Investigating and Enhancing Gaussian Process-based Fouling Detection Systems

Aayush Kucheria, University of Helsinki (2023)

Abstract

This report presents an investigation into enhancing the computational efficiency of a Gaussian Process (GP)-based fouling detection system for industrial pipes. The original system, implemented in Stan, successfully combined ultrasonic guided waves (UGWs) with GP modeling to detect and reconstruct fouling distributions. Our work explored the feasibility of porting this implementation to GPyTorch, a modern probabilistic programming framework, with the aim of improving computational performance. However, detailed architectural analysis revealed incompatibilities between the system's specialized requirements and standardized framework constraints. This discovery led to a strategic pivot, focusing instead on enhancing the original Stan implementation through neural network approaches.

The project's key contributions include a system architecture analysis, development of monitoring systems, and insights into implementing complex probabilistic models.

1. Introduction

Non-destructive evaluation (NDE) of industrial equipment is crucial for preventing costly repairs and maintaining operational efficiency. Recent work by Iablonskyi et al. [1] demonstrated a novel approach for fouling detection in closed systems using ultrasonic guided waves (UGWs) combined with Gaussian Process (GP) modeling. Their method, implemented in Stan [2], showed promising results in reconstructing fouling distributions with relatively few sensors. However, the computational demands of this implementation presented opportunities for potential efficiency improvements.

This report describes work undertaken during a summer internship at the University of Helsinki aimed at enhancing the computational efficiency of this GP-based fouling detection system. The project began with an investigation into modern probabilistic programming frameworks, particularly GPyTorch [3], as alternatives to the original Stan implementation. However, detailed analysis revealed fundamental architectural incompatibilities between the custom implementation requirements and GPyTorch's standardized structure. This discovery led to a pivot in the project's direction, focusing instead on enhancing the existing Stan implementation through neural network approaches.

The primary contributions of this work include:

1. Deep analysis of the system's architectural requirements and constraints
2. Identification of fundamental limitations in porting to standardized frameworks
3. Development of an enhanced Stan implementation incorporating neural network elements
4. Insights into the trade-offs between framework standardization and implementation flexibility

The remainder of this report details our investigation process, architectural findings, and the subsequent development of improvements to the original implementation. We conclude with a discussion of insights gained and potential directions for future research in this domain.

2. System Analysis

The first phase of the project involved developing an understanding of the fouling detection system's components and their interactions. This analysis proved important for later identifying architectural constraints.

2.1 Physical System Architecture

The system [1] employs ultrasonic guided waves (UGWs) for fouling detection through a helical path approach. Unlike conventional ultrasonic waves, UGWs are guided along specific paths by the geometric and material properties of the pipe structure. This enables:

- Long-distance propagation along the pipe's structure
- Accumulation of structural information during propagation
- Detection of both inner and outer structural flaws

The system's unique use of helical paths allows a single sensor pair to generate multiple measurement paths, providing dense structural information without requiring additional sensors. When these paths intersect fouled areas, they generate distinct features through:

- Changes in wave attenuation
- Phase shifts in the signal
- Modifications to wave propagation properties

As illustrated in Figure 1, the system's design leverages the cylindrical geometry of industrial pipes to maximize information gathering from minimal sensor deployment. The helical paths (shown as blue dots) provide multiple "views" of potential fouling regions from different angles. When a wave path intersects with fouling (shown in red), the wave's properties are altered, providing valuable diagnostic information. This clever use of geometry enables comprehensive pipe monitoring with remarkably few sensors, making the system both cost-effective and practical for industrial deployment.

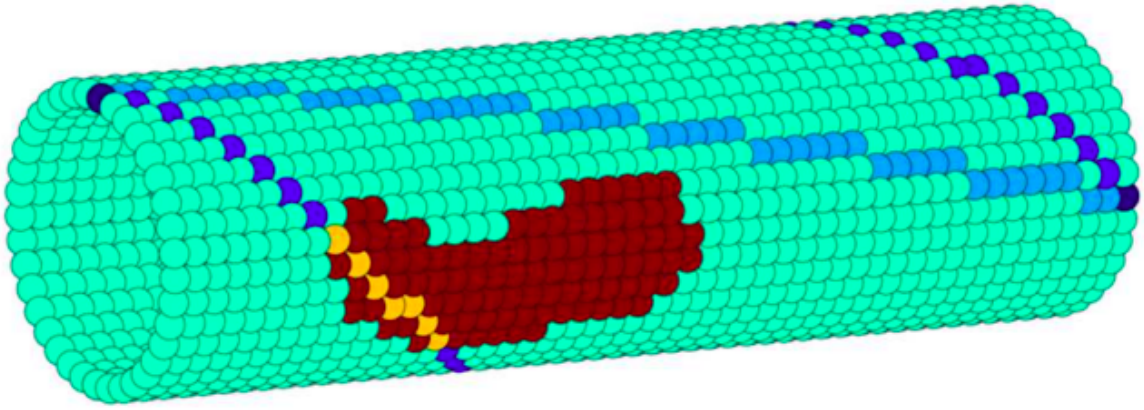

Figure 1: Schematic representation of the fouling detection system, taken from [1]. The cylindrical pipe structure (green) shows a fouling region (red) and helical wave paths (blue dots) between ultrasonic transducers. This configuration allows multiple measurement paths from a single sensor pair, enabling dense structural information collection without additional sensors.

2.2 Mathematical Framework

The mathematical framework combines Gaussian Process modeling with specific domain constraints. Two key assumptions about fouling distributions are encoded in the system:

1. Spatial Smoothness: Fouling varies continuously across spatial coordinates
2. Physical Non-negativity: Fouling cannot have negative values

These constraints are implemented through:

- A dual-kernel approach using:
    - Periodic kernel for the circumferential axis
    - Gaussian kernel for the longitudinal axis
- Non-linear transformation using a sigmoid function to ensure non-negativity [4].

2.3 Computational Architecture

The original Stan implementation integrated several complex components:

- Custom integral observations over helical paths
- Non-linear transformations of the GP outputs
- Multiple interacting kernel types
- Specialized parameter distributions

This integration relied heavily on Stan's flexibility in handling custom probabilistic models, particularly in areas of parameter initialization and transformation, custom observation models, and non-standard distribution combinations.

Understanding these architectural elements proved crucial for identifying why a direct port to more standardized frameworks would be challenging, ultimately leading to our revised approach.

3. Framework Investigation and Implementation Pivot

3.1 Initial GPyTorch Investigation

The project initially focused on porting the Stan [2] implementation to GPyTorch [3], a modern probabilistic programming framework known for its computational efficiency. However, detailed investigation revealed several fundamental incompatibilities:

1. Structural Constraints:
    a. GPyTorch's standardized architecture couldn't readily accommodate the custom integral observations required for helical path measurements
    b. The framework's handling of non-linear transformations didn't align with our specific needs
    c. Multiple kernel types needed specialized handling not available in the standard implementation
2. Parameter Handling:
    a. The original Stan implementation used inverse gamma distributions for certain parameters
    b. These distributions were discovered to be more Stan-specific implementation choices rather than fundamental requirements
    c. Attempts to simplify using uniform distributions revealed deeper architectural mismatches

3.2 Implementation Pivot

Recognition of these incompatibilities led to a strategic pivot. Rather than forcing GPyTorch to accommodate our specialized needs, we focused on enhancing the original Stan implementation through neural network approaches. This decision was driven by:

- Need to maintain the custom integral observation structure
- Requirement for flexible parameter transformations
- Desire to preserve the probabilistic nature of the model
- Potential for performance improvements within the existing framework

3.3 Enhanced Stan Implementation

The revised approach focused on implementing neural network elements within Stan, resulting in:

- Improved computational efficiency
- Maintained flexibility for custom observations
- Preserved probabilistic modeling capabilities
- Enhanced parameter optimization

4. Results and Analysis

4.1 Development of Monitoring Systems

A significant contribution of this work was the implementation of monitoring systems to track model behavior and performance. These systems were helpful for understanding:

- Parameter distributions and ranges during training
- Evolution of key variables throughout optimization
- Convergence patterns across different model configurations
- Interaction effects between model components

4.2 Parameter Optimization Findings

Investigation of parameter behavior revealed several key insights:

1. Distribution Effects:
    a. Simplified parameter distributions (uniform rather than inverse gamma) proved adequate
    b. Parameter ranges significantly impacted training stability
    c. Careful initialization was crucial for model performance
2. Optimization Dynamics:
    a. Training stability was heavily influenced by parameter interactions
    b. Balance between model complexity and trainability required careful management
    c. Monitoring systems proved useful for understanding convergence behavior

Figure 2 demonstrates examples of fouling reconstruction results. The left column shows the ground truth (generated fouling patterns), while the right column shows the system's reconstructed fouling distributions. Note how the reconstruction maintains the general shape and location of the fouling while exhibiting some expected smoothing due to the GP's smoothness constraints.

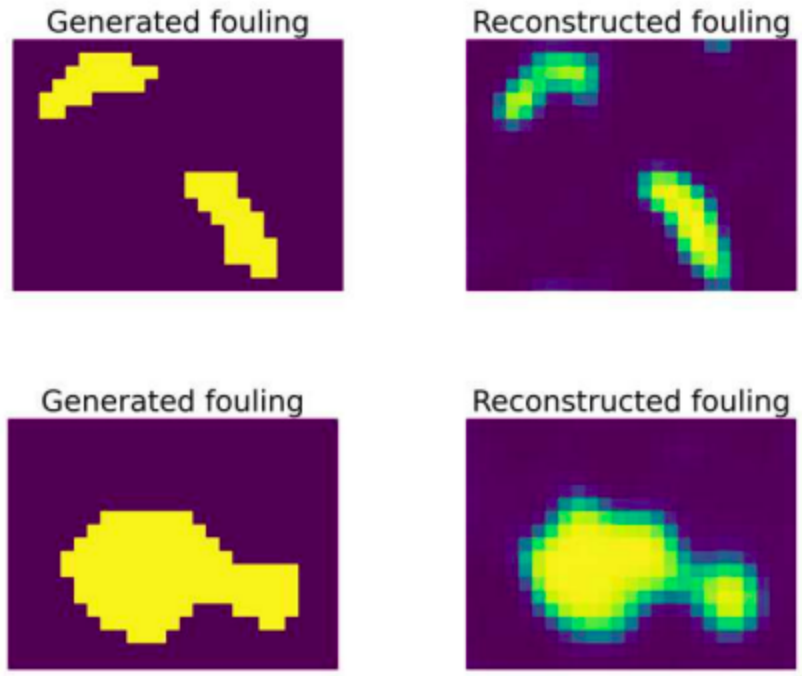

Figure 2: Comparison of generated fouling patterns (left column) with their reconstructed distributions (right column), taken from [1]. Top row shows the system's ability to reconstruct multiple distinct fouling regions, while the bottom row demonstrates reconstruction of more complex, continuous fouling patterns. The smooth transitions in the reconstructed images reflect the GP model's built-in smoothness assumptions, while still maintaining accurate localization of fouling regions.

5. Conclusions and Future Directions

5.1 Summary of Contributions

This project began with the aim of improving computational efficiency in a GP-based fouling detection system through modern frameworks, but evolved into a deeper investigation of system architecture and implementation approaches. Our main contributions include:

- Development of monitoring systems for tracking model behavior
- Identification of architectural constraints in standard frameworks
- Enhancement of the original Stan implementation through neural network approaches
- Insights into implementing specialized probabilistic models

5.2 Key Findings

Our investigation revealed several important insights:

1. Framework Compatibility: The original Stan implementation's flexibility, while computationally intensive, proved crucial for handling the system's specialized needs.
2. Implementation Trade-offs: The attempted GPyTorch port revealed important trade-offs between framework standardization and implementation flexibility.
3. Enhancement Strategy: Improving within an existing framework can sometimes be more effective than switching to a seemingly more efficient alternative.

5.3 Future Directions

Several directions for future work emerged from this investigation:

- Further optimization of the neural network approach within Stan
- Extension of the monitoring systems for deeper model insights
- Development of specialized frameworks for complex probabilistic models
- Integration with real-time industrial monitoring systems

5.4 Final Remarks

The project demonstrated that improving complex systems requires looking beyond simple framework transitions to understand fundamental architectural requirements. While our findings are specific to fouling detection, the insights about implementing specialized probabilistic models have broader implications for similar systems in other domains.

6. Acknowledgements

This work was conducted in a summer internship in the Multi-source probabilistic inference (MUPI) research group at the University of Helsinki. It was done under the project “AI for Ultrasonics” [5], with the supervision of Arto Klami and Denys Iablonskyi.

7. References

[1] Iablonskyi, D., Wei, H., Klami, A., Salmi, A., & Hæggström, E. "Reconstruction of Fouling Distribution from Aggregate Observations." 2022 IEEE International Ultrasonics Symposium (IUS), IEEE (2022)

[2] B. Carpenter, A. Gelman, M. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, and A. Riddell, “Stan: A Probabilistic Programming Language”, Journal of Statistical Software, 76(1), 1–32 (2017).

[3] Gardner, Jacob R., Geoff Pleiss, David Bindel, Kilian Q. Weinberger, and Andrew Gordon Wilson. "GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration." In Advances in Neural Information Processing Systems (2018)

[4] T. Sillanpää, T. Rauhala, J. Mäkinen, C. Rajani, K. Longi, A. Klami, A. Salmi, and Edward Hæggström, “Ultrasonic fouling detector powered by machine learning”, 2019 IEEE International Ultrasonics Symposium (IUS), 1639-1642 (2019).

[5] Multi-source Probabilistic Inference Group. "Research - Multi-source Probabilistic Inference." University of Helsinki. https://www.helsinki.fi/en/researchgroups/multi-source-probabilistic-inference/research (accessed November 2024)