integration period T, step size ϵ, initial sample $W_0 \in V_p(D)$
Output: Sample W^* from the posterior $p(W|D,\Theta)$
Initialization:

Sample $u \sim N(0, I_{p\times D})$ and project onto the tangent space:

$$u \leftarrow \Pi_{W_0}(u) = u - \frac{1}{2}W_0(W_0^\top u + u^\top W_0)$$

Compute initial Hamiltonian:

$$H(W_0, u_0) = -\log p(W_0|D,\Theta) + \frac{1}{2}u_0^\top u_0$$

For $m = 1, \dots, T$**:**

Momentum Update (Half-Step):

$$u \leftarrow u + \frac{\epsilon}{2}\nabla_W \log p(W|D,\Theta), \qquad u \leftarrow \Pi_W(u) = u - \frac{1}{2}W(W^\top u + u^\top W)$$

Position and Momentum Update (Geodesic Flow):

If $D = 1$ **(Hypersphere):**

$$[W(t), u(t)] = [W(0), u(0)] \begin{bmatrix} 1 & 0 \\ 0 & \nu^{-1} \end{bmatrix} \begin{bmatrix} \cos(\nu t) & -\sin(\nu t) \\ \sin(\nu t) & \cos(\nu t) \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \nu \end{bmatrix}$$

where $\nu = \|u(0)\|$

Else $(D > 1)$**:**

$$[W(t), u(t)] = [W(0), u(0)] \exp\left\{ t \begin{bmatrix} A & -S(0) \\ I & A \end{bmatrix} \right\} \begin{bmatrix} \exp(-tA) & 0 \\ 0 & \exp(-tA) \end{bmatrix}$$

where $t = \epsilon$; $A = W^\top u$; $S = u^\top u$; I identity.

Momentum Update (Half-Step):

$$u \leftarrow u + \frac{\epsilon}{2}\nabla_W \log p(W|D,\Theta), \qquad u \leftarrow \Pi_W(u) = u - \frac{1}{2}W(W^\top u + u^\top W)$$

Accept/Reject Step:

Compute $H(W^*, u^*) = -\log p(W^*|D,\Theta) + \frac{1}{2}u^{*\top}u^*$
Compute acceptance probability
$\alpha = \min\{1, \exp(-H(W^*, u^*) + H(W_0, u_0))\}$
Accept W^* with probability α if $\alpha > r \sim \text{Uniform}(0,1)$; otherwise, retain W_0

return W^*