

# Probabilistic Ancestral Inference from Incomplete Genetic Data

Makenna Worley

January 2026

## Abstract

This project investigates the efficacy of probabilistic machine-learning architectures in reconstructing latent states within high-dimensional, stochastic datasets. Using a multi-generational simulation framework as a high-fidelity data engine, the study evaluates how Bayesian inference, Hidden Markov Models (HMMs), and Graph Neural Networks (GNNs) recover missing features from intentionally degraded data. The system generates structured, hierarchical datasets with complete ground truth, enabling controlled benchmarking under varying levels of data masking.

Each simulation run produces a reproducible artifact bundle containing ground-truth matrices, masked observations, hierarchical ancestry representations, and structured run metadata. This metadata supports deterministic dataset reconstruction for fair benchmarking and comparative evaluation. The research analyzes the trade-offs between computational efficiency and inference precision while tracking model calibration and robustness. The final deliverable is a containerized, interactive web application that enables reproducible dataset generation and visualization-driven evaluation of probabilistic inference performance in sparse or incomplete hierarchical domains.

## 1 Introduction

The ability to reason under uncertainty remains a central challenge in computational modeling, particularly when dealing with high-dimensional datasets where hierarchical dependencies are obscured by missing or degraded observations. While data sparsity often limits the effectiveness of traditional analytical methods, probabilistic machine-learning models provide a principled framework for inferring hidden variables. This project treats rule-based stochastic simulation as a rigorous testing ground for algorithmic robustness, focusing on the mathematical challenge of recovering latent states from controlled, partially observed data.

Rather than relying on externally collected datasets, this work implements a simulation-driven approach that generates structured hierarchical data with complete ground truth.

Every internal state in the system is known at generation time, enabling direct quantitative validation of reconstruction accuracy. The generated datasets are systematically masked at configurable rates to evaluate how different probabilistic models—including Bayesian inference, Hidden Markov Models (HMMs), and Graph Neural Networks (GNNs)—perform under increasing levels of uncertainty. By using a discrete, rule-governed simulation engine, the data maintains clear structural dependencies, allowing performance metrics to reflect model behavior rather than ambiguity in the data source.

The technical foundation of this work is a reproducible data generation pipeline that preserves full run configurations and deterministic random seeds. Each simulation produces a structured metadata record that enables exact dataset reconstruction, ensuring that benchmarking comparisons are not confounded by stochastic variation in the generator. This reproducibility guarantees that evaluation metrics—such as precision, recall, and calibration—measure architectural strengths and inference behavior rather than noise. An interactive web-based interface orchestrates dataset generation and model evaluation, supporting visualization-driven analysis of computational cost versus inference precision across probabilistic frameworks.

The core objectives are:

- To design and implement a simulation and inference system that models latent dependencies within hierarchical stochastic data.
- To implement, train, and evaluate probabilistic and machine-learning models (Bayesian, HMM, and GNN) for high-dimensional state reconstruction under controlled masking.
- To develop an interactive web application for visualization, evaluation, and reproducible comparison of model performance across varying data degradation thresholds.

If successful, this project will establish a reproducible methodology for state reconstruction under uncertainty, offering a computational framework applicable to any domain that relies on incomplete or partially observed hierarchical stochastic data.

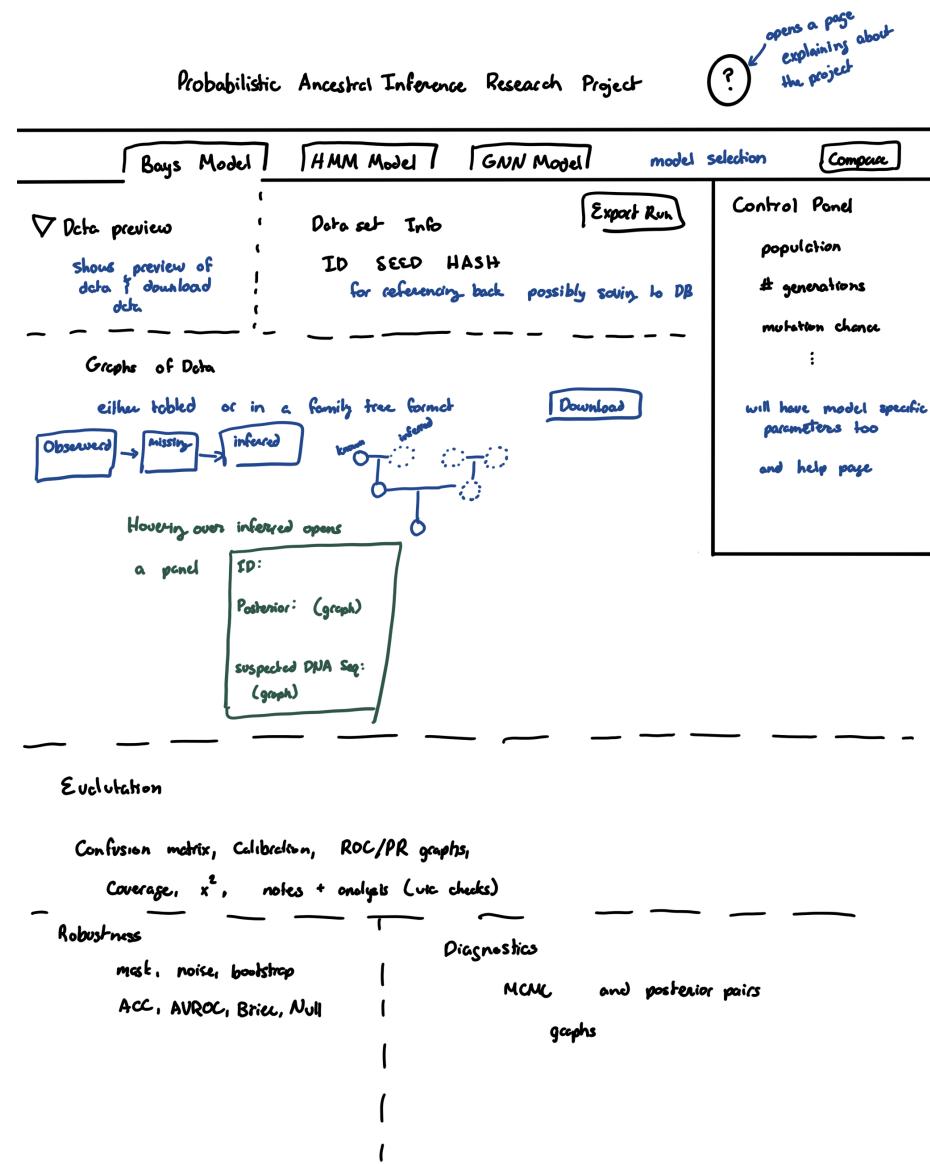


Figure 1: Design language inspired by prior interactive visualization work.

## 2 Background

### 2.1 Scientific Context

Inferring latent states within high-dimensional, stochastic systems is a persistent challenge in machine learning, particularly when datasets exhibit complex hierarchical dependencies and incomplete observations. Probabilistic architectures such as Bayesian inference, Hidden Markov Models (HMMs), and Graph Neural Networks (GNNs) offer distinct strategies for modeling uncertainty, structural priors, and state transitions across discrete features or time steps.

Table 1: Comparison of Probabilistic Inference Architectures for Stochastic Data

Method	Strengths	Limitations	Best Used When
<b>Bayesian Inference</b>	Highly interpretable; incorporates structural priors; models uncertainty explicitly.	High computational cost; slower convergence for large datasets.	Domain-specific knowledge is available and interpretability is critical.
<b>Hidden Markov Model (HMM)</b>	Efficient for sequential linkage; strong theoretical foundation for state transitions.	Assumes simplified conditional independence structures; limited in modeling complex non-linear dependencies.	Modeling sequential transitions or temporally linked discrete features.
<b>Graph-Based Model (GNN)</b>	Captures multi-way relationships and complex hierarchical structures; highly expressive and scalable.	Requires larger training sets and intensive hyperparameter tuning.	Modeling highly connected networks or population-level dependencies.

Existing inference approaches often struggle with missingness in sparse datasets. Traditional maximum-likelihood methods assume near-complete data or well-defined priors, making them sensitive to high masking rates. Similarly, sequential modeling frameworks can degrade when internal nodes are unobserved or sparsely sampled. These limitations motivate the development of benchmarking environments that explicitly evaluate robustness under controlled degradation.

Unlike empirical studies constrained by noisy real-world data, this project employs a stochastic simulation engine to create a fully controlled experimental setting. The system generates structured hierarchical datasets with complete ground truth, preserving both the underlying dependency structure and the full parameter configuration used during generation. This enables direct, quantitative evaluation of reconstruction performance without the confounding variability inherent to empirical datasets.

The simulation pipeline leverages the `msprime` framework to model multi-generational ancestry and mutation processes under discrete Mendelian rules. The output is a diploid dosage representation in which features correspond to independent loci and entries represent discrete states. From this complete representation, controlled masking is applied at configurable rates to produce partially observed datasets. Because the full generative configuration and deterministic seeds are preserved, each dataset can be reconstructed exactly, ensuring fair and reproducible benchmarking of probabilistic inference architectures. This modular design ensures that the primary focus remains on **algorithmic inference accuracy** rather than domain-specific modeling artifacts.

## 2.2 Technological Context

Recent advances in simulation frameworks and probabilistic programming make rigorous benchmarking of machine learning models under uncertainty both feasible and efficient. The `msprime` library serves as a high-performance stochastic engine, enabling the generation of large-scale, multi-generational datasets with discrete, rule-based dependencies. This framework provides fine-grained control over the underlying generative process, allowing the simulation of complex relational structures while maintaining strict scalability.

The data generation pipeline is engineered for deterministic reproducibility. Each simulation run preserves its full parameter configuration and random seed state, ensuring that both the complete ground-truth dataset and its masked counterpart can be reconstructed exactly. This controlled replay capability eliminates stochastic drift between experimental runs and provides a stable foundation for comparative model benchmarking.

For statistical modeling, the project employs PyMC (with the PyTensor backend) to implement Bayesian architectures that model uncertainty explicitly through structured priors. Sequential dependencies within the data are addressed using Hidden Markov Models (HMMs) implemented through libraries such as `pomegranate` or `hmmlearn`, which provide flexible state-transition representations. The graph-based approach leverages PyTorch Geometric to represent hierarchical relationships as nodes and edges within a learned graph structure, enabling the capture of multi-way dependencies across individuals and features.

The system architecture follows modern web and data engineering practices. Dataset generation and experiment orchestration are handled through a FastAPI backend, while results and configuration workflows are exposed through a custom React interface. This interactive layer enables parameter selection, experiment execution, and visualization of evaluation metrics such as reconstruction accuracy and calibration. The entire environment is containerized using Docker, ensuring modular deployment and cross-platform reproducibility.

These tool choices were selected for their open-source accessibility, active research communities, and demonstrated performance in simulation-driven AI research.

I bring direct technical experience to this stack:

- **Frontend Development:** Experience with `React` and `ReactFlow` from the CircuitCraft: a summer research project, where I contributed to an interactive circuit-drawing application using `ReactFlow`.
- **Backend Engineering:** Previous implementation of `FastAPI` in the MVP of my differential equations project, integrating mathematical modeling with a web-based backend.
- **Data Systems:** Ongoing validation of a lightweight relational storage system for managing simulation metadata and experiment runs.

Through this project, I aim to deepen my expertise in probabilistic modeling and simulation design while integrating full-stack engineering practices. The work bridges reproducible data engineering with applied machine learning, focusing on the practical challenges of interpretability, calibration, and recovery performance in sparse hierarchical data regimes.

## 3 Proposed Work

### 3.1 Proof of Concept

#### 3.1.1 Overview

The core proof of concept (PoC) for this project will be a functional pipeline capable of reconstructing latent states within hierarchical stochastic datasets. Using rule-based simulations to generate ground-truth data, the PoC will demonstrate that a Bayesian inference engine can accurately recover features intentionally removed via a controlled masking process. At minimum, the PoC will validate that the model can handle a specific "masking rate" threshold and visualize the reconstruction accuracy and posterior uncertainty through a web-based interface. This deliverable establishes the technical feasibility of using probabilistic architectures to solve state-recovery problems in sparse data regimes.

#### 3.1.2 Specific Tasks

1. Generate Synthetic Datasets: Use `msprime` to simulate high-fidelity stochastic data with discrete hierarchical dependencies.

2. Implement Masking Framework: Develop a systematic process to mask data entries (e.g., 20% masking rate) to emulate incomplete observations while preserving "truth" for validation.
3. Develop Bayesian Inference Engine: Train a baseline Bayesian model using PyMC to infer masked values based on hierarchical priors.
4. Dataset Stratification: Produce independent training, validation, and testing datasets to ensure statistical robustness and prevent overfitting.
5. Deploy Lightweight Dashboard: Build a **Streamlit** interface for real-time visualization of model performance and data degradation metrics.
6. Quantify Recovery Performance: Evaluate accuracy using statistical metrics, including chi-square tests and likelihood-based confidence intervals.

### 3.1.3 Rationale

The PoC focuses on the mathematical recovery of masked data points rather than domain-specific predictions. By working in a controlled simulation environment, we can perform quantitative validation against an absolute ground truth—a process often impossible with empirical data. Proving that a Bayesian model can successfully resolve these latent states validates the core logic of the inference framework before expanding to more complex architectures like HMMs or GNNs.

### 3.1.4 Deployment

The PoC has been deployed as a local **Streamlit** application to facilitate rapid demonstration and iterative testing. Simulation outputs, including truth and observed matrices, are stored alongside structured run metadata to enable deterministic replay capability. This local deployment provides a streamlined interface for loading diverse datasets, executing inference runs, and monitoring reconstruction performance without the overhead of distributed infrastructure. It serves as the validated foundation upon which the final full-stack web application is being developed.

## 3.2 Full Project Development

### 3.2.1 Specific Tasks

1. Scale Simulation Pipeline: Extend the existing `msprime`-based engine to support larger dataset dimensions and more complex hierarchical parameters.
2. Implement Comparative Architectures: Develop and tune multiple probabilistic models—Bayesian, HMM-based, and Graph Neural Networks (GNN)—to reconstruct latent states from masked data.

3. Systematic Robustness Testing: Regenerate training, validation, and testing datasets under varying masking-rate conditions to benchmark model degradation.
4. Expand Model-Serving Backend: Build upon the existing `FastAPI` backend to manage model inference, dataset retrieval, and structured run metadata storage.
5. Enhance Interactive Analytics Dashboard: Extend the `React` frontend for comparative performance visualization, utilizing dynamic graphing for accuracy and calibration metrics.
6. Advanced Statistical Validation: Validate reconstruction performance using chi-square, likelihood-ratio, and calibration metrics across all model types.
7. Containerization and Deployment: Finalize and optimize the `Docker`-based deployment to ensure full environment parity and research reproducibility.

### 3.2.2 Rationale

Stochastic datasets are frequently incomplete, yet probabilistic models are specifically designed to reason under uncertainty. By operating within a simulation-controlled environment, the project achieves absolute control over generative structure, noise, and masking parameters—enabling rigorous, reproducible experimentation. While biological inheritance rules provide the logical backbone of the simulation, the primary objective is evaluating computational robustness in a scalable inference framework. Independent dataset partitions mitigate overfitting, and the `FastAPI`–`React` architecture ensures the system remains maintainable, modular, and extensible beyond a localized prototype.

## 3.3 Plan of Work

1. Data Pipeline Engineering: Refine and scale the existing `msprime`-based pipeline for generating ground-truth matrices. Maintain deterministic seed-based replay to ensure any stochastic run can be reconstructed during evaluation.
2. Model Benchmarking: Implement and optimize the three core model types—Bayesian, HMM, and Graph-based. Focus on runtime efficiency for high-dimensional matrices and systematic documentation of architectural hyperparameters.
3. Robustness Profiling: Introduce controlled noise and variable masking thresholds to the datasets. Record error bounds and performance metrics (Precision, Recall, Calibration) to establish the limits of each model’s reconstruction ability.
4. Full-Stack Integration: Continue integration between `FastAPI` endpoints and the `React` dashboard to enable dynamic querying of model results and visualization of confidence intervals.

5. Optimization and Deployment: Identify and resolve performance bottlenecks in the inference engine. Finalize the `Docker` configuration and documentation to ensure the system is fully reproducible by other researchers.

**Absolute Minimum:** A functional stochastic data engine using `msprime` to generate truth and masked datasets, integrated with a baseline Bayesian inference model and a `Streamlit` dashboard for reporting recovery statistics.

**Expected:** All features of the Absolute Minimum plus a robust validation pipeline and a full-stack `FastAPI` and `React` dashboard. This tier includes comparative analysis between Bayesian inference and Hidden Markov Models (HMM).

**Aspirational:** All features of the Expected tier with the addition of a Graph Neural Network (GNN) model to evaluate how graph-based learning handles multi-way dependencies. This includes a comprehensive statistical comparison of reconstruction efficiency, accuracy, and computational overhead across all three probabilistic models.

### 3.4 Preliminary Work

Initial research and engineering have established a high-fidelity stochastic data generation pipeline capable of producing discrete dosage matrices from rule-based simulations. This work validated a deterministic replay strategy using preserved run configurations and random seed states to ensure exact reproducibility of experimental datasets. The engineering stack—including `Streamlit` for the initial PoC and the evolving `FastAPI`–`React` architecture—has been validated as a scalable foundation for probabilistic benchmarking.

## 4 Timeline

Table 2: Project Timeline

Phase	Dates	Time est.	Focus	Deliverables
<b>Phase 0: Data Engine &amp; Reproducibility</b>	Oct 2025	20 hrs	Implement multi-generational stochastic simulation pipeline. Establish deterministic seed-based replay and structured run metadata storage.	Reproducible data engine capable of generating ground-truth and masked hierarchical datasets.
<b>Phase 1: Proof of Concept (Streamlit MVP)</b>	Nov 2025	70 hrs	Implement baseline Bayesian model using PyMC. Validate reconstruction accuracy under controlled masking rates and develop initial Streamlit dashboard.	Functional PoC achieving reliable masked-state recovery with interactive visualization of uncertainty and accuracy metrics.
<b>Phase 2: Comparative Expansion</b>	Dec–Jan 2026	90 hrs	Implement HMM-based inference and perform systematic benchmarking across variable masking rates. Introduce calibration and likelihood-based evaluation metrics.	Comparative analysis suite with documented performance metrics for Bayesian and HMM architectures.
<b>Phase 3: Full-Stack Web Application</b>	Feb–Mar 2026	70 hrs	Expand FastAPI model-serving endpoints. Develop and integrate React frontend for interactive dataset configuration and multi-model comparison.	Containerized full-stack dashboard supporting dynamic experiment execution and visualization-driven benchmarking.
<b>Phase 4: Optimization &amp; Code Freeze</b>	Apr 2026	40 hrs	Finalize Docker deployment. Optimize inference runtime and complete reproducibility documentation. Conduct final robustness profiling across masking thresholds.	Production-ready, fully containerized system with comprehensive performance report and public documentation.

**Total estimated effort:** approximately 290 hours.

## 5 Evaluation

Evaluation will assess both quantitative reconstruction performance and system-level robustness across controlled masking regimes. The following dimensions define the evaluation framework:

- **Reconstruction Accuracy:** Measure the divergence between predicted values and the known ground-truth dataset using precision, recall, F1-score, and likelihood-based metrics. Statistical significance will be validated using chi-square and likelihood-ratio tests to ensure inferred latent states are mathematically consistent with observed structure.
- **Model Calibration:** Evaluate the reliability of probabilistic outputs by comparing predicted posterior confidence intervals against empirical recovery rates. Calibration curves and error distributions will be analyzed to detect overconfidence or underconfidence in high-uncertainty regimes.
- **System Performance:** Benchmark runtime efficiency and memory usage as dataset dimensionality increases (e.g., number of loci and simulated population size), quantifying computational scalability across architectures.
- **Computational Robustness:** Identify degradation thresholds by measuring reconstruction accuracy across a spectrum of controlled masking rates (e.g., 10%–80% missingness). Breakpoints will be analyzed to determine the stability limits of each probabilistic architecture.
- **Usability and Transparency:** Evaluate whether the React dashboard clearly communicates uncertainty, calibration behavior, and reconstruction metrics. The system will preserve structured run configurations to ensure experiment auditability and reproducibility.

### Ethical Considerations and Interpretability:

- **Data Privacy:** The exclusive use of synthetic, rule-based datasets eliminates privacy risks associated with empirical human or ecological data while maintaining a high-fidelity environment for algorithmic validation.
- **Algorithmic Interpretability:** Emphasis on Bayesian and HMM architectures supports transparent probabilistic reasoning, mitigating the "black-box" limitations often associated with deep learning approaches in sensitive research contexts.
- **Bias Mitigation:** Because the data generation process is fully controlled, sampling distributions and masking mechanisms are explicitly defined, reducing unintended bias and enabling objective comparative evaluation.

The expected outcome is a validated computational framework demonstrating that probabilistic models can accurately reconstruct latent states in hierarchical stochastic systems under controlled uncertainty. By coupling deterministic dataset replay with systematic robustness profiling, the project establishes a reproducible methodology for state inference applicable across diverse domains—from conservation modeling to broader data science contexts—where reasoning under incomplete information is essential.

## Conclusion

High-dimensional, hierarchical datasets are frequently constrained by missing or sparsely sampled observations, limiting the ability to model complex dependencies and predict latent structure with confidence. This project addresses that challenge by developing a simulation-driven benchmarking framework that evaluates the robustness of probabilistic architectures—specifically Bayesian models, Hidden Markov Models, and graph-based approaches—in reconstructing latent states from intentionally degraded datasets. By leveraging the `msprime` engine to generate reproducible, ground-truth data, the system enables rigorous, controlled evaluation of how probabilistic models behave under varying levels of uncertainty and masking.

The resulting framework integrates rule-based simulation, structured statistical modeling, and interactive visualization into a reproducible and scalable toolset for analyzing inference performance under incomplete information. A deterministic replay strategy ensures that experimental results reflect architectural strengths rather than stochastic variation in data generation. Through systematic robustness profiling and calibration analysis, the project establishes measurable thresholds at which different probabilistic models succeed or degrade.

The outcome is a validated computational methodology for state reconstruction under uncertainty, applicable to domains ranging from population modeling to broader data science contexts where hierarchical structure and missingness are intrinsic challenges. More broadly, this work contributes a reproducible, interpretable, and ethically grounded framework for reasoning under uncertainty—demonstrating how simulation-driven benchmarking can advance probabilistic machine learning in research environments where complete data is unattainable.

## 6 References

### References

- [1] Kelleher, J., Etheridge, A. M., & McVean, G. (2018). *Efficient coalescent simulation and genealogical analysis for large sample sizes*. *PLOS Computational Biology*, 14(5), e1006581.
- [2] Salvatier, J., Wiecki, T., & Fonnesbeck, C. (2016). *Probabilistic programming in Python using PyMC3*. *PeerJ Computer Science*.
- [3] Paszke, A., et al. (2019). *PyTorch: An imperative style, high-performance deep learning library*. *NeurIPS*.
- [4] Browning, S. R., & Browning, B. L. (2007). *Rapid and accurate haplotype phasing and missing-data inference for whole-genome association studies by use of localized haplotype clustering*. *The American Journal of Human Genetics*, 81(5), 1084–1097.
- [5] FastAPI Documentation (2024). *Modern web framework for building APIs with Python 3.7+*.
- [6] React Documentation (2024). *React: A JavaScript library for building user interfaces*.