Derivative Enhanced Gaussian Processes

Samuel Roberts†
Mechanical Engineering
Ph.D. Student

Mauricio Aristizabal†
James Warner‡
Juan Camilo Velasquez Gonzalez†
David Restrepo†
Harry Millwater†

†The University of Texas at San Antonio
‡ National Aeronautics and Space Administration. Langley Research Center

Gaussian Processes: An Introduction

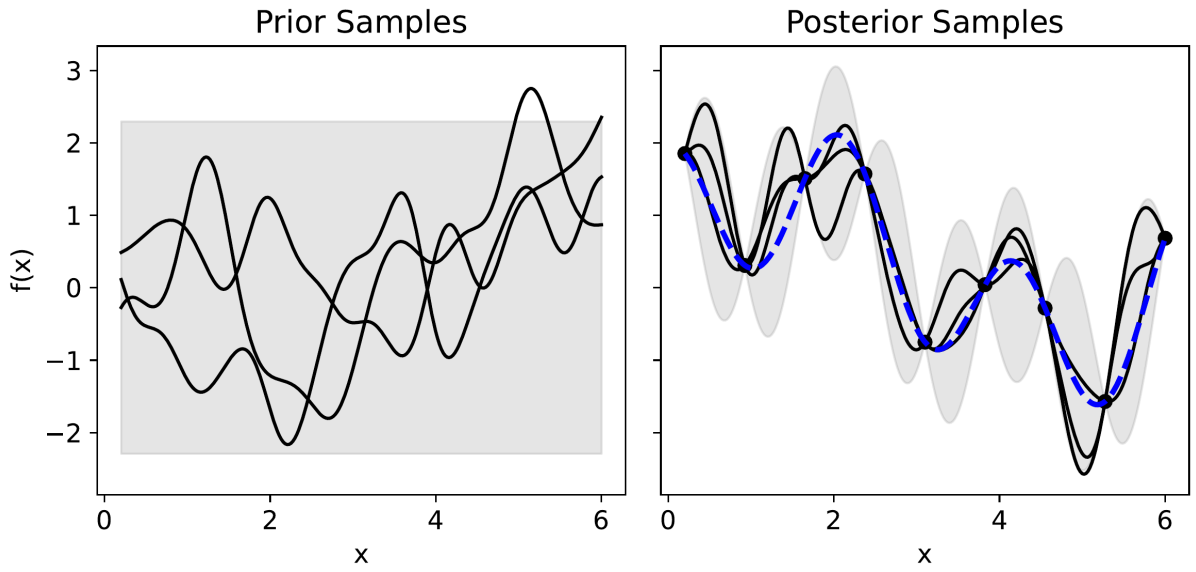
- What is a Gaussian Process?
  - A Gaussian Process (GP) is a probabilistic model where any finite collection of function values follows a multivariate normal distribution. It defines a distribution over functions:

$$f(\boldsymbol{x}) \sim GP(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}'))$$

- Where:

$m(\boldsymbol{x}) = E\big(f(\boldsymbol{x})\big)$ is the mean function

$k(\boldsymbol{x}, \boldsymbol{x}') = cov(\boldsymbol{x}, \boldsymbol{x}')$ is the covariance function (or kernel), defining correlations between points

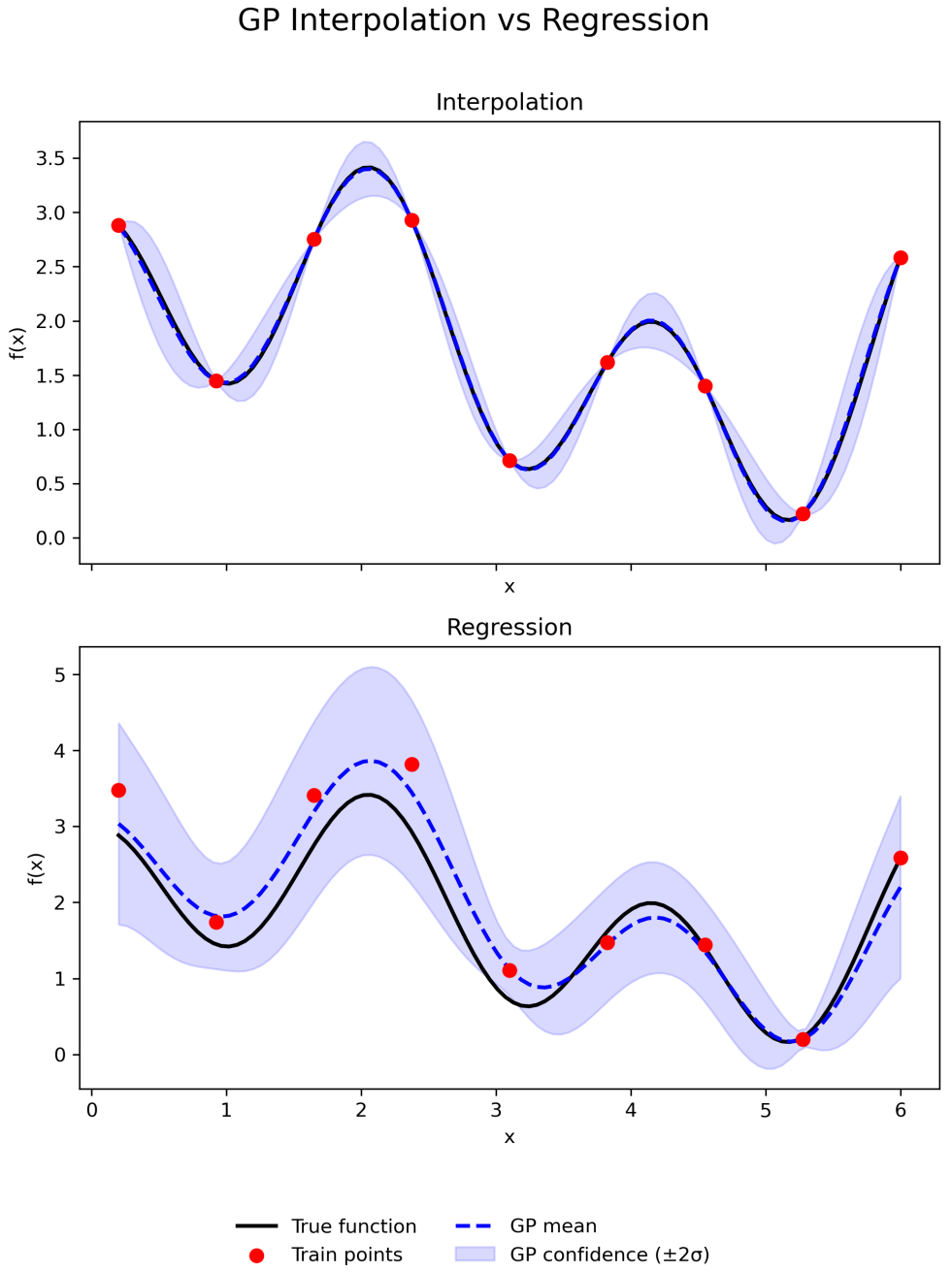


- ✅ **Fast to evaluate** after training – ideal for replacing expensive simulations
- 📈 **Provides uncertainty estimates** – confidence intervals highlight regions of low data density
- 🧠 **Non-parametric and flexible** – adapts to complex, nonlinear trends in data
- 🧩 **Supports derivative observations** – ideal for physics-informed or gradient-enhanced modeling

Why Use Gaussian Processes?

Gaussian Processes are widely used for:

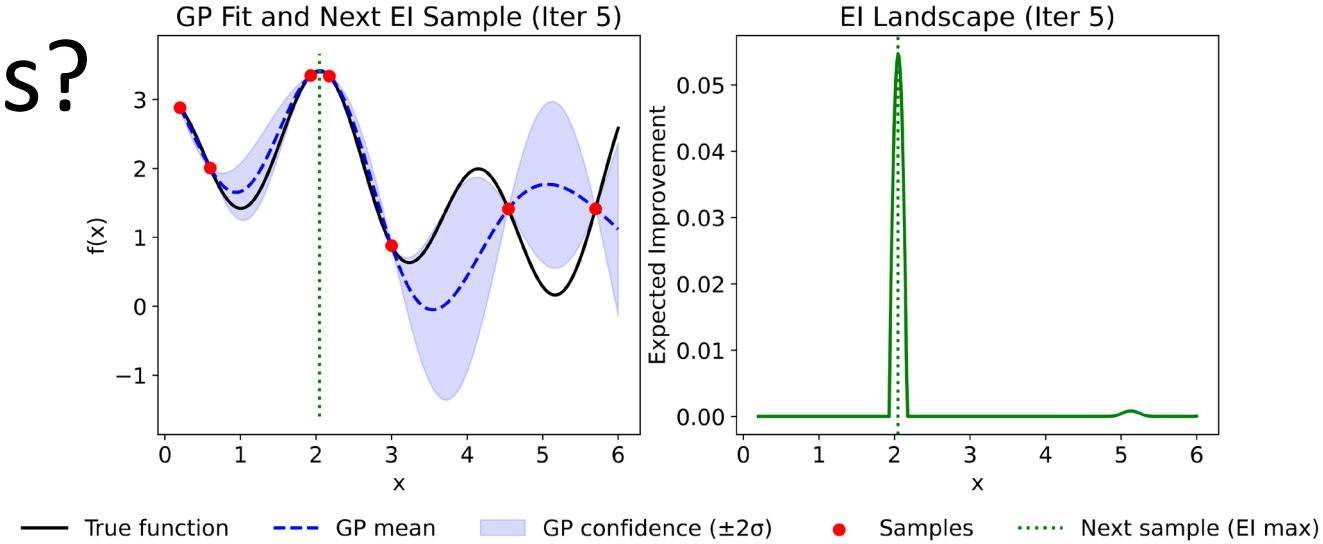
- **Interpolation**: Estimating smooth function values at unseen inputs while quantifying uncertainty
- **Regression**: Modeling functions from sparse or noisy data



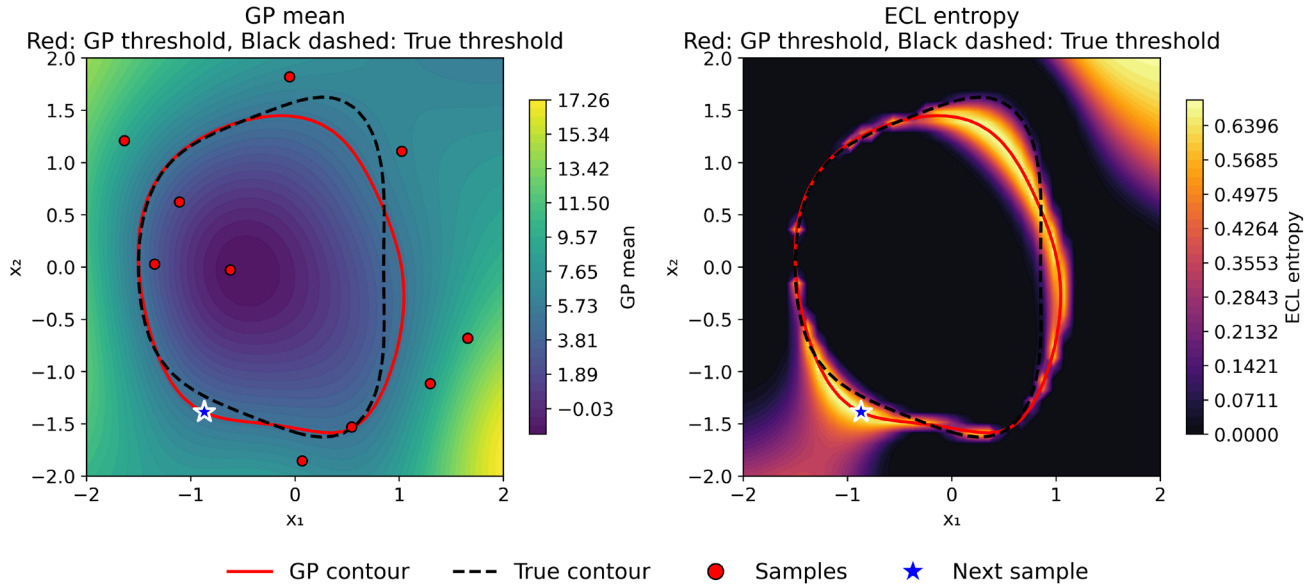
Why Use Gaussian Processes?

Gaussian Processes are widely used for:

- **Regression**: Modeling functions from sparse or noisy data
- **Interpolation**: Estimating smooth function values at unseen inputs while quantifying uncertainty
- **Bayesian active learning**: Guiding expensive simulations or experiments according to an acquisition function such as expected improvement EI or entropy-based contour locator ECL



Bayesian optimization with EI acquisition function

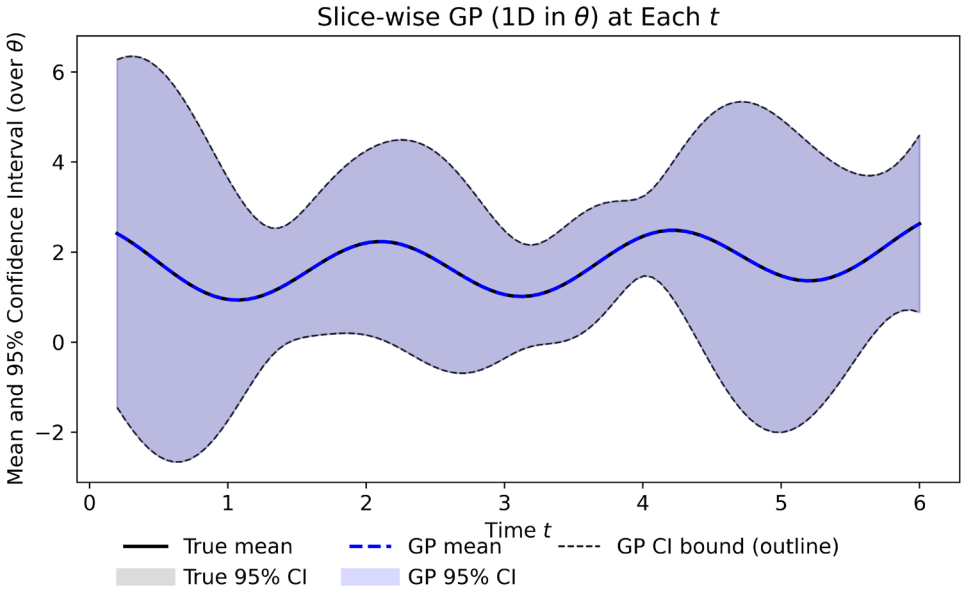


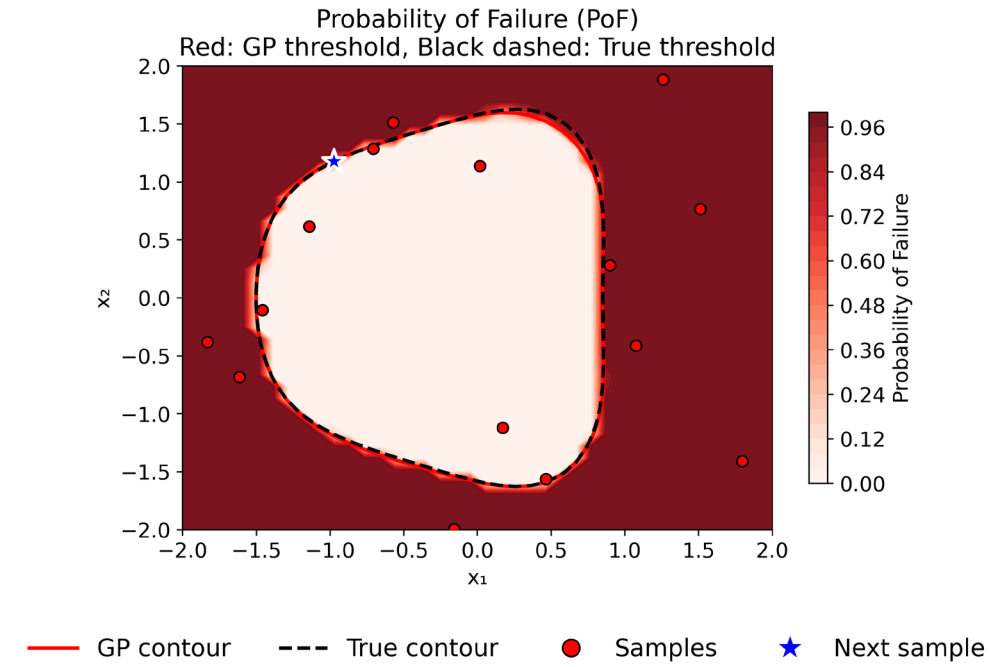
Bayesian active learning with ECL acquisition function

Why Use Gaussian Processes?

Gaussian Processes are widely used for:

- **Regression**: Modeling functions from sparse or noisy data
- **Interpolation**: Estimating smooth function values at unseen inputs while quantifying uncertainty
- **Bayesian active learning**: Guiding expensive simulations or experiments
- **Uncertainty Quantification**: Capturing model uncertainty explicitly/using the surrogate to perform tasks such as calculating statistical moments and quantifying probability of failure via Monte Carlo type procedures





Gaussian Processes

- First, assume a function we wish to model, f, and place a Gaussian Process prior on the function:

$$f(\boldsymbol{x}) \sim GP(\mathbf{0}, k(\boldsymbol{x}, \boldsymbol{x}'))$$

- Then given training data $D = \{\boldsymbol{x}^{(i)}, f(\boldsymbol{x}^{(i)})\}$ and test points $\boldsymbol{x}_*$, the joint distribution is:

$$\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{y}_* \end{pmatrix} \sim N(\mathbf{0}, \Sigma) \text{ where } \Sigma = \begin{pmatrix} k(\boldsymbol{x}, \boldsymbol{x}') & k(\boldsymbol{x}, \boldsymbol{x}_*) \\ k(\boldsymbol{x}^*, \boldsymbol{x}) & k(\boldsymbol{x}_*, \boldsymbol{x}_*') \end{pmatrix} = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix} \quad \text{where: } \boldsymbol{y} = f(\boldsymbol{x}^{(i)})$$

- The posterior distribution (predictive distribution) is:

$$p(\boldsymbol{y}_*|\boldsymbol{y}) \sim N(\Sigma_{12}\Sigma_{11}^{-1}\boldsymbol{y}, \Sigma_{22} - \Sigma_{12}^T\Sigma_{11}^{-1}\Sigma_{12})$$

$$\updownarrow$$

$$p(\boldsymbol{y}_*|\boldsymbol{y}) = N(\boldsymbol{\mu}_{\boldsymbol{y}_*|\boldsymbol{y}}, \Sigma_{\boldsymbol{y}_*|\boldsymbol{y}})$$

Predictive Mean → $\boldsymbol{\mu}_{\boldsymbol{y}_*|\boldsymbol{y}} = \Sigma_{12}^T\Sigma_{11}^{-1}(\boldsymbol{y})$

Predictive Variance → $\Sigma_{\boldsymbol{y}_*|\boldsymbol{y}} = \Sigma_{22} - \Sigma_{12}^T\Sigma_{11}^{-1}\Sigma_{12}$

Gradient Enhanced Gaussian Processes

- Following the work of ([4] [5]) assume a Gaussian Process prior:

$$f(\boldsymbol{x}) \sim GP(\boldsymbol{0}, \Sigma)$$

- Then given training data $D = \left\{\boldsymbol{x}^{(i)}, f\left(\boldsymbol{x}^{(i)}\right), \partial f\left(\boldsymbol{x}^{(i)}\right)\right\}$ and test points $\boldsymbol{x}^*$, we have that:

$$\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{y}_* \end{pmatrix} \sim N\left(\begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}\right)$$

where $\boldsymbol{y} = [\boldsymbol{y}_0, \boldsymbol{y}_1]^T$ and $\boldsymbol{y}_0 = f\left(\boldsymbol{x}^{(i)}\right), \boldsymbol{y}_1 = \partial f\left(\boldsymbol{x}^{(i)}\right)$ and

$$\Sigma_{11} = \begin{bmatrix} k(\boldsymbol{x}, \boldsymbol{x}') & \dfrac{\partial k(\boldsymbol{x}, \boldsymbol{x}')}{\partial \boldsymbol{x}} \\ \dfrac{\partial k(\boldsymbol{x}, \boldsymbol{x}')}{\partial \boldsymbol{x}} & \dfrac{\partial^2 k(\boldsymbol{x}, \boldsymbol{x}')}{\partial \boldsymbol{x} \partial \boldsymbol{x}'} \end{bmatrix}$$

*Note this formulation generalizes to higher order [6]

$$p(\boldsymbol{y}_*|\boldsymbol{y}) = N(\boldsymbol{\mu}_{\boldsymbol{y}_*|\boldsymbol{y}}, \Sigma_{\boldsymbol{y}_*|\boldsymbol{y}})$$

$$\boldsymbol{\mu}_{\boldsymbol{y}_*|\boldsymbol{y}} = \Sigma_{12}^T \Sigma_{11}^{-1}(\boldsymbol{y}) \longleftarrow \text{Predictive Mean}$$

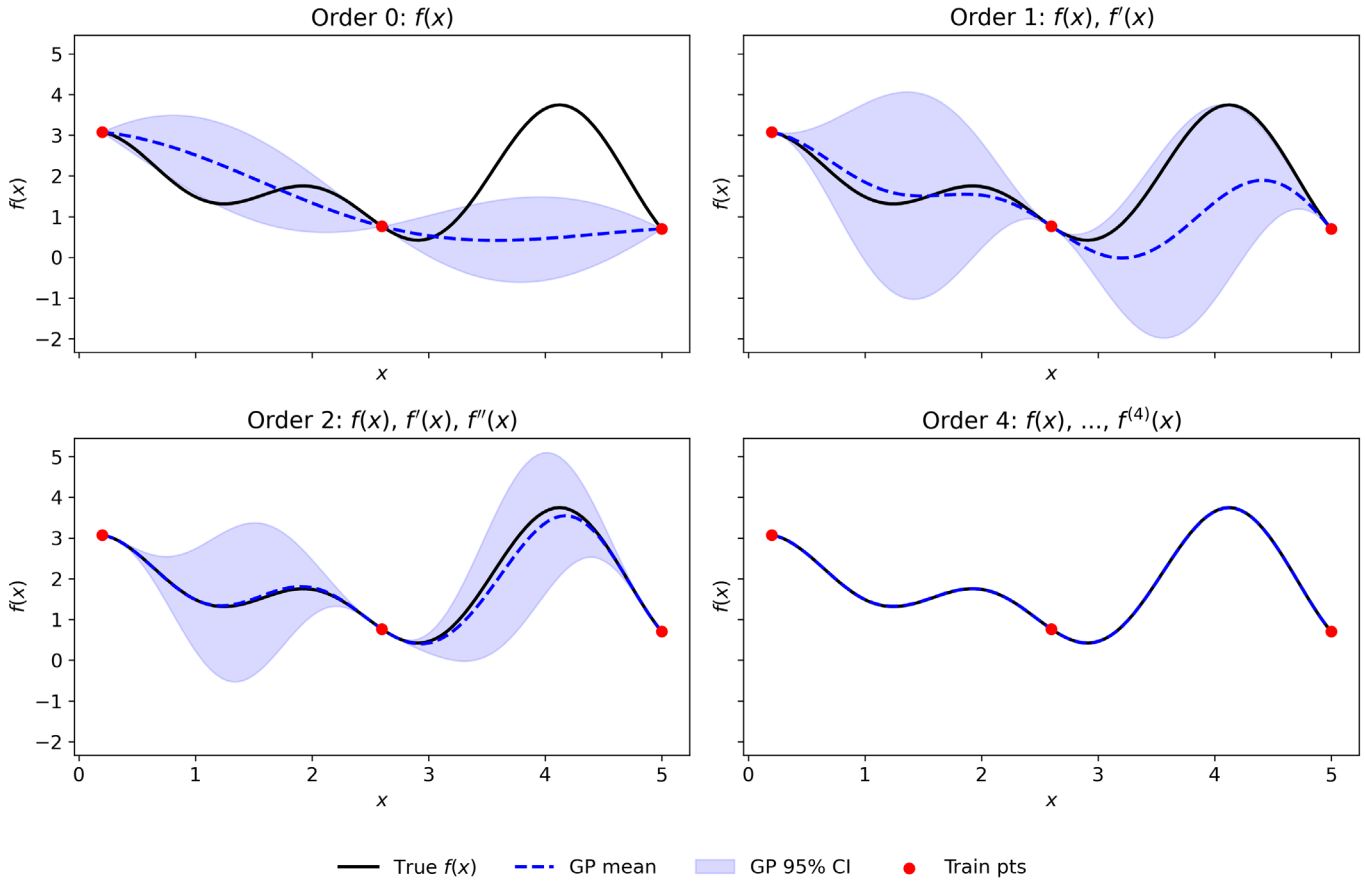
$$\Sigma_{\boldsymbol{y}_*|\boldsymbol{y}} = \Sigma_{22} - \Sigma_{12}^T \Sigma_{11}^{-1} \Sigma_{12} \longleftarrow \text{Predictive Variance}$$

In this formulation the size of Σ_{11} scales as $(n + nd)^2$ where d is the dimension of x

Higher-Order Derivative Enhanced Gaussian Processes

A Novel Advancement Enabled by HYPercomplex Automatic Differentiation

- This work introduces a **generalization of the Gaussian Process (GP) framework** that systematically incorporates **higher-order derivatives** (n > 2)
- This capability is made possible and motivated through **HYPAD** which enables efficient and scalable derivative computation



Hypercomplex Automatic Differentiation:

- The set of hypercomplex algebras contains those formed by one or more "**imaginary units**"
- E.G., (Complex Numbers $i^2 = -1$, Dual Numbers $\epsilon^2 = 0$)
- In general, HYPAD works by **perturbing an input parameter(s)** in one or more imaginary directions, **evaluating the model** at the perturbed input, and **extracting the imaginary part** of the evaluation:
- E.G., $f(x+ih) \approx \left(f(x) - \frac{1}{2}\frac{d^2f}{dx^2}h^2\right) + i\left(\frac{df}{dx}h - \frac{1}{3!}\frac{d^3f}{dx^3}ih^3\right) + HOT$

HYPAD Website

For sufficiently small h

$$f \approx \mathrm{Re}\big(f(x+ih)\big)$$

$$\frac{df}{dx} \approx \frac{\mathrm{Im}(f(x+ih))}{h}$$

Order Truncated Imaginary Algebra (OTI)

- The hypercomplex Order Truncated Imaginary (OTI) algebra is formed by imaginary directions constructed by the multiplication of imaginary basis $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$ with the property that **any imaginary direction with order greater than n is zero**

- $\mathbb{OTI}_m^n$ is the algebra formed with m **imaginary basis** and with **truncation order n**

OTI Library

E.G.,

$\mathbb{OTI}_3^1$: $a_0 + a_1\epsilon_1 + a_2\epsilon_2 + a_3\epsilon_3$ $\mathbb{OTI}_3^1$ computes all **first order derivatives** for **a function of three variables**

$\mathbb{OTI}_1^3$: $a_0 + a_1\epsilon_1 + a_2{\epsilon_1}^2 + a_3{\epsilon_1}^3$ $\mathbb{OTI}_1^3$ computes all **1st, 2nd, and 3rd order derivatives** for **a function of one variable**

$\mathbb{OTI}_2^2$: $a_0 + a_1\epsilon_1 + a_2\epsilon_2 + a_3{\epsilon_1}^2 + a_4{\epsilon_2}^2 + a_5\epsilon_1\epsilon_2$ $\mathbb{OTI}_2^2$ computes all **1st and 2nd order derivatives** for a function of **two variables**

Higher-Order Derivative Enhanced Gaussian Processes

- As before, assume a Gaussian Process prior:

$$f(\boldsymbol{x}) \sim GP(\mathbf{0}, \Sigma) \qquad \begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{y}_* \end{pmatrix} \sim N\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \right)$$

- The posterior distribution is:

$$p(\boldsymbol{y}^*|\boldsymbol{y}) \sim N(\Sigma_{12}^T \Sigma_{11}^{-1}\, \boldsymbol{y}, \Sigma_{22} - \Sigma_{12}^T \Sigma_{11}^{-1} \Sigma_{12})$$

$$p(\boldsymbol{y}_*|\boldsymbol{y}) = N(\boldsymbol{\mu}_{\boldsymbol{y}_*|\boldsymbol{y}}, \Sigma_{\boldsymbol{y}_*|\boldsymbol{y}})$$
$$\boldsymbol{\mu}_{\boldsymbol{y}_*|\boldsymbol{y}} = \Sigma_{12}^T \Sigma_{11}^{-1}(\boldsymbol{y})$$
$$\Sigma_{\boldsymbol{y}_*|\boldsymbol{y}} = \Sigma_{22} - \Sigma_{12}^T \Sigma_{11}^{-1} \Sigma_{12}$$

- In the fitting process a log marginal likelihood is maximized to determine the optimal hyperparameters for the covariance kernel k

$$\log\big(p(f^*|f)\big) = -\frac{1}{2} f^T \Sigma_{11}^{-1} f - \frac{1}{2}\log(\det(\Sigma_{11})) - \frac{n}{2}\log(2\pi)$$

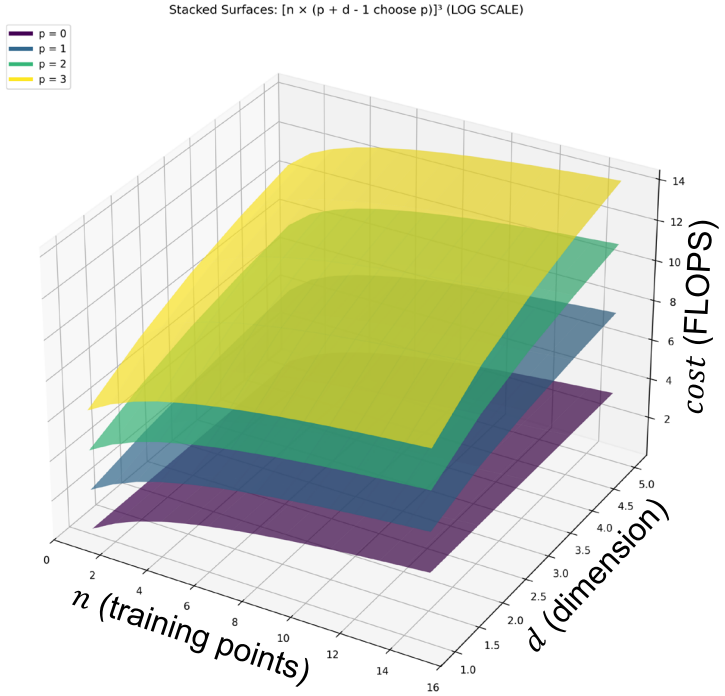
These operations must be performed many times and Σ_{11} scales with the dimension (d) and order (p) i.e., scales like $\left(\mathbf{n} \cdot \binom{\boldsymbol{p}+\boldsymbol{d}}{\boldsymbol{p}} - \mathbf{1},\right)^{\mathbf{2}}$ so for higher-order derivatives and-or higher-dimensional problems this training becomes computationally intractable

Analyzing Training Cost:

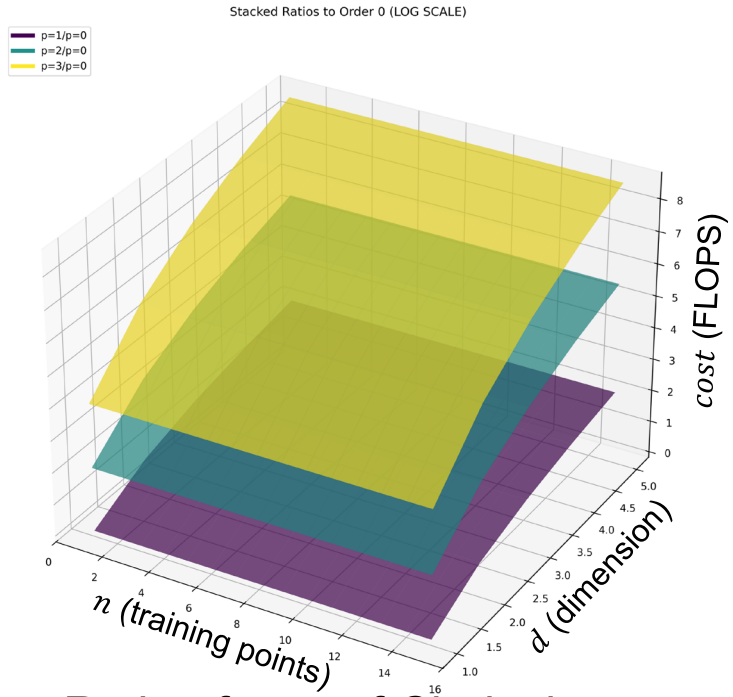
A log marginal likelihood is maximized to determine the optimal hyperparameters for the covariance kernel k

$$\log\left(p(f^*|f)\right) = -\frac{1}{2}f^T\Sigma_{11}^{-1}f\ -\frac{1}{2}\log(\det(\Sigma_{11})) - \frac{n}{2}\log(2\pi)$$

The primary expense of computing $\log\left(p(f^*|f)\right)$ comes in the form of the Cholesky decomposition of Σ_{11}



Cost of Cholesky decomposition of Σ_{11} by number of training points, dimension and order



Ratio of cost of Cholesky decomposition of Σ_{11} between order p and order 0

p	Cost Multiplier vs p=0
1	$\mathbf{125}\times$
2	$\mathbf{3.40e3}\times$
3	$\mathbf{4.30e4}\times$
4	$\mathbf{3.43e5}\times$
5	$\mathbf{2.00e6}\times$

$d = 5$

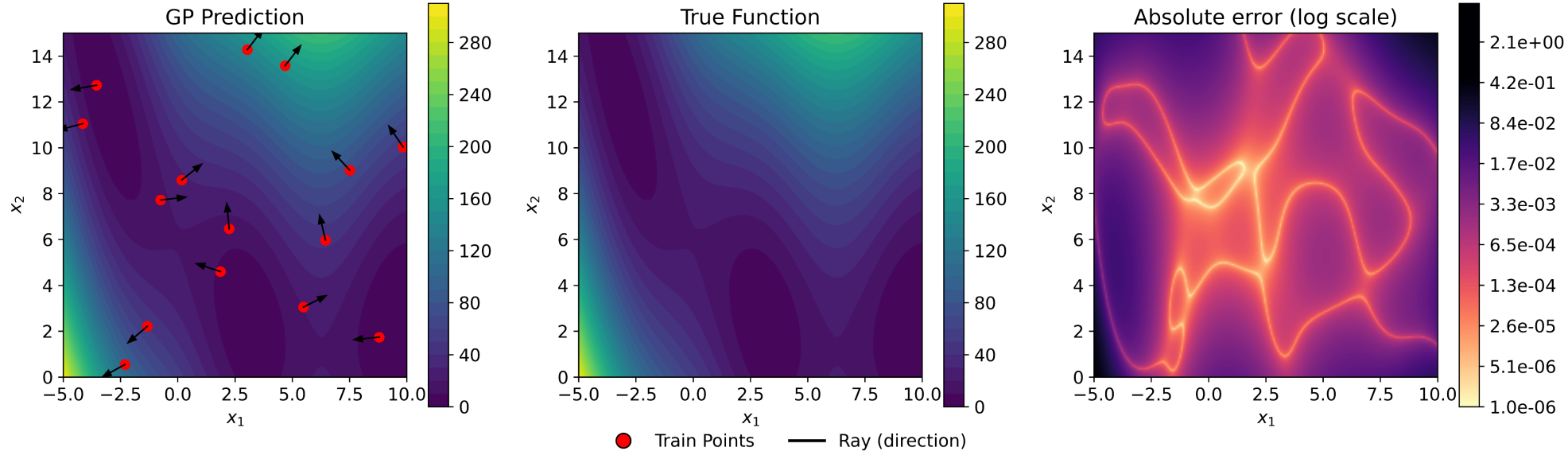
Addressing These Challenges

🧠 **Problem:**

- Computational cost increases with dimensionality and derivative order
- Full derivative inclusion → large covariance matrices + repeated inversions
- There are many proposed methodologies to address this problem [1], [2], [3]

Proposed Solution ⟶ ✅ **Directional Derivatives (DDEGP) [3]:**

The use of directional derivatives allows the use of higher-order derivative information in the Gaussian process, whilst simultaneously keeping the cost manageable



Directional Derivative Enhanced Gaussian Processes

- Assume a Gaussian Process prior:

$$f(\boldsymbol{x}) \sim GP(\boldsymbol{0}, \Sigma)$$

- Then, given training data $D = \left\{\boldsymbol{x}^{(i)}, f\left(\boldsymbol{x}^{(i)}\right), \frac{\partial f\left(\boldsymbol{x}^{(i)}\right)}{\partial \boldsymbol{v}_i}, \frac{\partial^2 f\left(\boldsymbol{x}^{(i)}\right)}{\partial \boldsymbol{v}_i^2}, \ldots \frac{\partial^2 f\left(\boldsymbol{x}^{(i)}\right)}{\partial \boldsymbol{v}_i^p}\right\}$ and test points $\boldsymbol{x}^*$, we have that:

$$\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{y}_* \end{pmatrix} \sim N\left(\begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}\right)$$

where $\boldsymbol{y} = [\boldsymbol{y}_0, \boldsymbol{y}_1]^T$ and $\boldsymbol{y}_0 = f\left(\boldsymbol{x}^{(i)}\right), \boldsymbol{y}_1 = \left\{\frac{\partial f\left(\boldsymbol{x}^{(i)}\right)}{\partial v^{(i)}}, \frac{\partial^2 f\left(\boldsymbol{x}^{(i)}\right)}{\partial \boldsymbol{v}_i^2}, \ldots \frac{\partial^2 f\left(\boldsymbol{x}^{(i)}\right)}{\partial \boldsymbol{v}_i^p}\right\}$ and

$$\Sigma_{11} = \begin{bmatrix} k(\boldsymbol{x}, \boldsymbol{x}') & \frac{\partial k\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}'\right)}{\partial \boldsymbol{v}_i} & \ldots & \frac{\partial^p k\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}'\right)}{\partial \boldsymbol{v}_i^p} \\ \frac{\partial k\left(\boldsymbol{x}, \boldsymbol{x}^{(j)}\right)}{\partial \boldsymbol{v}_j} & \frac{\partial^2 k\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right)}{\partial \boldsymbol{v}_j \partial \boldsymbol{v}_i} & \ldots & \frac{\partial^{(p+1)} k\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right)}{\partial \boldsymbol{v}_j \partial \boldsymbol{v}_i} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^p k\left(\boldsymbol{x}, \boldsymbol{x}^{(j)}\right)}{\partial \boldsymbol{v}_j^p} & \frac{\partial^{(p+1)} k\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right)}{\partial \boldsymbol{v}_j^p \partial \boldsymbol{v}_i} & \ldots & \frac{\partial^{2p} k\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right)}{\partial \boldsymbol{v}_j^p \partial \boldsymbol{v}_i^p} \end{bmatrix}$$

Note in this formulation the size of Σ_{11} scales as $(\boldsymbol{n} \times \boldsymbol{np})$ where $\boldsymbol{p}$ is the order of derivative

$$p(\boldsymbol{y}_*|\boldsymbol{y}) = N(\boldsymbol{\mu}_{\boldsymbol{y}_*|\boldsymbol{y}}, \Sigma_{\boldsymbol{y}_*|\boldsymbol{y}})$$

$$\boldsymbol{\mu}_{\boldsymbol{y}_*|\boldsymbol{y}} = \Sigma_{12}^T \Sigma_{11}^{-1}(\boldsymbol{y}) \longleftarrow \text{Predictive Mean}$$

$$\Sigma_{\boldsymbol{y}_*|\boldsymbol{y}} = \Sigma_{22} - \Sigma_{12}^T \Sigma_{11}^{-1} \Sigma_{12} \longleftarrow \text{Predictive Variance}$$

Gaussian Processes for PoF: A case study

Definition

- Active learning acquisition function that targets **decision boundaries**
- Focuses sampling on areas of maximum uncertainty, **specifically near a threshold**

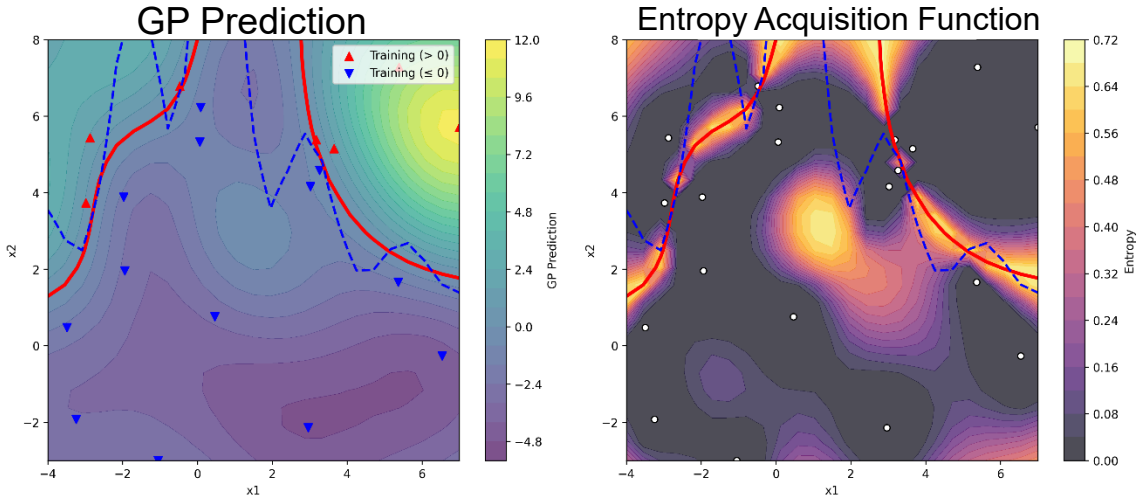
Formula

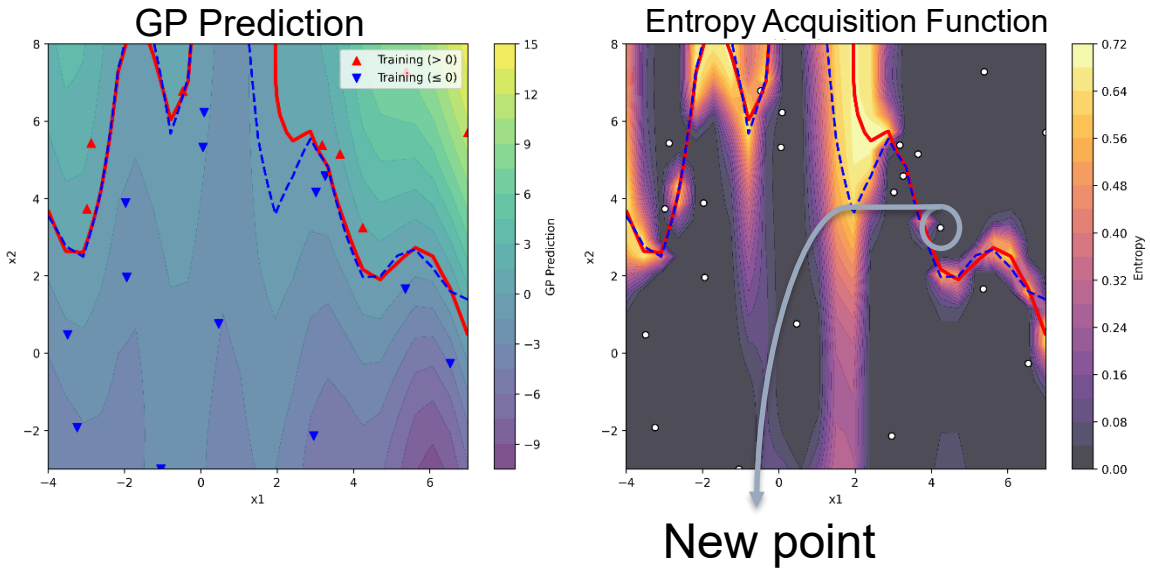
$$ECL(\boldsymbol{x}) = -\left(1 - \Phi\left(\frac{\mu_N(\boldsymbol{x}) - T}{\sigma_N(\boldsymbol{x})}\right)\right)\log\left(1 - \Phi\left(\frac{\mu_N(\boldsymbol{x}) - T}{\sigma_N(\boldsymbol{x})}\right)\right) - \Phi\left(\frac{\mu_N(\boldsymbol{x}) - T}{\sigma_N(\boldsymbol{x})}\right)\log\left(\Phi\left(\frac{\mu_N(\boldsymbol{x}) - T}{\sigma_N(\boldsymbol{x})}\right)\right)$$

- Select next training point according to: $\boldsymbol{x_{N+1}} = argmax ECL(\boldsymbol{x})$

Process

1. Fit GP model →
2. Calculate entropy at threshold →
3. Select highest entropy point →
4. Evaluate model at selected point →
5. Update GP model →
6. Repeat





Direction Selection:

- In ECL, the next training point is determined as: $\boldsymbol{x}_{N+1} = \operatorname{argmax}(ECL_N(\boldsymbol{x}))$

$$ECL_N(\boldsymbol{x}) = -\left(1 - \Phi\left(\frac{\mu_N(\boldsymbol{x}) - T}{\sigma_N(\boldsymbol{x})}\right)\right)\log\left(1 - \Phi\left(\frac{\mu_N(\boldsymbol{x}) - T}{\sigma_N(\boldsymbol{x})}\right)\right) - \Phi\left(\frac{\mu_N(\boldsymbol{x}) - T}{\sigma_N(\boldsymbol{x})}\right)\log\left(\Phi\left(\frac{\mu_N(\boldsymbol{x}) - T}{\sigma_N(\boldsymbol{x})}\right)\right)$$

- With directional derivatives there is an additional degree of freedom in selecting which at the point $\boldsymbol{x}_{N+1}$.
- To do this we introduce the following:

$$IMSER(x_{N+1}) \coloneqq \int (\sigma_{N+1}(\boldsymbol{x}) - \sigma_N(\boldsymbol{x}))d\boldsymbol{x} \qquad IMSER(x_{N+1}; \boldsymbol{v}) \coloneqq \int (\sigma_{N+1,\boldsymbol{v}}(\boldsymbol{x}) - \sigma_N(\boldsymbol{x}))d\boldsymbol{x}$$

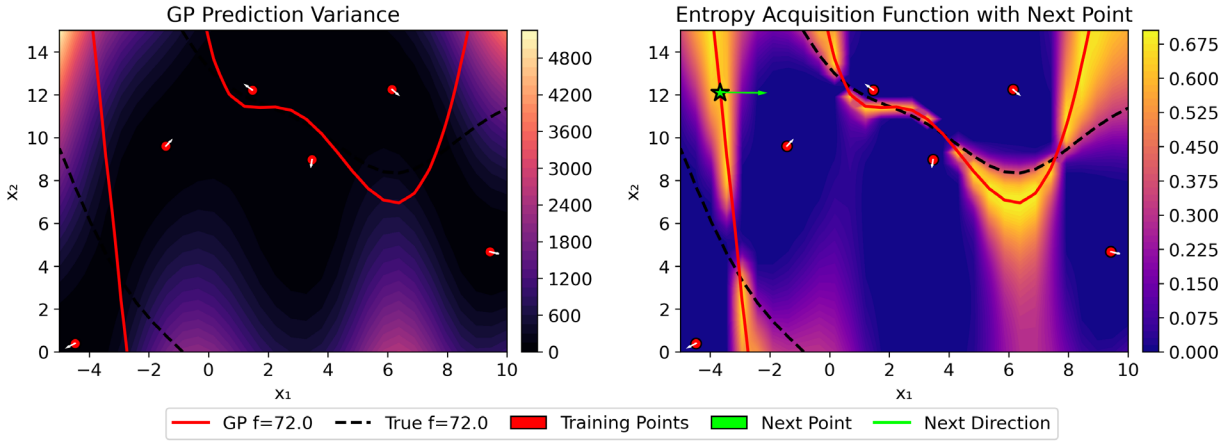
$$ER(x_{N+1}) \coloneqq \int \left(ECL_{N+1}(\boldsymbol{x}) - ECL_N(\boldsymbol{x})\right)d\boldsymbol{x} \qquad ER(x_{N+1}; \boldsymbol{v}) \coloneqq \int \left(ECL_{N+1;\boldsymbol{v}}(\boldsymbol{x}) - ECL_N(\boldsymbol{x})\right)d\boldsymbol{x}$$

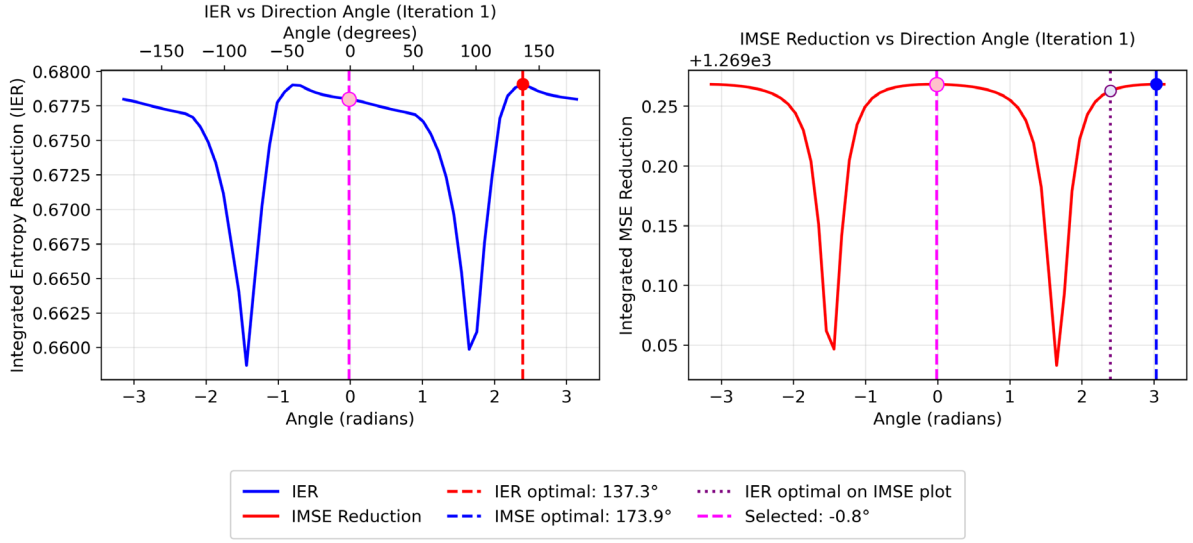
Here the integration is over a small ball centered at x_{N+1}

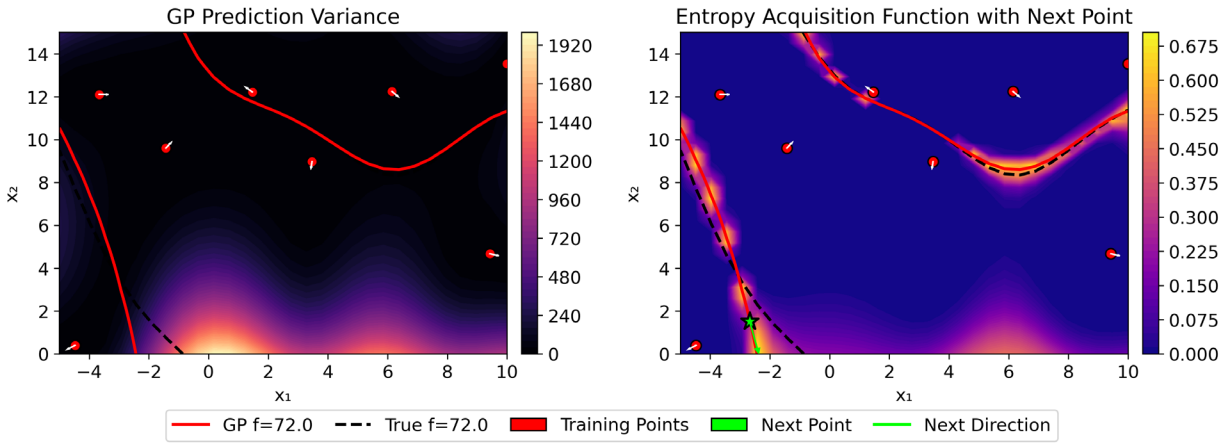
- Propose selecting $\boldsymbol{v}_{N+1}$ according to:

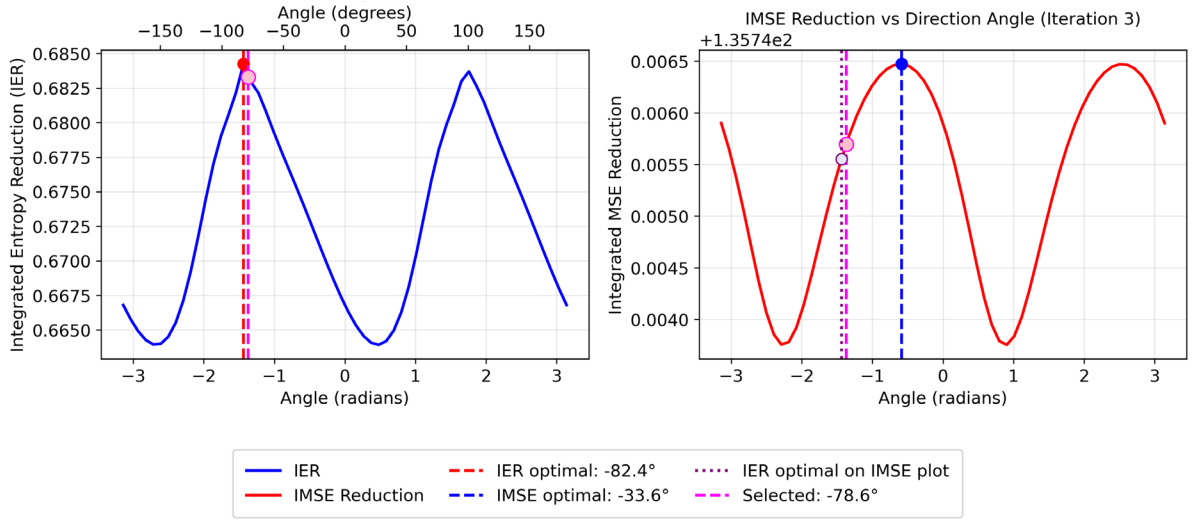
$$\boldsymbol{v}_{N+1} \coloneqq \underset{\boldsymbol{v}\ s.t\ ||\boldsymbol{v}||=1}{\operatorname{argmax}} \left(\underbrace{\frac{ER(\boldsymbol{x}_{N+1}; \boldsymbol{v})}{ER(\boldsymbol{x}_{N+1})}}_{\text{Exploitation}} + \underbrace{\frac{IMSE(\boldsymbol{x}_{N+1}; \boldsymbol{v})}{IMSE(\boldsymbol{x}_{N+1})}}_{\text{Exploration}}\right)$$

Direction Selection Intuition:









Selecting $\boldsymbol{v}_{N+1}$ according to: $\boldsymbol{v}_{N+1} \coloneqq \underset{\boldsymbol{v}\ s.t\ ||\boldsymbol{v}||=1}{\operatorname{argmax}} \left(\frac{ER(\boldsymbol{x}_{N+1};\boldsymbol{v})}{ER(\boldsymbol{x}_{N+1})} + \frac{IMSE(\boldsymbol{x}_{N+1};\boldsymbol{v})}{IMSE(\boldsymbol{x}_{N+1})} \right)$

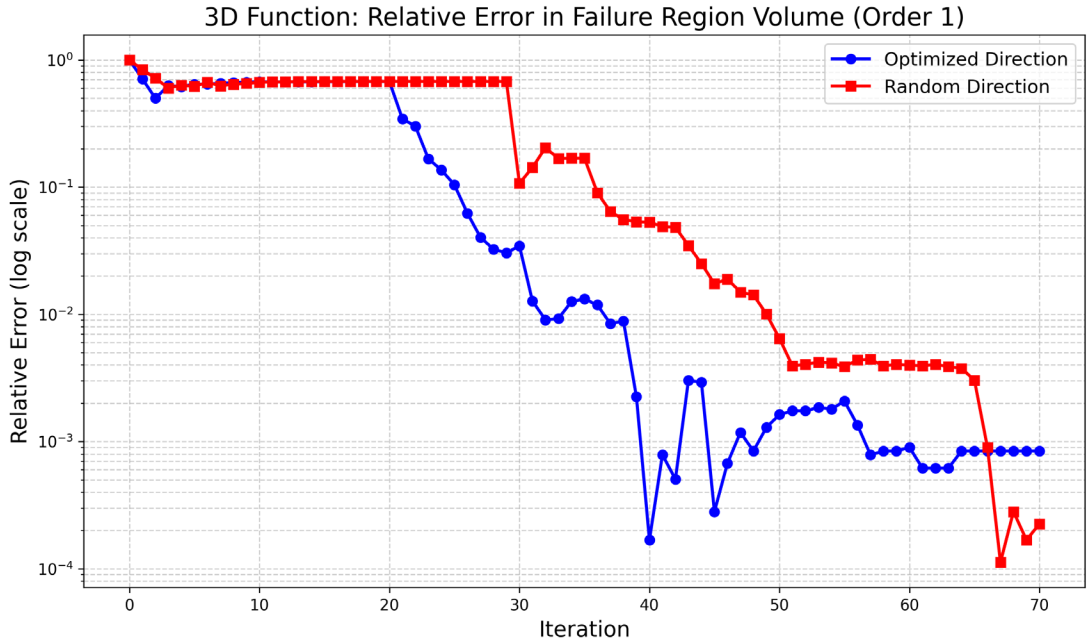
balances exploration and exploitation.

Direction Selection Case Study

Consider the function:

$$f(x_1, x_2, x_3) = 3\exp\left(-\frac{(x_1 - 1.5)^2 + (x_2 - 1)^2 + (x_3 - .5)^2}{.8}\right) + 2.5\exp\left(-\frac{(x_1 + 1)^2 + (x_2 + 1.5)^2 + (x_3 - 1.0)^2}{.6}\right) - 1.2$$

- Failure is defined to be whenever the function exceeds the threshold $T = 0$.
- The function was designed to have two disjoint failure regions.
- $Vol(G)$ is defined as the **analytic volume** of the failure region
- $Vol(\hat{G})$ is defined as the **GP volume** of the failure region
- The relative error is defined as: $\frac{abs\left(\boldsymbol{Vol(G) - Vol(\hat{G})}\right)}{\boldsymbol{Vol(G)}}$



Relative error in failure region volume, comparing randomly selected directions versus the optimized direction.

Derivative Enhanced GP ECL Case Study

The following benchmark functions are considered:

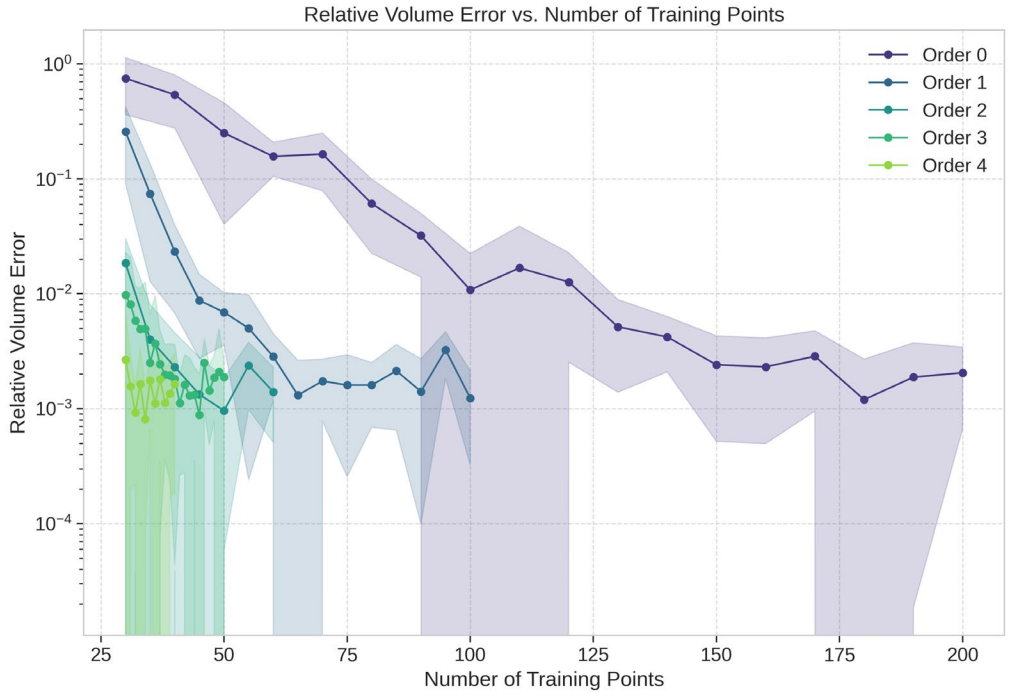
Function Name	Definition	Domain	Failure Criterion	Analytical Failure Volume	Quantile
Ishigami Function	$f(\boldsymbol{x}) = \sin(\mathrm{x}_1) + 7\sin^2(\mathrm{x}_2) + .05\mathrm{x}_3^4\sin(x_1)$	$x_i \in [-\pi, \pi]$ for $i = 1,2,3$	$f(x) > 10.244$	2.1783	.9999
Hartmann 6D	$f(x) = \sum_{i=1}^{4} \alpha_i \exp\left(-\sum_{j=1}^{6} A_{ij}(x_j - P_{ij})^2\right)$	$x_i \in [0,1]$ for $i = 1, \ldots, 6$	$f(x) > 2.63$	.011	.9989

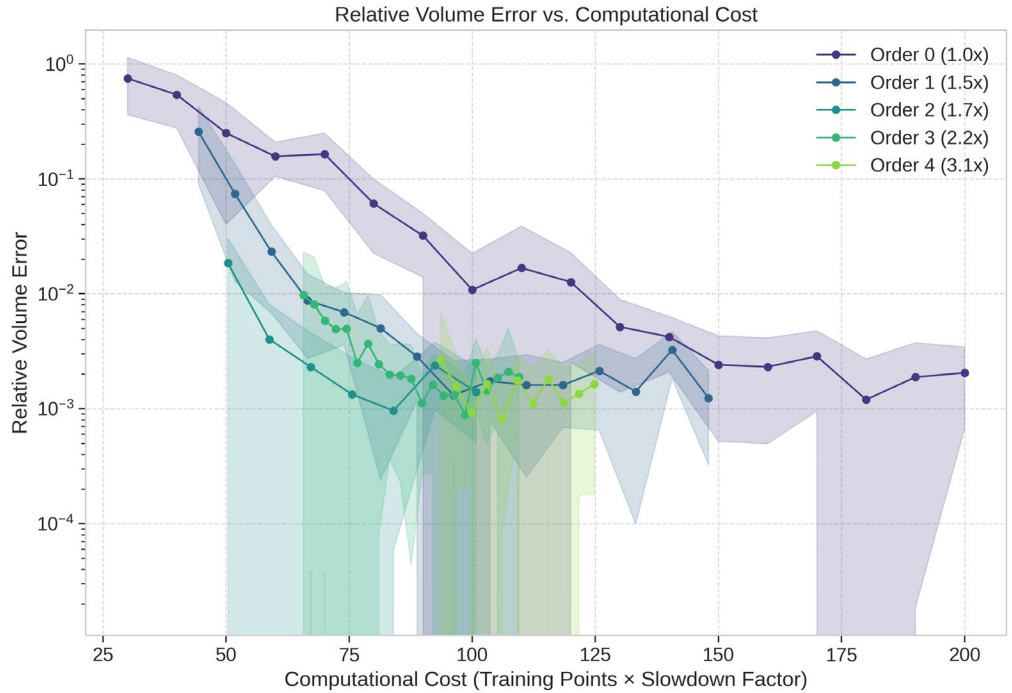
The following algorithm follows the methodology outlined in [7]

1. Initialize with 10d random training points
2. Train GP model on initial dataset/function and derivative evaluations
3. Sequential sampling loop:
    1. Generate candidate points across domain
    2. Evaluate ECL acquisition function at candidates
    3. Select point with highest entropy contribution
    4. Refine selection with local optimization
    5. Sample function/derivatives at optimized location
    6. Update GP model and repeat

Derivative Enhanced GP ECL Case Study

Ishigami function:





✅ A significant reduction in sample points needed to reach convergence.

✅ When accounting for the cost† of obtaining these derivatives, implementing up **to second order derivative information** gives **the best results**.

✅ In all cases there is a reduction in cost in terms of function evaluations

† Cost was calculated in accordance with results in [8]

Derivative Enhanced GP ECL Case Study

Hartmann 6D:

✅ A significant reduction in sample points needed to reach convergence.
✅ When accounting for the cost, higher order derivatives show a more favorable convergence trend when compared to no derivatives/first order derivatives

Derivative Enhanced GP ECL Case Study

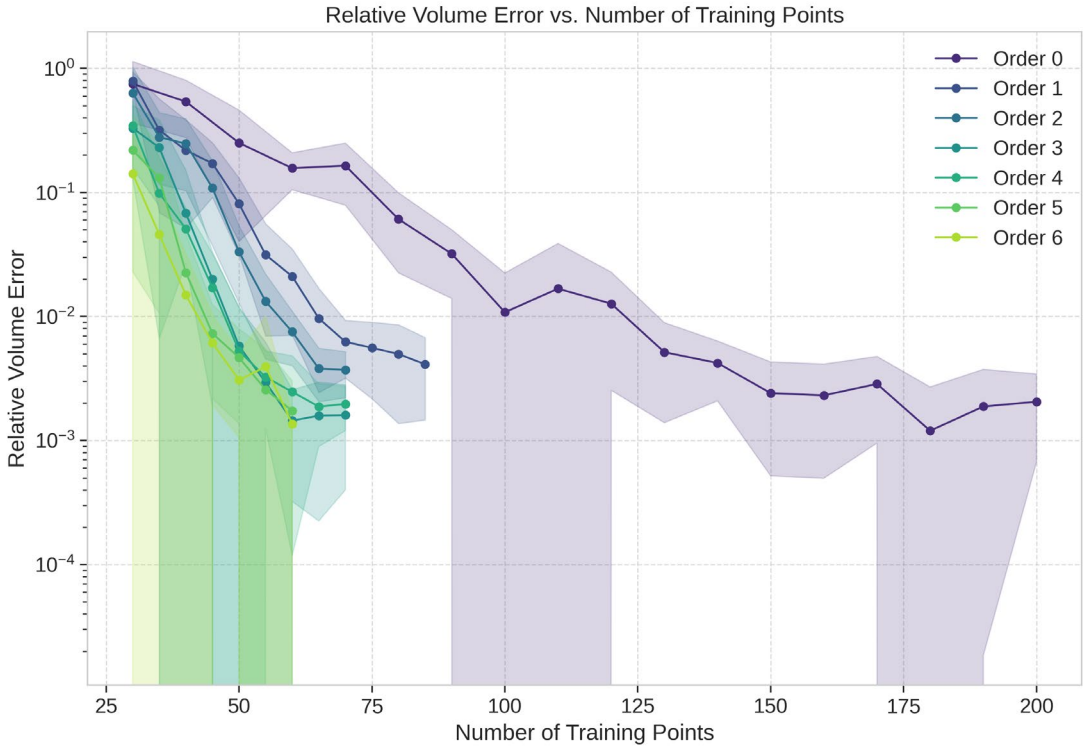
Function Name	Definition	Domain	Failure Criterion	Analytical Failure Volume	Quantile
Ishigami Function	$f(\boldsymbol{x}) = \sin(\mathrm{x}_1) + 7\sin^2(\mathrm{x}_2) + .05\mathrm{x}_3^4\sin(x_1)$	$x_i \in [-\pi, \pi]$ for $i = 1,2,3$	$f(x) > 10.244$	2.1783	.9999
Hartmann 6D	$f(x) = \sum_{i=1}^{4} \alpha_i \exp\left(-\sum_{j=1}^{6} A_{ij}\left(x_j - P_{ij}\right)^2\right)$	$x_i \in [0, 1]$ for $i = 1, \ldots, 6$	$f(x) > 2.63$	.011	.9989

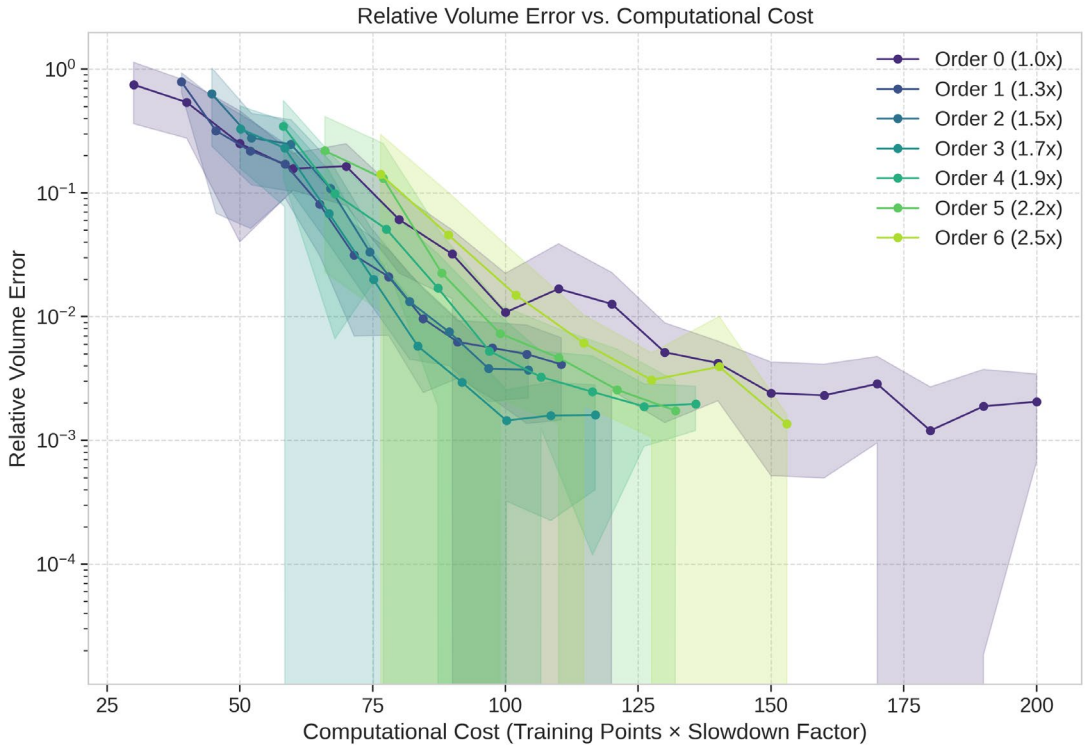
Here the following is proposed:

1. Initialize with 10d random training points with random initial directions
2. Train GP model on initial dataset, function/derivatives
3. Sequential sampling loop:
    1. Generate candidate points across domain
    2. Evaluate ECL acquisition function at candidates
    3. Select point with highest entropy contribution
    4. Refine selection with local optimization
    5. Select next direction
    6. Sample function/directional derivative at optimized location
    7. Update GP model and repeat

Derivative Enhanced GP ECL Case Study

Ishigami function:

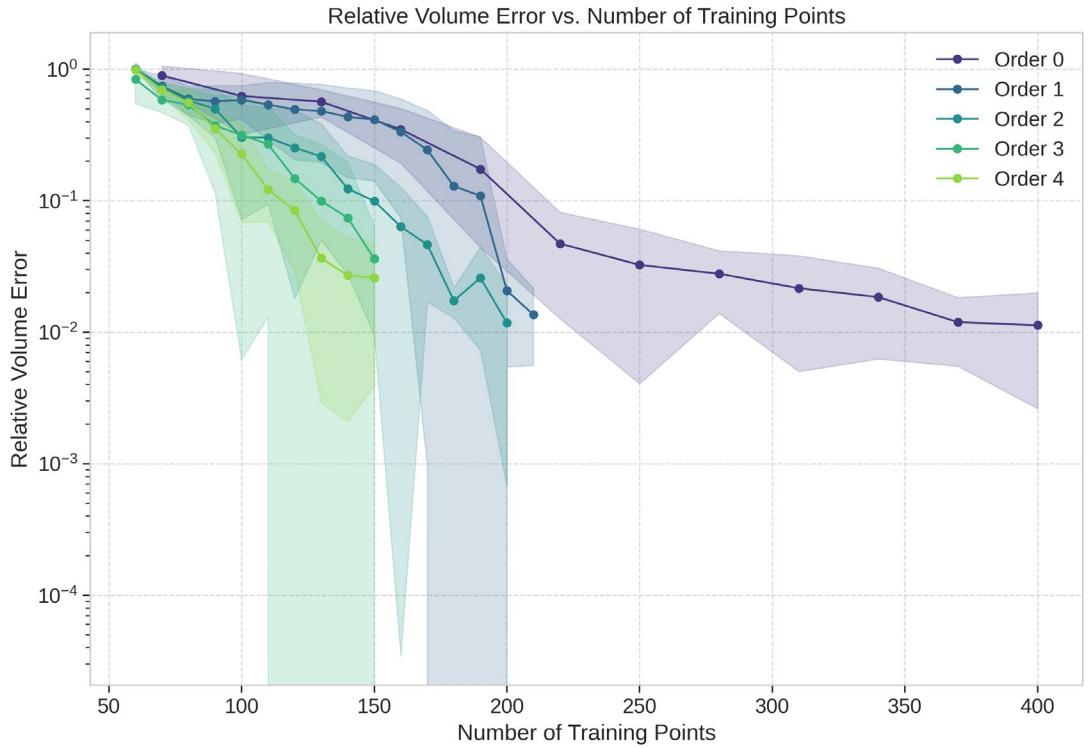


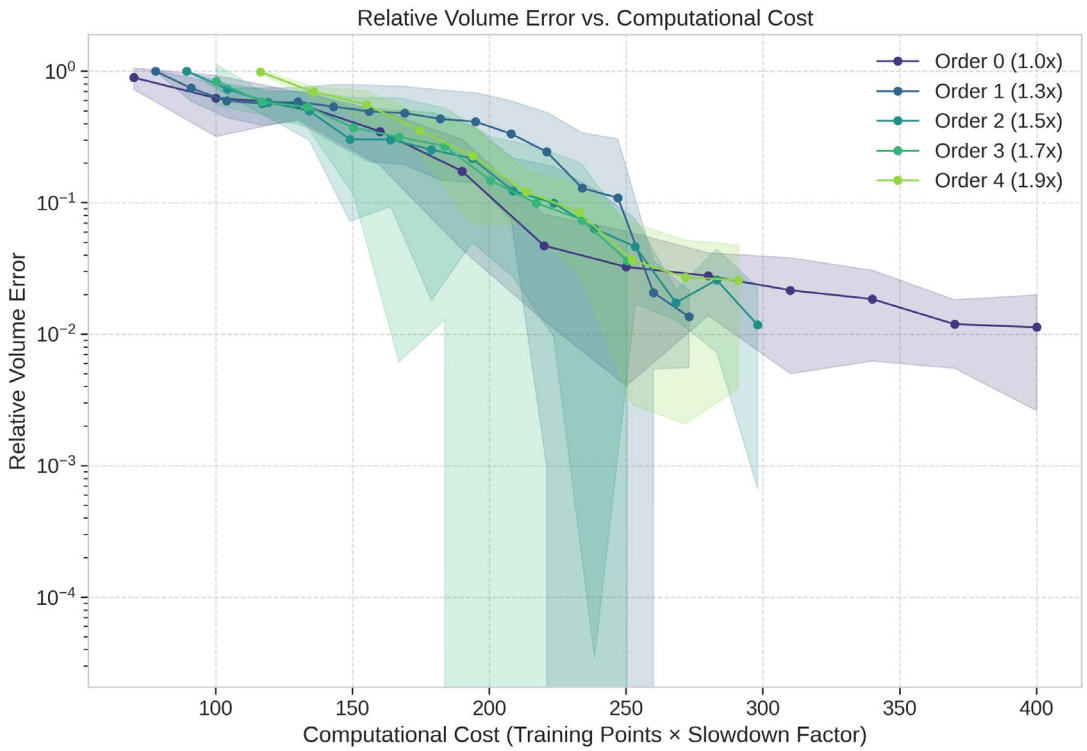


✅ A significant reduction in sample points needed to reach convergence.
✅ When accounting for the cost implementing up **to third order derivative information** gives **the best results**.
✅ In all cases there is a reduction in cost in terms of function evaluations

Derivative Enhanced GP ECL Case Study

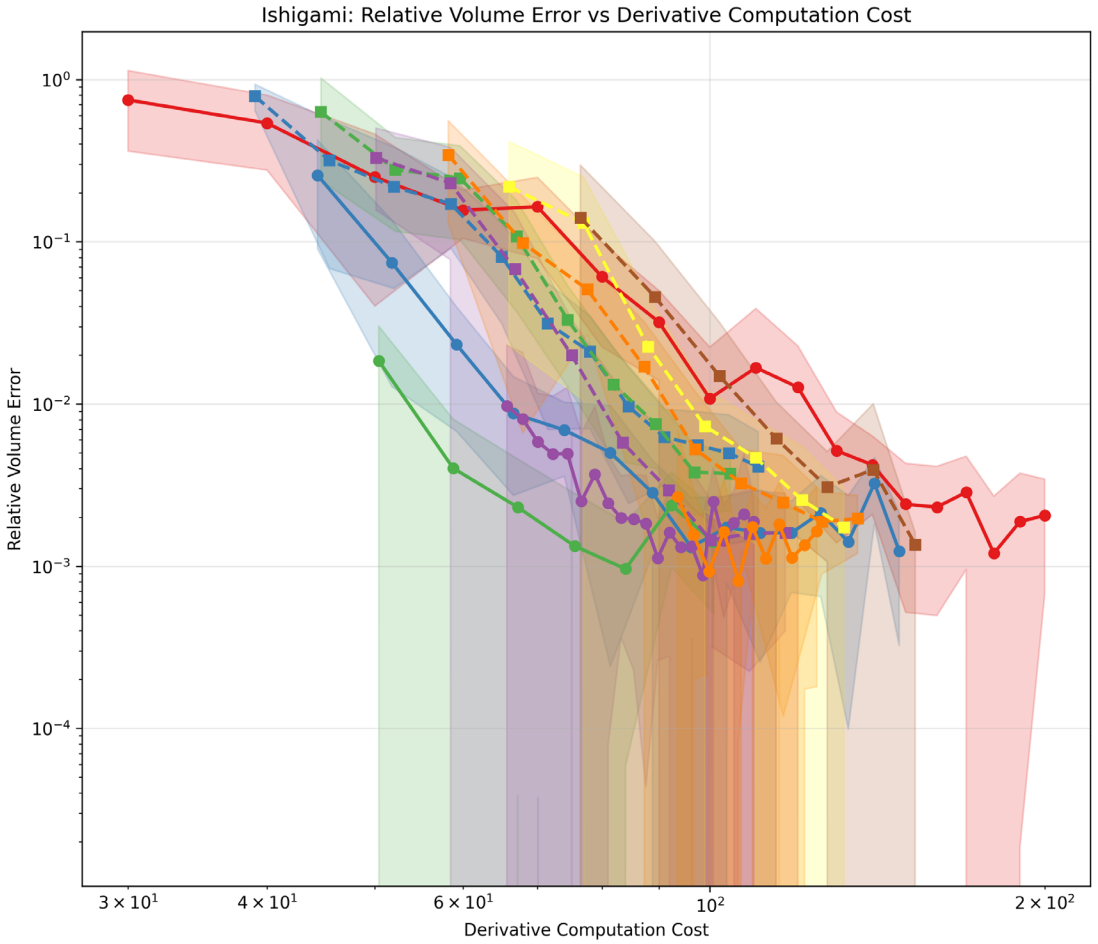
Hartmann 6D:





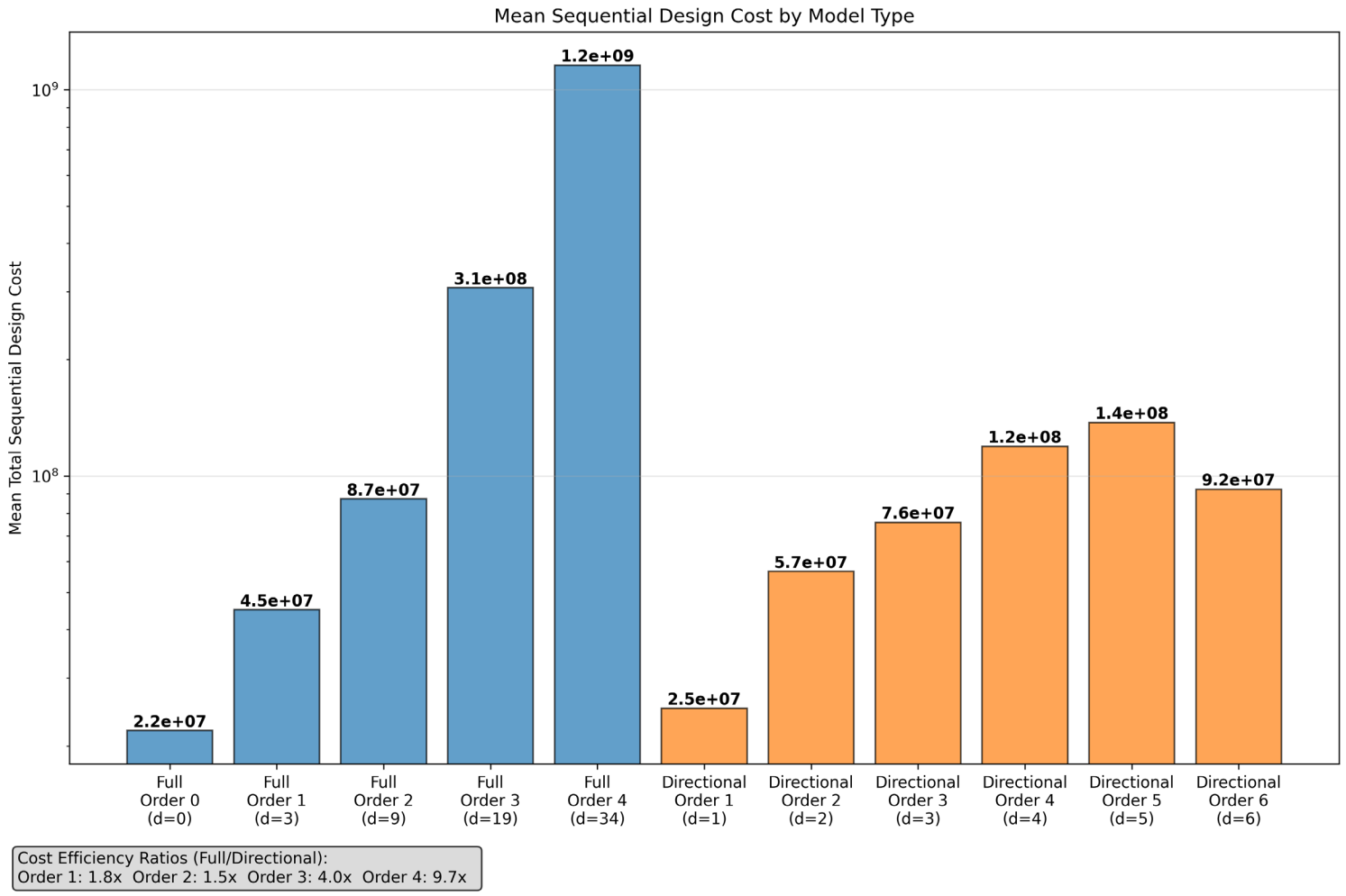
✅ A significant reduction in sample points needed to reach convergence.
✅ When accounting for the cost, first, second, and third order derivatives show a more favorable convergence trend when compared to no derivatives

Derivative Enhanced GP ECL Case Study



❌Using relative error in failure volume, **directional derivative enhanced gaussian process perform worse** than the full derivatives in terms of cost of function evaluations.

Derivative Enhanced GP ECL Case Study



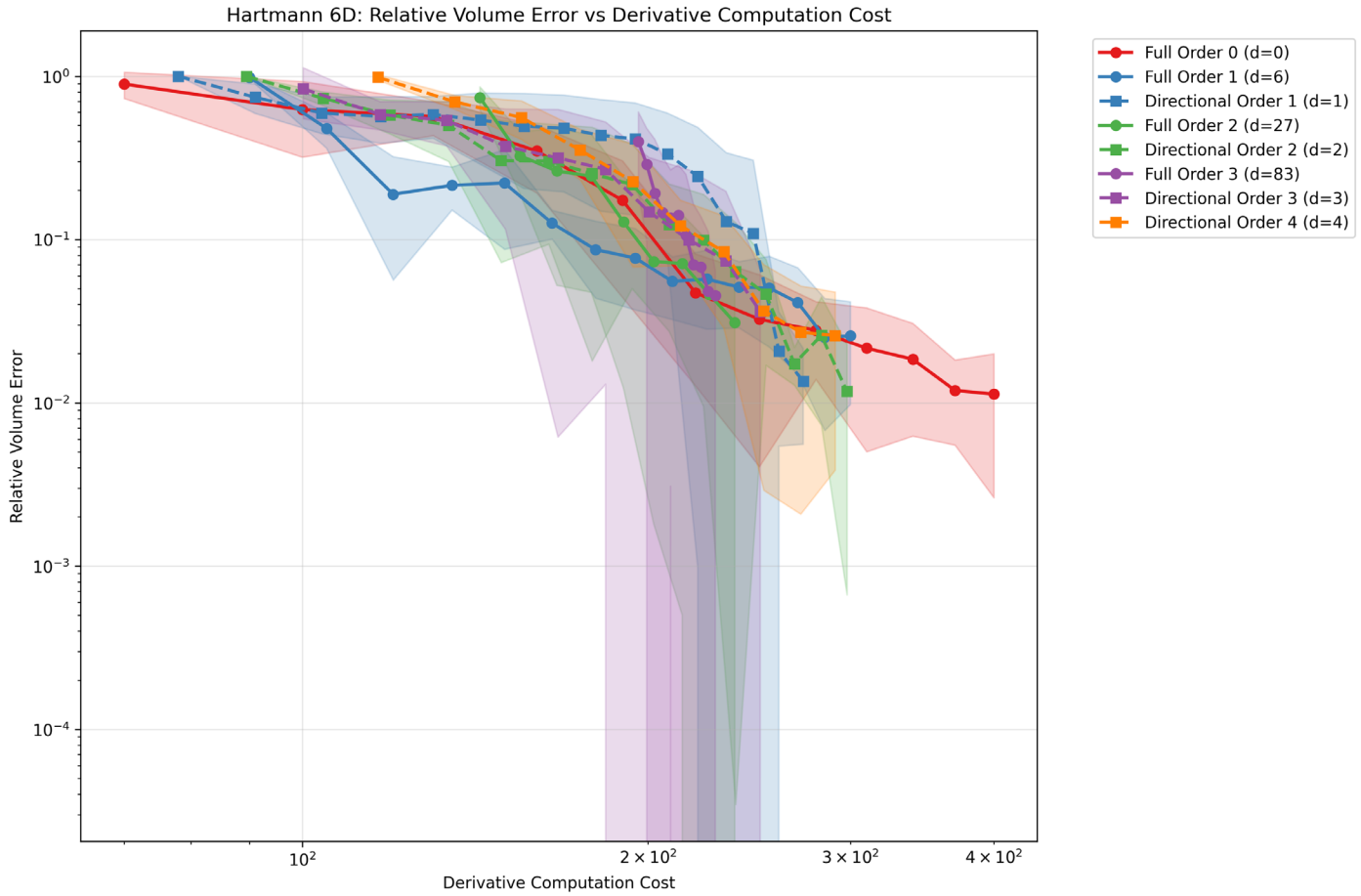
✅ When looking at the total training cost, directional derivatives far outperform the full derivative counterparts

Using first order directional derivatives:

Increases the training cost when **compared to the no derivative case** by 1.13 ×

Reduces the training cost when **compared to first order derivatives** by 1.8 ×

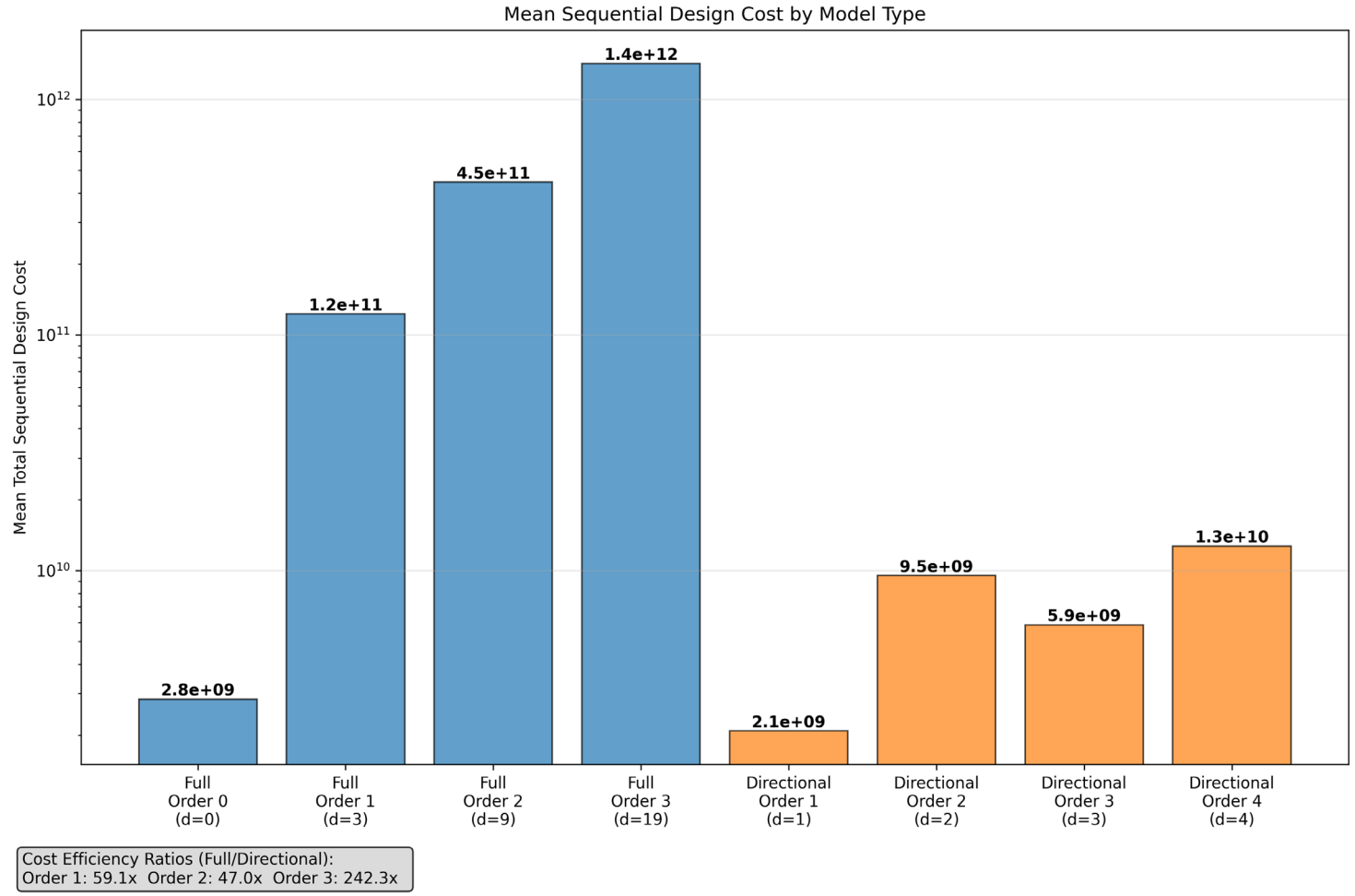
Derivative Enhanced GP ECL Case Study



✅ Using relative error in failure volume, **directional derivative enhanced gaussian processes outperform gradient enhanced gaussian process** in terms of cost of function evaluations.

❌ The convergence trend of **full higher order derivative enhanced gaussian processes outperforms directional derivative enhanced gaussian processes**

Derivative Enhanced GP ECL Case Study



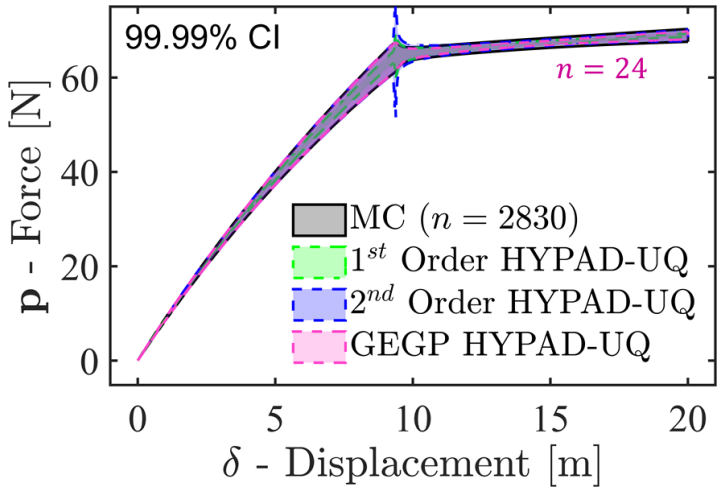
Using first order directional derivatives

Reduces the training cost when **compared to the no derivative case** by $1.33\times$

Reduces the training cost when **compared to first order derivatives** by $59\times$

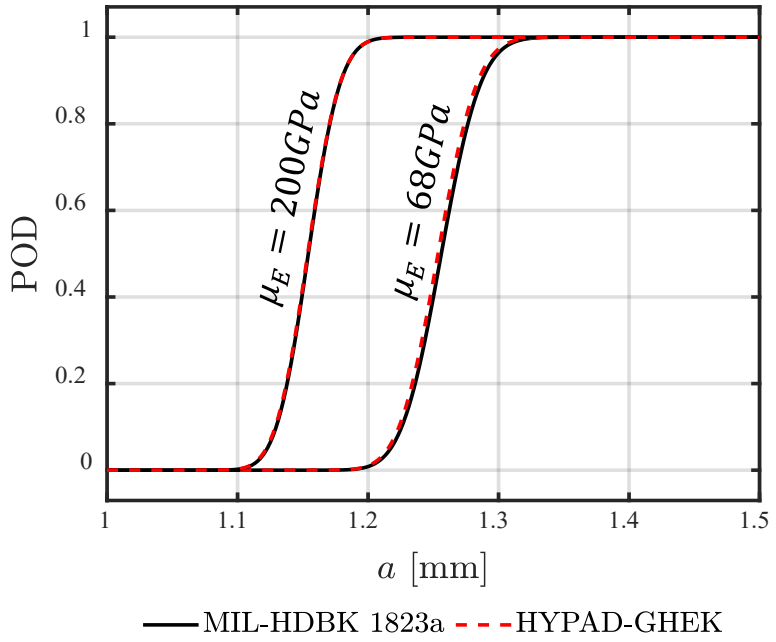
Current Applications

- **Post Buckling Behavior**
  The use of DEGP-based uncertainty quantification (UQ) to model the post-buckling response of architected materials.



- **Probability of Detection (PoD)**
  DEGP models are also used to estimate the Probability of Detection in structural health monitoring.

Method	Simulations	Time per Simulation	Run in Series
MIL-HDBK	2252	21 min	9450 min
HYPAD-GHEK	**20**	150 min	**3000 min**



Key Takeaways: Access to second-order derivatives ***reduces significantly*** the required number of samples

Conclusions:

Key Technical Achievements:

✅ **Directional derivatives drastically reduce computational complexity**

- **3D**: Full derivatives scale as 3→9→19 vs Directional 1→2→3 for orders 1-3
- **6D**: Full derivatives scale as 6→21→56 vs Directional 1→2→3 for orders 1-3
- **Exponential advantage in higher dimensions**

✅ **Real computational cost analysis reveals practical benefits**

- **Directional derivatives: are cheaper to obtain when compared to their full counterparts.**
- **The use of directional derivatives enables the encoding of higher order derivative information in a computationally tractable way.**

Conclusions:

Key Trade-off Understanding:

⚖️ **More function evaluations vs. dramatically reduced training costs**

- Directional derivatives may require **more sequential design iterations**
- But each GP training step has **orders of magnitude lower computational cost**

Practical Impact & Recommendations:

1. Adopt directional derivatives for high-dimensional reliability analysis ($d \geq 4$)
2. Use 1st – 3rd order derivatives for optimal cost-performance trade-off
3. Implement optimized direction selection using IER maximization
4. Consider full derivatives only for very low-dimensional problems ($d \leq 3$)

References

[1] L. Chen, H. Qiu, L. Gao, C. Jiang, and Z. Yang, “A Screening-Based Gradient-Enhanced Kriging Modeling Method for High-Dimensional Problems,” Applied Mathematical Modelling, vol. 69, pp. 15–31, 2019. doi: 10.1016/j.apm.2018.11.048

[2] Z.-H. Han, Y. Zhang, C.-X. Song, and K.-S. Zhang, “Weighted Gradient-Enhanced Kriging for High-Dimensional Surrogate Modeling and Design Optimization,” AIAA Journal, vol. 55, no. 12, pp. 4330–4346, 2017. doi: 10.2514/1.J055842

[3] D. Eriksson, K. Dong, E. H. Lee, D. Bindel, and A. G. Wilson, "Scaling Gaussian Process Regression with Derivatives," *arXiv preprint arXiv:1810.12283*, Oct. 2018. DOI: 10.48550/arXiv.1810.12283.

[4] S. Ulaganathan, I. Couckuyt, T. Dhaene, J. Degroote, and E. Laermans, "Performance study of gradient-enhanced Kriging," *Engineering with Computers*, vol. 32, no. 1, pp. 15–34, Jan. 2016, doi: 10.1007/s00366-015-0397-y

[5] W. Liu and S. Batill, "Gradient-Enhanced Response Surface Approximations Using Kriging Models," presented at the 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Atlanta, GA, Sep. 4–6, 2002, doi: 10.2514/6.2002-5456.

[6] W. Yamazaki, M. Rumpfkeil, and D. Mavriplis, "Design Optimization Utilizing Gradient/Hessian Enhanced Surrogate Model," presented at the 28th AIAA Applied Aerodynamics Conference, Chicago, IL, Jun. 28–Jul. 1, 2010, doi: 10.2514/6.2010-4363.

[7] D. A. Cole, R. B. Gramacy, J. E. Warner, G. F. Bomarito, P. E. Leser, and W. P. Leser, ‘Entropy-based adaptive design for contour finding and estimating reliability’, *arXiv [stat.ME]*, 2021.

[8] M. A. Cano, "Order Truncated Imaginary Algebra for Computation of Multivariable High-Order Derivatives in Finite Element Analysis." Order No. 29992936, Universidad EAFIT, Colombia, 2020.

Acknowledgements

- CONNECT- the CONsortium on Nuclear sECurity Technologies-Renewal, National Nuclear Security Administration, Sr. Personnel (E. Sooby PI), October 1, 2022 – September 30, 2027, $4,999,995, National Nuclear Security Agency (NNSA), DE-NA0004107

- Educate and train the best next-generation professionals with strong backgrounds in nuclear science, fissionable fuels fabrication and processing, nuclear materials characterization, nuclear forensic signatures, nuclear technology, and data and visual analytics.

- NASA- National Aeronautics and Space Administration, Langley Airforce Research Center (LaRC), Durability, Damage Tolerance, and Reliability Branch (DDTRB) WBS: 869021.05.23.02.75
    - *James Warner*
    - *Patrick Leser*
    - *Geoffrey Bomarito*

Questions?