MISSBGM: MISSINGNESS-AWARE DATA IMPUTATION VIA AI-POWERED BAYESIAN GENERATIVE MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

Missing data imputation remains a fundamental challenge in modern data science, especially when uncertainty quantification is essential. In this work, we propose MissBGM, an AI-powered tabular data imputation method via Bayesian generative modeling that combines the flexibility of neural networks with the statistical rigor of Bayesian inference. Unlike existing methods that often focus on point estimates or treat the missingness mechanism implicitly, MissBGM jointly models the data-generating and missingness mechanisms explicitly, producing posterior distribution over each missing entry. We develop a novel stochastic optimization algorithm with alternating updates of missing values, model parameters, and latent variables until convergence. For the theoretical analysis, we establish convergence to a uniquely separated pseudo-true tempered target under explicit criterion, approximation, and optimization conditions. Empirically, MissBGM outperforms classical and neural imputers across a range of settings. These results establish MissBGM as a principled and scalable solution for modern missing data imputation. The code and tutorials are available at `https://github.com/anonymous`.

1 INTRODUCTION

Missing data is pervasive in modern machine learning and data science, arising in domains such as healthcare, social science, and industrial analytics (Rubin, 1976; Little & Rubin, 2019; Enders, 2022; Schafer & Graham, 2002; Allison, 2009). It can substantially degrade downstream analyses and lead to misleading scientific conclusions if not handled properly. This challenge is particularly acute when the missingness pattern is complex, structured, or driven by the unobserved values themselves (Mitra et al., 2023). Developing methods that can flexibly capture these complex dependencies while remaining statistically principled and computationally scalable is therefore a central problem for modern missing data analysis.

Classic statistical methods, such as likelihood-based models (Schafer, 1997), expectation-maximization procedures (Dempster et al., 1977), and multiple imputation frameworks (Rubin, 1987; Van Buuren & Groothuis-Oudshoorn, 2011; Carpenter et al., 2023), offer a principled way of handling missing data. However, they rely heavily on restrictive parametric models or repeated conditional modeling, which limit the scalability in modern complex and non-linear high-dimensional data. By contrast, modern deep learning methods have greatly improved imputation accuracy by leveraging flexible neural architectures (Yoon et al., 2018; Mattei & Frellsen, 2019; Ipsen et al., 2020; Tashiro et al., 2021; Jolicoeur-Martineau et al., 2024; Zheng & Charoenphakdee, 2022; Zhang et al., 2025). However, those modern deep imputers primarily target point estimation and often rely on training-time masking distributions. They handle missingness mechanism implicitly or heuristically. Their multiple imputations are not automatically a principled posterior for missing values. A principled way for uncertainty quantification is essential for reliable and trustworthy data imputation (Begoli et al., 2019; Abdar et al., 2021; Hossain & Milani, 2025).

To address these limitations, we propose MissBGM, a **Miss**ing data imputation framework based on **B**ayesian **G**enerative **M**odeling that jointly models the data-generating process, and missingness mechanism, providing both point estimate and uncertainty quantification for missing values. By explicitly modeling the masking process, MissBGM accounts for complex and non-ignorable missingness patterns that are common in real-world data. To make this framework practical at scale,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

we develop an alternating stochastic optimization algorithm that iteratively updates missing values, latent variables, and model parameters in a mini-batch manner until convergence. In this way, MissBGM closes the gap between the flexibility of modern AI and the statistical rigor of Bayesian inference, offering a powerful and scalable solution for modern missing data imputation. We summarize the contributions of MissBGM as follows.

- MissBGM bridges the gap between powerful deep learning models and statistically principled Bayesian inference for missing data imputation with uncertainty quantification by explicitly modeling potentially non-ignorable missingness mechanisms.
- we design a novel alternating stochastic optimization algorithm for efficient mini-batch training and independent per-sample posterior inference, making the MissBGM framework scalable for large-scale applications.
- We characterize the criterion-induced target of an idealized profiled estimator and prove convergence under explicit separation and optimization conditions. Moreover, we demonstrate the superior performance of MissBGM across a range of settings.

2 Related work

2.1 Classical statistical imputation methods

The classical statistical imputation methods, including likelihood-based methods (Schafer, 1997), EM-type algorithms (Dempster et al., 1977), and multiple imputation methods (Rubin, 1987; Van Buuren & Groothuis-Oudshoorn, 2011; Carpenter et al., 2023), are grounded in explicit assumptions about the data-generating process and missingness mechanisms. Tree-based methods (Wilson et al., 2023; Doove et al., 2014) that incorporate multiple imputation often yield strong empirical performance on tabular data with nonlinear feature dependencies. Although these methods are relatively interpretable, they generally rely on repeated conditional modeling heuristics rather than a fully generative modeling framework. More importantly, most classical statistical imputation approaches are well justified under simplified missingness assumptions and become less reliable when the masking mechanism is complex, structured, or non-ignorable (Rubin, 1976). They also scale poorly in large-scale or high-dimensional settings.

2.2 Deep learning-based imputation methods

Recent advances in generative AI have led to the development of neural data imputation methods. These include variational autoencoder (VAE)-based approaches (Mattei & Frellsen, 2019; Ivanov et al., 2018; Nazabal et al., 2020; Fortuin et al., 2020; Peis et al., 2022), generative adversarial network (GAN)-based methods (Yoon et al., 2018; Li et al., 2019), and diffusion-style methods (Tashiro et al., 2021; Zheng & Charoenphakdee, 2022; Jolicoeur-Martineau et al., 2024). For example, GAIN (Yoon et al., 2018) formulates imputation as an adversarial learning problem, using a generator–discriminator architecture to produce plausible completions of incomplete data. ForestDiffusion (Jolicoeur-Martineau et al., 2024) combines diffusion-style generative modeling with tree-based learning, aiming to better adapt diffusion ideas to tabular data. TabCSDI (Zheng & Charoenphakdee, 2022) builds on conditional score-based diffusion models and performs imputation by iteratively denoising incomplete samples conditioned on observed entries.

The existing methods emphasize accurate reconstruction and either assume ignorable missingness or treat the mask simply as conditioning information or an auxiliary input, rather than as part of an explicit probabilistic missingness model. As a result, these methods provide limited interpretability regarding why entries are missing, and they do not directly support coherent inference under a specified missingness mechanism. This limitation becomes especially important in real-world settings where missingness may be structured, complex, or potentially non-ignorable.

A small but highly relevant line of work models the missingness process itself. For example, not-MIWAE (Ipsen et al., 2020) extends MIWAE (Mattei & Frellsen, 2019) to non-ignorable missingness settings by introducing a learned model for the mask conditional on the data. GINA (Ma & Zhang, 2021) combines an identifiable VAE with a neural missingness model for non-ignorable missingness settings. These methods explicitly model the missingness process. However, they are

typically based on variational frameworks, where posterior inference is from encoder-based approximations. In practice, such approximations can be restrictive and may yield under-estimated uncertainty (Margossian & Saul, 2023).

Positioning of MissBGM MissBGM bridges the gap between a scalable neural imputer and a statistically principled Bayesian model for missing data imputation. Its data component builds on recent advances in Bayesian generative models (Liu & Wong, 2026b;a; Luo & Liu, 2026), while its central methodological advance is to jointly model the data-generating process with an explicit probabilistic model of the missingness mechanism.

Unlike most neural imputers that condition on the mask implicitly, MissBGM explicitly models the missingness mechanism. Unlike VAE-based methods, such as MIWAE (Mattei & Frellsen, 2019) and GINA (Ma & Zhang, 2021), which use a shared encoder to parameterize approximate posteriors across samples, MissBGM performs sample-specific inference for latent variables and missing entries. This avoids committing sample-level inference to a fixed amortized posterior family and directly supports posterior-sampling-based uncertainty quantification.

3 Problem Setup

We consider a dataset with n samples and p features, with complete data matrix $\mathbf{X} \in \mathbb{R}^{n\times p}$ and missingness indicator matrix $\mathbf{R} \in \{0,1\}^{n\times p}$. An entry $r_{ij} = 1$ indicates that j-th feature of i-th sample is observed while $r_{ij} = 0$ indicates a missing entry. We partition $\mathbf{x}_i \in \mathbb{R}^p$ into $\mathbf{x}_i = (\mathbf{x}_{i,\mathrm{obs}}, \mathbf{x}_{i,\mathrm{mis}})$, where $\mathbf{x}_{i,\mathrm{obs}}$ and $\mathbf{x}_{i,\mathrm{mis}}$ are the observed and missing components of $\mathbf{x}_i$, respectively.

Our objective is to infer the posterior distribution of the missing values given the observed data and missingness indicator matrix, which is given by

$$p(\mathbf{X}_{\mathrm{mis}} \mid \mathbf{X}_{\mathrm{obs}}, \mathbf{R}) .$$

The missingness mechanism is typically classified into three categories (Rubin, 1976; Little & Rubin, 2019):

- **Missing Completely At Random (MCAR).** The missingness is independent of the data, so $p(\mathbf{R} \mid \mathbf{X}) = p(\mathbf{R})$. In this case, the mask $\mathbf{R}$ carries no information about the values of $\mathbf{X}$, and one can in principle ignore $\mathbf{R}$ when inferring $\mathbf{X}_{\mathrm{mis}}$.

- **Missing At Random (MAR).** The missingness depends on the observed values but not on the missing ones, i.e. $p(\mathbf{R} \mid \mathbf{X}) = p(\mathbf{R} \mid \mathbf{X}_{\mathrm{obs}})$. Under MAR (and MCAR), the mask $\mathbf{R}$ is conditionally independent of $\mathbf{X}_{\mathrm{mis}}$ given $\mathbf{X}_{\mathrm{obs}}$, and hence

$$p(\mathbf{X}_{\mathrm{mis}} \mid \mathbf{X}_{\mathrm{obs}}, \mathbf{R}) = p(\mathbf{X}_{\mathrm{mis}} \mid \mathbf{X}_{\mathrm{obs}}) .$$

In other words, under MAR or MCAR (also known as *ignorable* missingness), valid imputation can be performed by conditioning on the observed data only.

- **Missing Not At Random (MNAR).** The missingness mechanism depends on both observed and missing values. In the MNAR regime (*non-ignorable* missingness), the mask $\mathbf{R}$ carries information about the missing values, and we typically have $p(\mathbf{X}_{\mathrm{mis}} \mid \mathbf{X}_{\mathrm{obs}}, \mathbf{R}) \neq p(\mathbf{X}_{\mathrm{mis}} \mid \mathbf{X}_{\mathrm{obs}})$. The observed-data law alone does not generally identify this conditional distribution and additional modeling or scientific restrictions are required.

In our setting, we allow for potentially non-ignorable (MNAR) patterns by explicitly modeling the missingness mechanism. Hence we aim to learn a joint generative model of both the data-generating process and the missingness mechanism that yields a faithful approximation to the posterior $p(\mathbf{X}_{\mathrm{mis}} \mid \mathbf{X}_{\mathrm{obs}}, \mathbf{R})$. Our goal is not only to accurately reconstruct the missing values, but to recover the full conditional distribution of the missing entries, providing both point imputations (e.g. posterior means) and uncertainty estimates (e.g. credible intervals).

4 Method

MissBGM specifies a joint latent-variable model for data-generating process and missingness mechanism, with alternating stochastic optimization over latent variables, missing entries, and global model parameters.

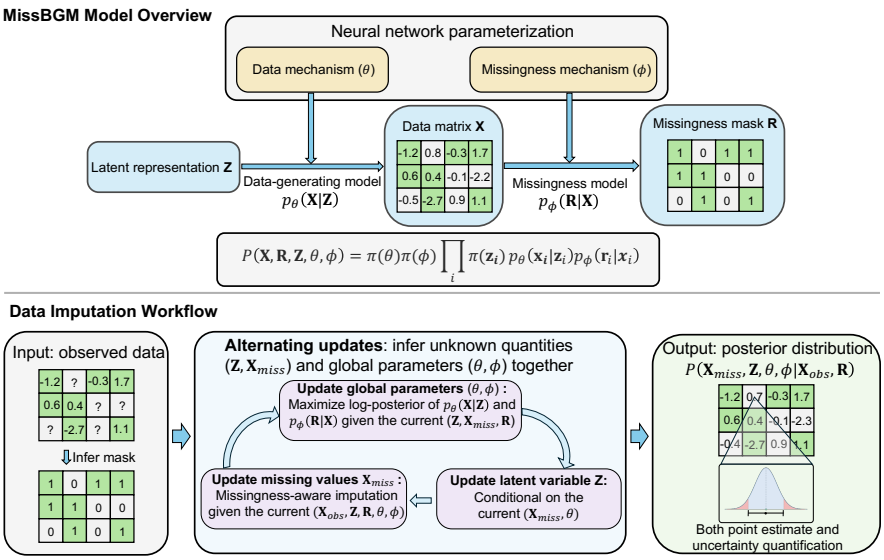

Figure 1: Overview of MissBGM. The model jointly represents a data-generating process $p_\theta(\mathbf{X} \mid \mathbf{Z})$ and a missingness mechanism $p_\phi(\mathbf{R} \mid \mathbf{X})$. During inference, observed entries remain fixed while latent variables and missing values are updated alternately. Per-sample posterior inference then provides both point estimates and uncertainty quantification for each missing value.

4.1 Data and missingness generative process

For each sample $i \in \{1, \ldots, n\}$, let $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})^\top \in \mathbb{R}^p$ denote the complete data vector and let $\mathbf{r}_i = (r_{i1}, \ldots, r_{ip})^\top \in \{0, 1\}^p$ denote the corresponding observation mask. We introduce a latent variable $\mathbf{z}_i \in \mathbb{R}^d$ in the latent space and define the joint model

$$p(\mathbf{X}, \mathbf{R}, \mathbf{Z}, \theta, \phi) = \pi(\theta)\pi(\phi) \prod_{i=1}^{n} \pi(\mathbf{z}_i)\, p_\theta(\mathbf{x}_i \mid \mathbf{z}_i)\, p_\phi(\mathbf{r}_i \mid \mathbf{x}_i), \tag{1}$$

where $\pi(\mathbf{z}_i) = \mathcal{N}(0, I_d)$, $\pi(\theta)$ and $\pi(\phi)$ are prior distributions on the network parameters, $p_\theta(\mathbf{x}_i \mid \mathbf{z}_i)$ is the data model, and $p_\phi(\mathbf{r}_i \mid \mathbf{x}_i)$ is the missingness model. d is the dimension of the latent variable. All the prior distributions are chosen to be multivariate normal distributions for simplicity.

By default, the data model $p_\theta(\mathbf{x}_i \mid \mathbf{z}_i)$ is set to be a multivariate normal distribution for continuous variables and a Bernoulli or Categorical distribution for discrete variables. In a typical continuous setting (e.g., tabular data), the data model is given by

$$\mathbf{x}_i \mid \mathbf{z}_i, \theta \sim \mathcal{N}(\mu_\theta(\mathbf{z}_i), \Sigma_\theta(\mathbf{z}_i)), \tag{2}$$

where both mean and covariance matrix are learnable functions of latent variable $\mathbf{z}_i$ parameterized by θ. In practice, we can further simplify the covariance matrix to be a diagonal matrix $\Sigma_\theta(\mathbf{z}_i) = \mathrm{diag}(\sigma^2_{\theta,1}(\mathbf{z}_i), \ldots, \sigma^2_{\theta,p}(\mathbf{z}_i))$. We represent $(\mu_\theta(\mathbf{z}_i), \Sigma_\theta(\mathbf{z}_i))$ as a neural network g_θ with two output heads, and the weights are shared across all samples.

We use another neural network $f_\phi(\mathbf{x}_i) \in \mathbb{R}^p$ to model the missingness mechanism and define

$$r_{ij} \mid \mathbf{x}_i, \phi \sim \mathrm{Bernoulli}(\mathrm{sig}(f_{\phi,j}(\mathbf{x}_i))), \qquad j = 1, \ldots, p, \tag{3}$$

where $\mathrm{sig}(x) = 1/(1 + \exp(-x))$. The probability of observing feature j may depend on the full completed data vector $\mathbf{x}_i$, and hence the model can capture the comprehensive missingness process in a flexible and data-driven way.

To account for uncertainty in model parameter space, we can also adopt Bayesian neural networks (Goan & Fookes, 2020) for g_θ and f_ϕ to place priors on the network parameters. This gives a flexible nonparametric representation of both the data-generating mechanism and the missingness mechanism while preserving a coherent probabilistic model for uncertainty quantification.

4.2 ALTERNATING STOCHASTIC OPTIMIZATION

We designed a stochastic optimization algorithm to update the model parameters, latent variables, and missing values iteratively until convergence. The joint posterior distribution is given by

$$p(\mathbf{Z}, \mathbf{X}_{\text{mis}}, \theta, \phi \mid \mathbf{X}_{\text{obs}}, \mathbf{R}) \propto \pi(\theta)\pi(\phi) \prod_{i=1}^{n} \pi(\mathbf{z}_i)\, p_\theta(\mathbf{x}_i \mid \mathbf{z}_i)\, p_\phi(\mathbf{r}_i \mid \mathbf{x}_i), \tag{4}$$

where $\mathbf{X} = (\mathbf{X}_{\text{obs}}, \mathbf{X}_{\text{mis}})$. Since the true posterior joint distribution is not tractable, we approximate the target by the following iterative procedure. Specifically, we iteratively (i) update the latent variables $\mathbf{Z}$ from $p(\mathbf{Z} \mid \mathbf{X}, \mathbf{R}, \theta, \phi)$, (ii) update the missing values $\mathbf{X}_{\text{mis}}$ from $p(\mathbf{X}_{\text{mis}} \mid \mathbf{Z}, \mathbf{X}_{\text{obs}}, \mathbf{R}, \theta, \phi)$, and (iii) update the model parameters θ and ϕ from $p(\theta, \phi \mid \mathbf{Z}, \mathbf{X}_{\text{mis}}, \mathbf{X}_{\text{obs}}, \mathbf{R})$.

Step (i): update latent variable. Up to an additive constant, the log-posterior is given by

$$\log p(\mathbf{z}_i \mid \mathbf{x}_i, \mathbf{r}_i, \theta, \phi) = \log \pi(\mathbf{z}_i) + \log p_\theta(\mathbf{x}_i \mid \mathbf{z}_i) + C_1. \tag{5}$$

Under diagonal covariance structure, the objective function is given by

$$\mathcal{L}_{z_i} = \frac{1}{2}\|\mathbf{z}_i\|_2^2 + \sum_{j=1}^{p} \left[\frac{(x_{ij} - \mu_{\theta,j}(\mathbf{z}_i))^2}{2\sigma_{\theta,j}^2(\mathbf{z}_i)} + \frac{1}{2} \log \sigma_{\theta,j}^2(\mathbf{z}_i) \right], \tag{6}$$

where $\mu_{\theta,j}(\mathbf{z}_i)$ and $\sigma_{\theta,j}^2(\mathbf{z}_i)$ are the j-th element of the mean and covariance vector, respectively. We update $\mathbf{z}_i$ by minimizing the objective function using gradient descent in (6) in parallel for all samples.

Step (ii): update missing values. Up to an additive constant, we define the β-tempered log-posterior by

$$\log p_\beta(\mathbf{x}_{i,\text{mis}} \mid \mathbf{z}_i, \mathbf{x}_{i,\text{obs}}, \mathbf{r}_i, \theta, \phi) = \log p_\theta(\mathbf{x}_i \mid \mathbf{z}_i) + \beta \log p_\phi(\mathbf{r}_i \mid \mathbf{x}_i) + C_2. \tag{7}$$

Under the diagonal Gaussian data model in (2) and the Bernoulli missingness model in (3), the objective function is given by

$$\mathcal{L}_{x_{i,\text{mis}}} = \sum_{j=1}^{p} \frac{(x_{ij} - \mu_{\theta,j}(\mathbf{z}_i))^2}{2\sigma_{\theta,j}^2(\mathbf{z}_i)} - \beta \sum_{j=1}^{p} \left[r_{ij} \log \text{sig}(f_{\phi,j}(\mathbf{x}_i)) + (1 - r_{ij}) \log(1 - \text{sig}(f_{\phi,j}(\mathbf{x}_i))) \right], \tag{8}$$

where β is a tempering coefficient to balance the data and missingness terms. When $\beta = 1$, this target coincides with the ordinary conditional posterior. We update $\mathbf{x}_{i,\text{mis}}$ by minimizing the objective function using gradient descent in (8) in parallel for all samples.

Step (iii): update model parameters. Up to an additive constant, the log-posterior is given by

$$\begin{cases} \log p(\theta \mid \mathbf{Z}, \mathbf{X}) = \log \pi(\theta) + \log p_\theta(\mathbf{X} \mid \mathbf{Z}) + C_3, \\ \log p(\phi \mid \mathbf{X}, \mathbf{R}) = \log \pi(\phi) + \log p_\phi(\mathbf{R} \mid \mathbf{X}) + C_4. \end{cases} \tag{9}$$

To account for variation in model parameters with BNNs, which use variational inference (VI) (Blei et al., 2017) to approximate (9), we can use the following KL-weighted variational objectives (see Appendix A):

$$\begin{cases} \mathcal{L}_\theta = \mathbb{E}_{q_\lambda(\theta)}[\log p_\theta(\mathbf{X} \mid \mathbf{Z})] - \tau \,\text{KL}\big(q_\lambda(\theta) \,\|\, \pi(\theta)\big), \\ \mathcal{L}_\phi = \mathbb{E}_{q_\psi(\phi)}[\log p_\phi(\mathbf{R} \mid \mathbf{X})] - \tau \,\text{KL}\big(q_\psi(\phi) \,\|\, \pi(\phi)\big), \end{cases} \tag{10}$$

where τ is the Kullback-Leibler divergence (KL) weight for both variational families. Two variational distributions $q_\lambda(\theta) = \mathcal{N}(\theta \mid \mu_\lambda, \sigma_\lambda^2)$ and $q_\psi(\phi) = \mathcal{N}(\phi \mid \mu_\psi, \sigma_\psi^2)$ are used to approximate the posterior distributions of θ and ϕ, respectively. $\lambda = (\mu_\lambda, \sigma_\lambda^2)$ and $\psi = (\mu_\psi, \sigma_\psi^2)$ are the learnable variational parameters. Diagonal-covariance Gaussians are used for variational distributions.

We update θ and ϕ by first maximizing the KL-weighted variational objectives with respect to λ and ψ in (10) with stochastic gradients on mini-batches of samples, and then sampling θ and ϕ from the variational distributions $q_\lambda(\theta)$ and $q_\psi(\phi)$, respectively.

After iterations of steps (i)-(iii) until convergence, we can obtain the point estimates of the missing values of the training dataset. The full alternating stochastic optimization algorithm is summarized in Algorithm 1.

4.3 POSTERIOR INFERENCE AND UNCERTAINTY QUANTIFICATION

Once the MissBGM model is trained, it performs conditional inference for incomplete samples without retraining the global network parameters. Given a (possibly new) test dataset with missingness pattern $\mathbf{R}$, our goal is to draw approximate samples from the β-tempered conditional posterior of the latent variables and missing values,

$$p_\beta(\mathbf{Z}, \mathbf{X}_{\text{mis}} \mid \mathbf{X}_{\text{obs}}, \mathbf{R}, \theta, \phi) \propto \prod_{i=1}^{n} \pi(\mathbf{z}_i)\, p_\theta(\mathbf{x}_i \mid \mathbf{z}_i)\, p_\phi^\beta(\mathbf{r}_i \mid \mathbf{x}_i), \tag{11}$$

with the global parameters (θ, ϕ) fixed (or drawn from fixed variational distributions $q_\lambda(\theta)$ and $q_\psi(\phi)$ in the BNN setting). All inference is per-sample, so the posterior sampling from (11) can be done in parallel across samples to improve efficiency.

Warm-start by MAP. We initialize $(\mathbf{Z}, \mathbf{X}_{\text{mis}})$ at a local maximum a posteriori (MAP) estimate by iterating only steps (i) and (ii) of Algorithm 1 for a number of iterations T_{warm}. We denote the resulting warm-start state by $(\mathbf{Z}^\star, \mathbf{X}_{\text{mis}}^\star)$, which already provides a good point estimate of the latent variables and missing values.

HMC-within-Gibbs sampler. To sample from the full conditional posterior (11), we use the Hamiltonian Monte Carlo (HMC) within Gibbs sampler (Tierney, 1994; Neal, 2011). Starting from $(\mathbf{Z}^\star, \mathbf{X}_{\text{mis}}^\star)$, we draw posterior samples by alternating two HMC updates that share the same target (11):

$$\begin{cases} \log p(\mathbf{z}_i \mid \mathbf{x}_i, \theta) = \log\ \pi(\mathbf{z}_i) + \log p_\theta(\mathbf{x}_i \mid \mathbf{z}_i) + C_5, \\ \log p_\beta(\mathbf{x}_{i,\text{mis}} \mid \mathbf{x}_{i,\text{obs}}, \mathbf{r}_i, \mathbf{z}_i, \theta, \phi) = \log\ p_\theta(\mathbf{x}_i \mid \mathbf{z}_i) + \beta \log\ p_\phi(\mathbf{r}_i \mid \mathbf{x}_i) + C_6. \end{cases} \tag{12}$$

The two targets in (12) are exactly the negative per-sample objectives that drove the MAP warm-start in steps (i) and (ii), so the sampler can be viewed as a stochastic refinement of MAP that further explores posterior uncertainty rather than a point estimate. At each Gibbs sweep, we (a) run one HMC step on $\mathbf{z}_i$ with $\mathbf{x}_i$ held fixed, then (b) run one HMC step on $\mathbf{x}_{i,\text{mis}}$ with the just-updated $\mathbf{z}_i$ held fixed. Note that we use the same temperature coefficient β as training stage.

Both HMC kernels use leapfrog integration with a small number of leapfrog steps and an adaptive step size controlled by a simple dual-averaging scheme targeting an acceptance rate of approximately 0.75. Step-size adaptation is restricted to the first portion of the burn-in window and is then frozen, so that the post-burn-in draws form a valid Markov chain with respect to the fixed target. Since the targets in (12) are differentiable in both $\mathbf{z}_i$ and $\mathbf{x}_{i,\text{mis}}$, gradient-based proposals mix substantially better than random-walk Metropolis, especially when p is moderately large.

Uncertainty quantification. After discarding burn-in iterations, we collect S joint draws $\{(\mathbf{Z}^{(s)}, \mathbf{X}_{\text{mis}}^{(s)})\}_{s=1}^S$ and reconstruct the corresponding completed matrices $\widetilde{\mathbf{X}}^{(s)} = \mathbf{R} \odot \mathbf{X}_{\text{obs}} + (1 - \mathbf{R}) \odot \mathbf{X}_{\text{mis}}^{(s)}$. From these posterior samples, we compute a posterior mean as a point estimate, and an empirical prediction interval (PI) with a user-specified significance level α from the lower and upper quantiles.

$$\widehat{x}_{ij} = \frac{1}{S} \sum_{s=1}^{S} x_{ij}^{(s)}, \qquad \widehat{\text{PI}}_{ij}^{1-\alpha} = \Big[Q_{\alpha/2}\big(\{x_{ij}^{(s)}\}_{s=1}^S\big),\ Q_{1-\alpha/2}\big(\{x_{ij}^{(s)}\}_{s=1}^S\big) \Big], \tag{13}$$

where $Q_q(\cdot)$ denotes the q-quantile function. Unlike point-estimate imputers that only return $\widehat{\mathbf{X}}_{\text{mis}}$, MissBGM thus delivers a full posterior of the missing values, providing much richer information for downstream tasks.

5 THEORETICAL ANALYSIS

To characterize an idealized statistical target associated with the β-tempered local updates, we study a profiled M-estimator rather than the finite-epoch blockwise iterates. The inner supremum jointly profiles the two local blocks under the tempered complete-data objective while the outer criterion is the associated global objective for (θ, ϕ). For one observed pair $O = (\mathbf{x}_{\text{obs}}, \mathbf{r})$, define

$$m_\beta(O; \theta, \phi) := \sup_{\mathbf{z}, \mathbf{x}_{\text{mis}}} \Big\{ \log \pi(\mathbf{z}) + \log p_\theta(\mathbf{x} \mid \mathbf{z}) + \beta \log p_\phi(\mathbf{r} \mid \mathbf{x}) \Big\}, \qquad \mathbf{x} = (\mathbf{x}_{\text{obs}}, \mathbf{x}_{\text{mis}}), \tag{14}$$

and let $L_n^\beta(\theta,\phi) = n^{-1}\sum_{i=1}^n m_\beta(O_i;\theta,\phi)$ with population target $L^\beta(\theta,\phi) = \mathbb{E}_{P_0}[m_\beta(O;\theta,\phi)]$. Associate each (θ,ϕ) with the normalized completed-data law $q^\beta_{\theta,\phi}(\mathbf{x},\mathbf{r}) \propto \int \pi(\mathbf{z})p_\theta(\mathbf{x}\mid\mathbf{z})p_\phi(\mathbf{r}\mid\mathbf{x})^\beta d\mathbf{z}$ and its missing-value conditional kernel $K^\beta_{\theta,\phi}(\cdot\mid O)$. The profiled criterion and this integrated law are distinct objects and the separation condition below explicitly links them.

Theorem 1 (Convergence to the criterion-induced imputation law). Assume A1–A6 in Appendix B: uniform convergence of the profiled criterion, vanishing sieve and optimization errors, and a profile-to-law separation condition ensuring that near-maximizers of L^β induce a unique completed-data law q^*_β. If $(\hat\theta_n,\hat\phi_n)$ approximately maximizes L_n^β, then

$$\begin{cases} L^{\beta,*} - L^\beta(\hat\theta_n,\hat\phi_n) = o_p(1), \\ \left\| q^\beta_{\hat\theta_n,\hat\phi_n} - q^*_\beta \right\|_{\mathrm{TV}} = o_p(1), \\ \int \left\| K^\beta_{\hat\theta_n,\hat\phi_n}(\cdot\mid O) - K^*_\beta(\cdot\mid O)\right\|_{\mathrm{TV}} q^*_{\beta,O}(dO) = o_p(1). \end{cases} \tag{15}$$

where $L^{\beta,*} = \sup_{\theta,\phi} L^\beta(\theta,\phi)$, $q^*_{\beta,O}$ is the observed-data marginal of q^*_β, and K^*_β is its conditional kernel. We provide detailed proof in Appendix B, a corrected margin-rate result, and an optional posterior-mean corollary under a Gaussian-compatible moment envelope.

This theorem is a criterion-consistency result, not a generic MNAR identification theorem. Assumption A6 is a substantive profile-to-law condition and is neither implied by neural-network flexibility nor generally verifiable from observed data alone under unrestricted MNAR. Recovery of the true MNAR conditional law at $\beta = 1$ additionally requires identifying restrictions and profile-to-law Fisher consistency. In suitable settings, identification can be established using a valid shadow variable (nonresponse instrument) under appropriate model restrictions (Wang et al., 2014) or completeness and related regularity conditions (Miao et al., 2024). Thus, correct specification and $\beta = 1$ alone do not guarantee recovery of the true missing-value conditional law.

6 Experiments

6.1 Experimental setup

Datasets. We evaluate MissBGM on a synthetic benchmark and four real-world tabular datasets that span a wide range of sample sizes and feature dimensions. The synthetic benchmark uses the same conditional Gaussian complete-data process under two masking regimes. Point-imputation experiments use upper-tail self-masked MNAR, whereas the uncertainty-quantification benchmark uses MCAR to isolate interval calibration from non-ignorable extrapolation. The real-data benchmark includes four commonly used datasets from the UCI Machine Learning Repository (Asuncion et al., 2007) with different sample sizes ($n = 178$ to $13,500$) and feature dimensions ($p = 13$ to $5,000$). We add an MNAR missingness mechanism to each real-world dataset with varying missing rates, where the observation probability depends on both the target value itself and a nonlinear summary of the full feature vector. The full details of the datasets and missingness mechanism are given in Appendix C.

Evaluation metrics. For point imputation, we report the root mean squared error (RMSE) on missing entries. For uncertainty quantification, we additionally report the width ratio (WR) and empirical coverage of the predicted $(1-\alpha)$ intervals. WR is the estimated average interval width divided by the oracle average interval width, so values closer to 1 indicate better-calibrated interval width. Estimated posterior S.D.s and intervals are computed from 1,000 multiple imputations (or posterior samples) per missing entry.

Baseline methods.

We compare MissBGM against nine data imputation baselines, including classical and nonparametric baselines (Mean), optimal-transport imputation (OT) (Muzellec et al., 2020), iterative chained equations (ICE) (Van Buuren & Groothuis-Oudshoorn, 2011), the ensemble multiple-imputation method (miceforest) (Wilson et al., 2023), the tree-based diffusion generative imputation method (ForestDiffusion) (Jolicoeur-Martineau et al., 2024), generative AI baselines

Table 1: RMSE on synthetic datasets with varying sample sizes and missing rates. We report both mean and standard deviation of RMSE across 10 runs. Best result in each row is **bolded**.

Size	Rate	Mean	OT	Miceforest	ICE	GAIN	ForestDiffusion	TabCSDI	not-MIWAE	GINA	MissBGM
0.5k	0.3	1.849	1.664 (0.006)	1.656 (0.011)	1.652 (0.015)	1.792 (0.018)	1.617 (0.013)	1.665 (0.015)	1.725 (0.014)	**1.610 (0.018)**	1.620 (0.011)
	0.5	1.817	1.640 (0.008)	1.630 (0.014)	1.611 (0.010)	1.877 (0.014)	1.641 (0.010)	1.626 (0.010)	1.621 (0.016)	1.608 (0.017)	**1.585 (0.009)**
	0.7	1.887	1.728 (0.008)	1.708 (0.012)	1.696 (0.014)	1.992 (0.015)	1.795 (0.011)	1.712 (0.013)	1.732 (0.015)	1.693 (0.019)	**1.673 (0.012)**
1k	0.3	1.798	1.602 (0.007)	1.596 (0.011)	1.591 (0.011)	1.671 (0.017)	1.575 (0.009)	1.615 (0.010)	1.574 (0.015)	1.578 (0.016)	**1.568 (0.009)**
	0.5	1.765	1.574 (0.005)	1.570 (0.013)	1.556 (0.009)	1.649 (0.012)	1.567 (0.012)	1.589 (0.009)	**1.532 (0.017)**	1.541 (0.015)	1.538 (0.011)
	0.7	1.838	1.674 (0.009)	1.644 (0.014)	1.636 (0.012)	1.808 (0.015)	1.727 (0.009)	1.667 (0.012)	1.636 (0.014)	1.633 (0.017)	**1.619 (0.011)**
5k	0.3	1.816	1.632 (0.005)	1.627 (0.010)	1.618 (0.012)	1.617 (0.013)	1.629 (0.008)	1.659 (0.010)	1.601 (0.012)	1.610 (0.014)	**1.599 (0.010)**
	0.5	1.793	1.603 (0.004)	1.595 (0.009)	1.586 (0.010)	1.650 (0.010)	1.599 (0.007)	1.616 (0.009)	1.578 (0.013)	**1.568 (0.015)**	1.574 (0.008)
	0.7	1.870	1.697 (0.006)	1.676 (0.011)	1.666 (0.009)	1.799 (0.012)	1.736 (0.009)	1.686 (0.010)	1.676 (0.013)	1.670 (0.016)	**1.662 (0.009)**
10k	0.3	1.772	1.592 (0.008)	1.587 (0.008)	1.577 (0.010)	1.632 (0.011)	1.608 (0.009)	1.612 (0.007)	1.573 (0.010)	1.580 (0.013)	**1.562 (0.006)**
	0.5	1.750	1.566 (0.005)	1.558 (0.007)	1.548 (0.009)	**1.536 (0.009)**	1.575 (0.008)	1.575 (0.009)	1.542 (0.009)	1.546 (0.012)	1.539 (0.005)
	0.7	1.824	1.661 (0.004)	1.636 (0.009)	1.625 (0.007)	1.999 (0.012)	1.728 (0.007)	1.665 (0.010)	1.630 (0.012)	1.646 (0.011)	**1.624 (0.008)**

(GAIN (Yoon et al., 2018) and TabCSDI (Zheng & Charoenphakdee, 2022)), and two MNAR baselines (not-MIWAE (Ipsen et al., 2020) and GINA (Ma & Zhang, 2021)). The full details of baselines implemented are given in Appendix D. The implementation details of MissBGM are given in Appendix E. The detailed instructions for using MissBGM software are given in Appendix F.

6.2 Synthetic experiments

We first evaluate MissBGM on synthetic benchmark by varying the sample size $n \in \{0.5\text{k}, 1\text{k}, 5\text{k}, 10\text{k}\}$ and the missing rate $r \in \{0.3, 0.5, 0.7\}$. $p = 50$ for each configuration and the experiment is repeated for 10 runs. As shown in Table 1, MissBGM attains the lowest missing-entry RMSE in 8 of the 12 configurations, the second-lowest RMSE in 3, and the third-lowest RMSE in 1. Classical baselines (OT, miceforest, ICE) are competitive but consistently outperformed as they do not model the masking mechanism. The generative AI imputers, such as GAIN, ForestDiffusion, and TabCSDI, are competitive only in selected settings. Overall, these results demonstrate that MissBGM provides robust and superior imputation accuracy across diverse settings.

Next, we evaluate uncertainty quantification under MCAR using the same complete-data process as the MNAR point-imputation benchmark. This MCAR experiment isolates interval calibration from the severe extrapolation induced by deterministic upper-tail MNAR and does not establish calibration under general MNAR. We compare MissBGM with four baselines that support multiple imputations and use 1,000 completions per missing entry. We report the RMSE, the width ratio (WR) relative to the oracle interval width, and empirical coverage at significance level α. As shown in Table 2, MissBGM attains the lowest RMSE and coverage closest to the nominal $1 - \alpha = 0.95$, together with a WR of 0.994.

Table 2: Results averaged over 10 runs on a synthetic benchmark ($n = 5000$, $p = 50$, missing rate $r = 0.2$, and $\alpha = 0.05$). The lowest RMSE and closest-to-target WR (1) and coverage ($1-\alpha = 0.95$) are **bolded**.

Method	RMSE	WR	Coverage
Miceforest	1.225	0.774	0.287
GAIN	1.087	0.003	0.002
ForestDiffusion	0.960	0.859	0.885
TabCSDI	1.262	0.071	0.068
MissBGM	**0.946**	**0.994**	**0.934**

6.3 Real-world experiments

We benchmark MissBGM against nine baselines on four standard tabular datasets in Table 3, spanning a wide range of sample sizes and feature dimensions, from Wine ($n = 178$, $p = 13$) to Gisette ($n = 13{,}500$, $p = 5{,}000$). MissBGM achieves the lowest mean RMSE across all four datasets. The advantage is substantial on the high-dimensional Gisette dataset, where MissBGM reduces RMSE by 10.2% relative to the best competing method. Notably, four baselines, Miceforest, ICE, ForestDiffusion, and not-MIWAE

Table 3: RMSE on real-world datasets. We report both mean and standard deviation of RMSE across 10 runs. Best result in each column is **bolded**.

Method	Wine	Breast	Concrete	Gisette
Mean	1.043	0.809	0.976	0.978
OT	0.697 (0.007)	0.583 (0.005)	0.538 (0.003)	0.605 (0.004)
Miceforest	0.698 (0.011)	0.599 (0.008)	0.556 (0.009)	-
ICE	0.829 (0.014)	0.618 (0.009)	0.662 (0.012)	-
GAIN	0.849 (0.017)	0.651 (0.013)	0.715 (0.015)	0.696 (0.009)
ForestDiffusion	0.779 (0.009)	0.672 (0.006)	0.655 (0.008)	-
TabCSDI	0.691 (0.008)	0.607 (0.006)	0.569 (0.007)	0.627 (0.004)
not-MIWAE	0.798 (0.014)	0.615 (0.013)	0.652 (0.011)	-
GINA	0.722 (0.017)	0.610 (0.012)	0.821 (0.010)	0.962 (0.008)
MissBGM	**0.682 (0.007)**	**0.511 (0.005)**	**0.529 (0.006)**	**0.543 (0.005)**

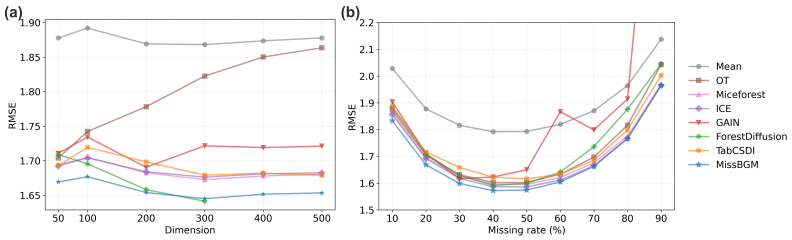

Figure 2: Robustness experiments on synthetic benchmark. (a) RMSE with varying dimensionality p at $n = 5000$ and missing rate 20%. (b) RMSE with varying missing rates at $n = 5000$ and $p = 50$.

fail to scale to Gisette (denoted by $-$). The results demonstrate that MissBGM scales well to high-dimensional data while maintaining superior imputation accuracy.

6.4 Robustness and Scalability

We next assess the robustness of MissBGM for feature dimensionality and missingness severity (Figure 2 and Appendix G). In Figure 2(a), OT degrades sharply with increasing p, reflecting the well-known difficulty of optimal transport in higher-dimensional settings. MissBGM achieves the lowest RMSE across all dimensions, except for $p = 300$, where it is slightly outperformed by ForestDiffusion. In Figure 2(b), MissBGM achieves the lowest RMSE across the entire range, where competitors such as GAIN diverge and tree-based methods like ForestDiffusion deteriorate noticeably. We explain the non-monotonic results of missing rate in Section C.1. These results suggest that MissBGM is robust to both dimensionality and missingness.

We further examine scalability from a computational perspective by comparing running time across methods on a moderate synthetic benchmark (Section H). The results show that MissBGM remains computationally practical and substantially faster than computationally intensive baselines such as miceforest and diffusion-based imputers, supporting its applicability to larger-scale problems.

6.5 Ablation Studies

We study the role of the tempering coefficient β, which weights the missingness term in (8) and the conditional target (11). $\beta = 0$ corresponds to ignoring the mask entirely. As shown in Table 4, modeling the mask with a small but non-zero β is beneficial in most scenarios. At the same time, performance degrades when β becomes too large ($\beta = 1$), suggesting that the mask should inform, but not dominate the imputation. Overall, we adopt $\beta = 0.01$ across all experiments as the default. We also study the optional Bayesian neural network version of MissBGM (Appendix I).

Table 4: Ablation study of MissBGM on the tempering coefficient β. Average RMSE over 10 runs on four real-world datasets. Best result in each row is **bolded**.

Dataset	$\beta = 0$	$\beta = 0.01$	$\beta = 0.1$	$\beta = 1$
Wine	0.706	**0.682**	0.693	0.707
Breast	0.529	0.511	**0.508**	0.536
Concrete	0.548	**0.529**	0.536	0.562
Gisette	0.567	**0.543**	0.556	0.569

7 Conclusion

We presented MissBGM, a missingness-aware Bayesian generative modeling approach for data imputation. MissBGM bridges the expressive flexibility of modern AI with the statistical rigor of Bayesian inference, enabling scalable tabular imputation. As with any joint model of data and missingness, reliability depends on appropriate specification of both the generative mechanism and the observation process. Neural networks training and Markov chain Monte Carlo (MCMC) can also computationally intensive relative to simpler imputers. Future work includes (a) investigating identifiability of latent variables by leveraging recent advances in nonlinear independent component analysis (ICA) theory (Khemakhem et al., 2020), and (b) exploring more advanced architectures for g_θ and f_ϕ when the data dependence is more complex (e.g., non-stationary).

AI USE STATEMENT

In this work, we used ChatGPT 5.6 and ChatGPT 6 mainly to edit the manuscript for readability, including sentence- and paragraph-level language polishing, and to check that symbols, equation numbers, formula notations. After implementing an initial version of this work, we used these AI tools to help organize the codebase, improve readability, and prepare a public Python PyPI package with external APIs for public use. After writing an initial proof of the main theoretical claims, we used the same tools to suggest additional related literature and extra assumptions that could make the argument more complete.

We did not use generative AI tools to formulate conceptual frameworks, generate data, design experiments, implement methods, or interpret empirical results. All AI-suggested changes were inspected by the authors and either accepted or rejected. We take responsibility for the final content of this work, including the text, code, claims, and results.

REPRODUCIBILITY STATEMENT

We include a self-contained, anonymized, and installable source archive in the supplementary material. After extracting the archive, the experiments can be reproduced through three complementary interfaces:

- **Local package and Python API:** install the source package with `pip install -e .` and follow the documented API example in the README.
- **Command-line entry point:** run `python main.py -c <config>` with any of the six supplied YAML configuration files.
- **Jupyter notebook:** open and execute `tutorial.ipynb` for an end-to-end example covering data generation, MAP and posterior-mean imputation, posterior prediction intervals, and calibration metrics.

The six YAML configurations record the model, optimization, missingness, and posterior-sampling settings for the two synthetic experiments and the four real-data experiments (Wine, Concrete, Breast, and Gisette). The released pipeline includes the data simulators and automated UCI data acquisition and preprocessing, and saves the resolved model configuration and per-epoch training history as `params.txt` and `training_history.csv`.

Algorithm 1 specifies the training procedure; Appendix F maps the documented API to an executable example; Appendix E records the posterior sampler, hyperparameters, and computing environment; and Appendices C and D document data generation and preprocessing, the shared evaluation inputs, and all baseline configurations used in the comparisons. The supplementary archive is the review-stage artifact; links to the public repository and package release will be added after acceptance.

REFERENCES

Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information fusion*, 76:243–297, 2021.

Paul D Allison. Missing data. *The SAGE handbook of quantitative methods in psychology*, 23:72–89, 2009.

Arthur Asuncion, David Newman, et al. Uci machine learning repository, 2007.

Edmon Begoli, Tanmoy Bhattacharya, and Dimitri Kusnezov. The need for uncertainty quantification in machine-assisted medical decision making. *Nature Machine Intelligence*, 1(1):20–23, 2019.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

James R Carpenter, Jonathan W Bartlett, Tim P Morris, Angela M Wood, Matteo Quartagno, and Michael G Kenward. *Multiple imputation and its application*. John Wiley & Sons, 2023.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society: series B (methodological)*, 39(1): 1–22, 1977.

Joshua V Dillon, Ian Langmore, Dustin Tran, Eugene Brevdo, Srinivas Vasudevan, Dave Moore, Brian Patton, Alex Alemi, Matt Hoffman, and Rif A Saurous. Tensorflow distributions. *arXiv preprint arXiv:1711.10604*, 2017.

Lisa L Doove, Stef Van Buuren, and Elise Dusseldorp. Recursive partitioning for missing data imputation in the presence of interaction effects. *Computational statistics & data analysis*, 72: 92–104, 2014.

Craig K Enders. *Applied missing data analysis*. Guilford Publications, 2022.

Vincent Fortuin, Dmitry Baranchuk, Gunnar Rätsch, and Stephan Mandt. Gp-vae: Deep probabilistic time series imputation. In *International conference on artificial intelligence and statistics*, pp. 1651–1661. PMLR, 2020.

Ethan Goan and Clinton Fookes. Bayesian neural networks: An introduction and survey. In *Case Studies in Applied Bayesian Data Science: CIRM Jean-Morlet Chair, Fall 2018*, pp. 45–87. Springer, 2020.

Zarin Tahia Hossain and Mostafa Milani. Beyond accuracy: An empirical study of uncertainty estimation in imputation. *arXiv preprint arXiv:2511.21607*, 2025.

Niels Bruun Ipsen, Pierre-Alexandre Mattei, and Jes Frellsen. not-miwae: Deep generative modelling with missing not at random data. *arXiv preprint arXiv:2006.12871*, 2020.

Oleg Ivanov, Michael Figurnov, and Dmitry Vetrov. Variational autoencoder with arbitrary conditioning. *arXiv preprint arXiv:1806.02382*, 2018.

Alexia Jolicoeur-Martineau, Kilian Fatras, and Tal Kachman. Generating and imputing tabular data via diffusion and flow-based gradient-boosted trees. In *International conference on artificial intelligence and statistics*, pp. 1288–1296. PMLR, 2024.

Ilyes Khemakhem, Diederik Kingma, Ricardo Monti, and Aapo Hyvarinen. Variational autoencoders and nonlinear ica: A unifying framework. In *International conference on artificial intelligence and statistics*, pp. 2207–2217. PMLR, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Steven Cheng-Xian Li, Bo Jiang, and Benjamin Marlin. Misgan: Learning from incomplete data with generative adversarial networks. *arXiv preprint arXiv:1902.09599*, 2019.

Roderick J. A. Little and Donald B. Rubin. *Statistical Analysis with Missing Data*. John Wiley & Sons, Hoboken, NJ, 3 edition, 2019.

Qiao Liu and Wing Hung Wong. An ai-powered bayesian generative modeling approach for causal inference in observational studies. *Journal of the American Statistical Association*, 0(ja):1–20, 2026a. doi: 10.1080/01621459.2026.2654227.

Qiao Liu and Wing Hung Wong. A bayesian generative modeling approach for arbitrary conditional inference. *arXiv preprint arXiv:2601.05355*, 2026b.

Qiao Liu, Zhongren Chen, and Wing Hung Wong. An encoding generative modeling approach to dimension reduction and covariate adjustment in causal inference with observational studies. *Proceedings of the National Academy of Sciences*, 121(23):e2322376121, 2024.

Guyue Luo and Qiao Liu. Bgm-iv: an ai-powered bayesian generative modeling approach for instrumental variable analysis. *arXiv preprint arXiv:2605.07029*, 2026.

Chao Ma and Cheng Zhang. Identifiable generative models for missing not at random data imputation. *Advances in Neural Information Processing Systems*, 34:27645–27658, 2021.

Charles C Margossian and Lawrence K Saul. The shrinkage-delinkage trade-off: An analysis of factorized gaussian approximations for variational inference. In *Uncertainty in Artificial Intelligence*, pp. 1358–1367. PMLR, 2023.

Pierre-Alexandre Mattei and Jes Frellsen. Miwae: Deep generative modelling and imputation of incomplete data sets. In *International conference on machine learning*, pp. 4413–4423. PMLR, 2019.

Wang Miao, Lan Liu, Yilin Li, Eric J Tchetgen Tchetgen, and Zhi Geng. Identification and semiparametric efficiency theory of nonignorable missing data with a shadow variable. *ACM/JMS Journal of Data Science*, 1(2):1–23, 2024.

Robin Mitra, Sarah F McGough, Tapabrata Chakraborti, Chris Holmes, Ryan Copping, Niels Hagenbuch, Stefanie Biedermann, Jack Noonan, Brieuc Lehmann, Aditi Shenvi, et al. Learning from data with structured missingness. *Nature Machine Intelligence*, 5(1):13–23, 2023.

Boris Muzellec, Julie Josse, Claire Boyer, and Marco Cuturi. Missing data imputation using optimal transport. In *International Conference on Machine Learning*, pp. 7130–7140. PMLR, 2020.

Alfredo Nazabal, Pablo M Olmos, Zoubin Ghahramani, and Isabel Valera. Handling incomplete heterogeneous data using vaes. *Pattern Recognition*, 107:107501, 2020.

Radford M Neal. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, pp. 47–95, 2011.

Ignacio Peis, Chao Ma, and José Miguel Hernández-Lobato. Missing data imputation and acquisition with deep hierarchical models and hamiltonian monte carlo. *Advances in Neural Information Processing Systems*, 35:35839–35851, 2022.

Donald B Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

Donald B. Rubin. *Multiple Imputation for Nonresponse in Surveys*. Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., 1987. ISBN 9780471087052. doi: 10.1002/9780470316696. URL https://doi.org/10.1002/9780470316696.

Joseph L Schafer. *Analysis of incomplete multivariate data*. chapman and hall/CRC, 1997.

Joseph L Schafer and John W Graham. Missing data: our view of the state of the art. *Psychological methods*, 7(2):147, 2002.

Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in neural information processing systems*, 34:24804–24816, 2021.

Luke Tierney. Markov chains for exploring posterior distributions. *the Annals of Statistics*, pp. 1701–1728, 1994.

Stef Van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, 45:1–67, 2011.

Sheng Wang, Jun Shao, and Jae Kwang Kim. An instrumental variable approach for identification and estimation with nonignorable nonresponse. *Statistica Sinica*, pp. 1097–1116, 2014.

Samuel Von Wilson, Bogdan Cebere, James Myatt, and Samuel Wilson. miceforest, 2023. URL https://github.com/AnotherSamWilson/miceforest/.

Jinsung Yoon, James Jordon, and Mihaela Schaar. Gain: Missing data imputation using generative adversarial nets. In *International conference on machine learning*, pp. 5689–5698. PMLR, 2018.

Zhaoyang Zhang, Ziqi Chen, Qiao Liu, Jinhan Xie, and Hongtu Zhu. Sampling-guided heterogeneous graph neural network with temporal smoothing for scalable longitudinal data imputation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 3912–3920, 2025.

Shuhan Zheng and Nontawat Charoenphakdee. Diffusion models for missing value imputation in tabular data. *arXiv preprint arXiv:2210.17128*, 2022.

A Derivation of KL-weighted variational objectives

This section first derives the standard ELBOs for the untempered conditional posteriors in Eq. (10) from the conditional log-posteriors in Eq. (9). Since the derivations for θ and ϕ are fully parallel, we present both in the same notation.

Step 1: conditional posteriors. From Eq. (9), the conditional posteriors are, up to normalizing constants,

$$p(\theta \mid \mathbf{Z}, \mathbf{X}) \propto p_\theta(\mathbf{X} \mid \mathbf{Z})\, \pi(\theta), \qquad p(\phi \mid \mathbf{X}, \mathbf{R}) \propto p_\phi(\mathbf{R} \mid \mathbf{X})\, \pi(\phi). \tag{16}$$

Equivalently, introducing the corresponding marginal likelihoods,

$$p(\theta \mid \mathbf{Z}, \mathbf{X}) = \frac{p_\theta(\mathbf{X} \mid \mathbf{Z})\, \pi(\theta)}{p(\mathbf{X} \mid \mathbf{Z})}, \qquad p(\phi \mid \mathbf{X}, \mathbf{R}) = \frac{p_\phi(\mathbf{R} \mid \mathbf{X})\, \pi(\phi)}{p(\mathbf{R} \mid \mathbf{X})}, \tag{17}$$

where

$$p(\mathbf{X} \mid \mathbf{Z}) = \int p_\theta(\mathbf{X} \mid \mathbf{Z})\, \pi(\theta)\, d\theta, \qquad p(\mathbf{R} \mid \mathbf{X}) = \int p_\phi(\mathbf{R} \mid \mathbf{X})\, \pi(\phi)\, d\phi. \tag{18}$$

These marginal likelihoods are generally intractable because they require integrating over all network parameters.

Step 2: variational approximation for θ. Introduce a variational family $q_\lambda(\theta)$ to approximate $p(\theta \mid \mathbf{Z}, \mathbf{X})$. Starting from the log evidence, we write

$$\log p(\mathbf{X} \mid \mathbf{Z}) = \log \int q_\lambda(\theta) \frac{p_\theta(\mathbf{X} \mid \mathbf{Z})\, \pi(\theta)}{q_\lambda(\theta)}\, d\theta. \tag{19}$$

Applying Jensen's inequality gives

$$\log p(\mathbf{X} \mid \mathbf{Z}) \geq \int q_\lambda(\theta) \log \frac{p_\theta(\mathbf{X} \mid \mathbf{Z})\, \pi(\theta)}{q_\lambda(\theta)}\, d\theta \tag{20}$$

$$= \mathbb{E}_{q_\lambda(\theta)}[\log p_\theta(\mathbf{X} \mid \mathbf{Z})] + \mathbb{E}_{q_\lambda(\theta)}[\log \pi(\theta)] - \mathbb{E}_{q_\lambda(\theta)}[\log q_\lambda(\theta)]\,. \tag{21}$$

Using the definition of the KL divergence,

$$\mathrm{KL}\big(q_\lambda(\theta) \,\|\, \pi(\theta)\big) = \mathbb{E}_{q_\lambda(\theta)}[\log q_\lambda(\theta) - \log \pi(\theta)]\,, \tag{22}$$

we obtain the standard ELBO

$$\mathcal{L}_\theta^{\mathrm{std}} = \mathbb{E}_{q_\lambda(\theta)}[\log p_\theta(\mathbf{X} \mid \mathbf{Z})] - \mathrm{KL}\big(q_\lambda(\theta) \,\|\, \pi(\theta)\big), \tag{23}$$

with

$$\log p(\mathbf{X} \mid \mathbf{Z}) \geq \mathcal{L}_\theta^{\mathrm{std}}. \tag{24}$$

An equivalent decomposition, useful for interpretation, is

$$\log p(\mathbf{X} \mid \mathbf{Z}) = \mathcal{L}_\theta^{\mathrm{std}} + \mathrm{KL}\big(q_\lambda(\theta) \,\|\, p(\theta \mid \mathbf{Z}, \mathbf{X})\big), \tag{25}$$

which shows that maximizing the ELBO is equivalent to minimizing the KL divergence between the variational approximation and the exact conditional posterior.

Step 3: variational approximation for ϕ. The same argument applies to the missingness-model parameters. Let $q_\psi(\phi)$ approximate $p(\phi \mid \mathbf{X}, \mathbf{R})$. Then

$$\log p(\mathbf{R} \mid \mathbf{X}) = \log \int q_\psi(\phi) \frac{p_\phi(\mathbf{R} \mid \mathbf{X})\, \pi(\phi)}{q_\psi(\phi)}\, d\phi, \tag{26}$$

so Jensen's inequality yields

$$\log p(\mathbf{R} \mid \mathbf{X}) \geq \mathbb{E}_{q_\psi(\phi)}[\log p_\phi(\mathbf{R} \mid \mathbf{X})] + \mathbb{E}_{q_\psi(\phi)}[\log \pi(\phi)] - \mathbb{E}_{q_\psi(\phi)}[\log q_\psi(\phi)] \tag{27}$$

$$= \mathbb{E}_{q_\psi(\phi)}[\log p_\phi(\mathbf{R} \mid \mathbf{X})] - \mathrm{KL}\big(q_\psi(\phi) \,\|\, \pi(\phi)\big). \tag{28}$$

Thus the standard ELBO for ϕ is

$$\mathcal{L}_\phi^{\mathrm{std}} = \mathbb{E}_{q_\psi(\phi)}[\log p_\phi(\mathbf{R} \mid \mathbf{X})] - \mathrm{KL}\big(q_\psi(\phi) \,\|\, \pi(\phi)\big), \tag{29}$$

with the analogous decomposition

$$\log p(\mathbf{R} \mid \mathbf{X}) = \mathcal{L}_\phi^{\mathrm{std}} + \mathrm{KL}\big(q_\psi(\phi) \,\|\, p(\phi \mid \mathbf{X}, \mathbf{R})\big). \tag{30}$$

Step 4: KL-weighted variational objectives. For the global weight updates, we replace the unit KL coefficients in the standard ELBOs by a regularization weight $\tau \geq 0$:

$$\begin{aligned}\mathcal{L}_\theta &= \mathbb{E}_{q_\lambda(\theta)}\big[\log p_\theta(X \mid Z)\big] - \tau\,\mathrm{KL}\big(q_\lambda(\theta) \,\|\, \pi(\theta)\big),\\ \mathcal{L}_\phi &= \mathbb{E}_{q_\psi(\phi)}\big[\log p_\phi(R \mid X)\big] - \tau\,\mathrm{KL}\big(q_\psi(\phi) \,\|\, \pi(\phi)\big).\end{aligned} \tag{31}$$

When $\tau = 1$, these are the standard conditional ELBOs derived above. For general τ, we refer to them as KL-weighted variational objectives; in particular, $0 \leq \tau < 1$ does not generally preserve the lower-bound property for the original log evidence. Setting $\tau = 0$ removes the KL penalties but does not, by itself, reduce the variational networks to deterministic networks.

Step 5: Gaussian variational families. We use separate factorized Gaussian variational families for the two networks:

$$\begin{aligned}q_\lambda(\theta) &= \mathcal{N}\big(\theta; \mu_\lambda, \mathrm{diag}(\sigma_\lambda^2)\big)\,,\\ q_\psi(\phi) &= \mathcal{N}\big(\phi; \mu_\psi, \mathrm{diag}(\sigma_\psi^2)\big)\,,\end{aligned} \tag{32}$$

where μ_λ, μ_ψ and $\sigma_\lambda^2, \sigma_\psi^2$ are vectors of learnable means and positive variances, respectively. The optimized quantities are the variational parameters $\lambda = (\mu_\lambda, \sigma_\lambda^2)$ and $\psi = (\mu_\psi, \sigma_\psi^2)$. Both objectives are optimized using stochastic gradients with respect to these variational parameters.

B Theory assumptions and proofs

This section formalizes Theorem 1 as a result for an idealized profiled M-estimator. It does not assert that finite-epoch alternating optimization attains the required optimization accuracy, nor does it analyze finite-MCMC error or the optional BNN estimator.

Write $O = (\mathbf{X}_{\mathrm{obs}}, \mathbf{R})$ for a generic observed pair and let P_0 denote its data-generating distribution. For one realization $o = (\mathbf{x}_{\mathrm{obs}}, \mathbf{r})$, define

$$m_\beta(o; \theta, \phi) := \sup_{\mathbf{z}, \mathbf{x}_{\mathrm{mis}}} \Big\{ \log \pi(\mathbf{z}) + \log p_\theta(\mathbf{x} \mid \mathbf{z}) + \beta \log p_\phi(\mathbf{r} \mid \mathbf{x}) \Big\}, \qquad \mathbf{x} = (\mathbf{x}_{\mathrm{obs}}, \mathbf{x}_{\mathrm{mis}}), \tag{33}$$

with empirical and population criteria

$$L_n^\beta(\theta, \phi) = \frac{1}{n} \sum_{i=1}^n m_\beta(O_i; \theta, \phi), \qquad L^\beta(\theta, \phi) = \mathbb{E}_{P_0}\big[m_\beta(O; \theta, \phi)\big]. \tag{34}$$

To keep the asymptotic scaling standard, these criteria omit the $O(n^{-1})$ global prior terms $n^{-1} \log \pi(\theta)$ and $n^{-1} \log \pi(\phi)$.

For each (θ, ϕ), define the normalizing constant

$$C_{\theta,\phi,\beta} := \sum_{\mathbf{r} \in \{0,1\}^p} \int_{\mathbb{R}^p} \int \pi(\mathbf{z}) p_\theta(\mathbf{x} \mid \mathbf{z}) p_\phi(\mathbf{r} \mid \mathbf{x})^\beta \, d\mathbf{z}\, d\mathbf{x}, \tag{35}$$

and the associated completed-data law

$$q_{\theta,\phi}^\beta(\mathbf{x}, \mathbf{r}) := C_{\theta,\phi,\beta}^{-1} \int \pi(\mathbf{z}) p_\theta(\mathbf{x} \mid \mathbf{z}) p_\phi(\mathbf{r} \mid \mathbf{x})^\beta \, d\mathbf{z}. \tag{36}$$

At $\beta = 0$, the factor involving p_ϕ is defined to be absent. For discrete coordinates, the corresponding integrals are replaced by sums. Let $q_{\theta,\phi,O}^\beta$ be the observed-data marginal of (36), and let $K_{\theta,\phi}^\beta(d\mathbf{u} \mid o)$ be a regular conditional distribution of $\mathbf{X}_{\mathrm{mis}} = \mathbf{u}$ given $O = o$. The criterion in (33) profiles over $(\mathbf{z}, \mathbf{x}_{\mathrm{mis}})$, whereas (36) integrates over $\mathbf{z}$. They are therefore distinct objects. Assumption A6 below supplies the explicit profile-to-law link needed by the theorem. We now state the regularity conditions.

Assumption A1 (Sampling, measurability, and domination). The observed pairs $O_1, \ldots, O_n$ are i.i.d. from P_0. For every $(\theta, \phi) \in \Theta \times \Phi$, the map $O \mapsto m_\beta(O; \theta, \phi)$ is measurable and finite P_0-almost surely. There exists an integrable envelope M_β such that

$$\sup_{(\theta,\phi) \in \Theta \times \Phi} \big| m_\beta(O; \theta, \phi) \big| \leq M_\beta(O) \quad P_0\text{-a.s.}, \qquad \mathbb{E}_{P_0}\big[M_\beta(O)\big] < \infty. \tag{37}$$

The normalizing constant in (35) satisfies $0 < C_{\theta,\phi,\beta} < \infty$ for every (θ, ϕ).

Assumption A2 (Compact sieves, continuity, and existence of a maximizer). There exist compact sieve sets $\Theta_n \subset \Theta$ and $\Phi_n \subset \Phi$ such that $\Theta_1 \subset \Theta_2 \subset \cdots$, $\Phi_1 \subset \Phi_2 \subset \cdots$, and $\overline{\cup_{n\geq 1}(\Theta_n \times \Phi_n)} = \Theta \times \Phi$. For P_0-almost every O, the map $(\theta, \phi) \mapsto m_\beta(O; \theta, \phi)$ is continuous on each $\Theta_n \times \Phi_n$, and L^β extends continuously to a compactification on which its supremum is attained.

Assumption A3 (Uniform convergence of the profiled criterion). With

$$\omega_n := \sup_{(\theta,\phi)\in\Theta_n\times\Phi_n} \left|L_n^\beta(\theta, \phi) - L^\beta(\theta, \phi)\right|, \tag{38}$$

we have $\omega_n = O_p(a_n)$ for some deterministic sequence $a_n \downarrow 0$.

Assumption A4 (Sieve approximation error). Define

$$r_n := \sup_{(\theta,\phi)\in\Theta\times\Phi} L^\beta(\theta, \phi) - \sup_{(\theta,\phi)\in\Theta_n\times\Phi_n} L^\beta(\theta, \phi). \tag{39}$$

Then $r_n \to 0$.

Assumption A5 (Vanishing optimization error). The fitted estimator $(\hat\theta_n, \hat\phi_n) \in \Theta_n \times \Phi_n$ satisfies

$$\delta_n := \sup_{(\theta,\phi)\in\Theta_n\times\Phi_n} L_n^\beta(\theta, \phi) - L_n^\beta(\hat\theta_n, \hat\phi_n) = O_p(b_n), \qquad b_n \downarrow 0. \tag{40}$$

Assumption A6 (Profile-to-law separation). The maximizer set

$$\mathcal{M}_\beta^* := \arg\max_{(\theta,\phi)\in\Theta\times\Phi} L^\beta(\theta, \phi) \tag{41}$$

is nonempty, and every element of $\mathcal{M}_\beta^*$ induces the same completed-data law, denoted by q_β^*. For two completed-data laws with densities q_1 and q_2, define

$$\|q_1 - q_2\|_{\mathrm{TV}} := \frac{1}{2} \sum_{\mathbf{r}\in\{0,1\}^p} \int_{\mathbb{R}^p} |q_1(\mathbf{x}, \mathbf{r}) - q_2(\mathbf{x}, \mathbf{r})|\, d\mathbf{x}. \tag{42}$$

For every $\varepsilon \in (0, 1]$, assume

$$\Delta_\beta(\varepsilon) := L^{\beta,*} - \sup_{\{(\theta,\phi):\, \|q_{\theta,\phi}^\beta - q_\beta^*\|_{\mathrm{TV}} \geq \varepsilon\}} L^\beta(\theta, \phi) > 0, \qquad L^{\beta,*} := \sup_{\theta,\phi} L^\beta(\theta, \phi). \tag{43}$$

Set $\Delta_\beta(0) = 0$ and define the upper generalized inverse

$$\Delta_\beta^\dagger(t) := \sup\Big(\{\varepsilon \in [0, 1] : \Delta_\beta(\varepsilon) \leq t\} \cup \{0\}\Big). \tag{44}$$

The function Δ_β is nondecreasing, and (44), rather than an infimum over the same sublevel set, is the inverse needed for an upper distance bound. Assumption A6 is a criterion-level profile-to-law identification condition: it explicitly supplies the bridge between the profiled objective and the integrated law. It is not implied by correct model specification or by generic MNAR assumptions.

Assumption A7 (Moment envelope; only for posterior means). Let $q_{\beta,O}^*$ be the observed-data marginal of q_β^* and $K_\beta^*(d\mathbf{u} \mid o)$ its missing-value conditional kernel. There exist $\rho > 0$ and a nonnegative function G with $\int G(o) q_{\beta,O}^*(do) < \infty$ such that, for every (θ, ϕ) satisfying $L^{\beta,*} - L^\beta(\theta, \phi) \leq \rho$ and for $q_{\beta,O}^*$-almost every o,

$$\int \|\mathbf{u}\|_2^2 K_{\theta,\phi}^\beta(d\mathbf{u} \mid o) + \int \|\mathbf{u}\|_2^2 K_\beta^*(d\mathbf{u} \mid o) \leq G(o). \tag{45}$$

This condition permits unbounded Gaussian support; it requires the near-optimal conditional second moments to be dominated, which is compatible with bounded conditional means and variances but is not automatic for unrestricted neural decoders.

Assumption A8 (Optional true-law Fisher consistency). This assumption is not used for the pseudo-true result. For the specialization $\beta = 1$, suppose the true completed-data law q_0 belongs to the model and every population maximizer of the profiled criterion induces q_0:

$$(\theta, \phi) \in \mathcal{M}_1^* \quad \Longrightarrow \quad q^1_{\theta,\phi} = q_0. \tag{46}$$

Condition (46) combines a model-specific MNAR identification requirement with Fisher consistency of the profiled, rather than integrated, criterion. It does not follow from correct specification alone.

Theorem 1. Under Assumptions A1–A6, let

$$T_n := 2\omega_n + r_n + \delta_n. \tag{47}$$

Then

$$L^{\beta,*} - L^{\beta}(\hat{\theta}_n, \hat{\phi}_n) \le T_n = o_p(1), \tag{48}$$

$$\big\| q^{\beta}_{\hat{\theta}_n,\hat{\phi}_n} - q^*_\beta \big\|_{\mathrm{TV}} \le \Delta^\dagger_\beta(T_n) = o_p(1), \tag{49}$$

and

$$\int \big\| K^{\beta}_{\hat{\theta}_n,\hat{\phi}_n}(\cdot \mid o) - K^*_\beta(\cdot \mid o) \big\|_{\mathrm{TV}} q^*_{\beta,O}(do) \le 2 \big\| q^{\beta}_{\hat{\theta}_n,\hat{\phi}_n} - q^*_\beta \big\|_{\mathrm{TV}} = o_p(1). \tag{50}$$

Corollary A1 (Polynomial-margin rate). If, for some $c > 0$, $\kappa > 0$, and $\varepsilon_0 > 0$,

$$\Delta_\beta(\varepsilon) \ge c\varepsilon^\kappa, \qquad 0 < \varepsilon \le \varepsilon_0, \tag{51}$$

then

$$\big\| q^{\beta}_{\hat{\theta}_n,\hat{\phi}_n} - q^*_\beta \big\|_{\mathrm{TV}} = O_p\big((a_n + r_n + b_n)^{1/\kappa}\big). \tag{52}$$

Corollary A2 (Posterior means under a moment envelope). Under A1–A7, define

$$\hat{\mu}_n(o) := \int \mathbf{u}\, K^{\beta}_{\hat{\theta}_n,\hat{\phi}_n}(d\mathbf{u} \mid o), \qquad \mu^*_\beta(o) := \int \mathbf{u}\, K^*_\beta(d\mathbf{u} \mid o). \tag{53}$$

Then

$$\int \|\hat{\mu}_n(o) - \mu^*_\beta(o)\|_2^2 q^*_{\beta,O}(do) = o_p(1). \tag{54}$$

Consequently, for

$$\mathcal{R}^*_\beta(\mu) := \mathbb{E}_{q^*_\beta}\big[\|\mathbf{X}_{\mathrm{mis}} - \mu(O)\|_2^2\big], \tag{55}$$

we have

$$\mathcal{R}^*_\beta(\hat{\mu}_n) - \mathcal{R}^*_\beta(\mu^*_\beta) = o_p(1). \tag{56}$$

Corollary A3 (Optional true-law specialization). If A8 additionally holds at $\beta = 1$, then $q^*_1 = q_0$ and $K^*_1(\cdot \mid O)$ is the true missing-value conditional law. This specialization is deliberately conditional on (46); the unrestricted MissBGM model is not claimed to identify an arbitrary MNAR law.

Lemma A1 (Population near-optimality of the fitted estimator). Under Assumptions A3–A5,

$$\sup_{(\theta,\phi)\in\Theta\times\Phi} L^{\beta}(\theta,\phi) - L^{\beta}(\hat{\theta}_n, \hat{\phi}_n) \le 2\omega_n + r_n + \delta_n. \tag{57}$$

Proof. By the definition of r_n,

$$\sup_{(\theta,\phi)\in\Theta\times\Phi} L^{\beta}(\theta,\phi) \le \sup_{(\theta,\phi)\in\Theta_n\times\Phi_n} L^{\beta}(\theta,\phi) + r_n.$$

By the definition of ω_n,

$$\sup_{(\theta,\phi)\in\Theta_n\times\Phi_n} L^{\beta}(\theta,\phi) \le \sup_{(\theta,\phi)\in\Theta_n\times\Phi_n} L^{\beta}_n(\theta,\phi) + \omega_n.$$

By the definition of δ_n,

$$\sup_{(\theta,\phi)\in\Theta_n\times\Phi_n} L^{\beta}_n(\theta,\phi) \le L^{\beta}_n(\hat{\theta}_n, \hat{\phi}_n) + \delta_n.$$

Applying the definition of ω_n once more,

$$L^{\beta}_n(\hat{\theta}_n, \hat{\phi}_n) \le L^{\beta}(\hat{\theta}_n, \hat{\phi}_n) + \omega_n.$$

Combining the four displays gives (57). □

Lemma A2 (Completed-data-law convergence). Under Assumptions A3–A6, with T_n as in (47),

$$\left\| q^{\beta}_{\hat{\theta}_n,\hat{\phi}_n} - q^*_\beta \right\|_{\mathrm{TV}} \le \Delta^\dagger_\beta(T_n) = o_p(1). \tag{58}$$

If the polynomial margin (51) holds, the rate in (52) follows.

Proof. Let $d_n := \|q^{\beta}_{\hat{\theta}_n,\hat{\phi}_n} - q^*_\beta\|_{\mathrm{TV}}$. Because $(\hat{\theta}_n, \hat{\phi}_n)$ belongs to the set over which the second supremum in (43) is taken at $\varepsilon = d_n$,

$$\Delta_\beta(d_n) \le L^{\beta,*} - L^\beta(\hat{\theta}_n, \hat{\phi}_n) \le T_n. \tag{59}$$

Thus d_n belongs to the sublevel set in (44), proving $d_n \le \Delta^\dagger_\beta(T_n)$. For any fixed $\varepsilon > 0$, monotonicity gives

$$\mathbb{P}(d_n \ge \varepsilon) \le \mathbb{P}\{T_n \ge \Delta_\beta(\varepsilon)\} \longrightarrow 0,$$

because $T_n = o_p(1)$ and $\Delta_\beta(\varepsilon) > 0$. This proves (58). Notice that monotonicity alone does not justify replacing the random bound by $O_p\{\Delta^\dagger_\beta(a_n + r_n + b_n)\}$. Under (51), however, (59) yields $d_n \le (T_n/c)^{1/\kappa}$ with probability tending to one, and $T_n = O_p(a_n + r_n + b_n)$ gives (52). □

Lemma A3 (Integrated conditional-law convergence). For any two completed-data laws q_1, q_2 with observed-data marginals $q_{1,O}, q_{2,O}$ and missing-value kernels K_1, K_2,

$$\int \|K_1(\cdot \mid o) - K_2(\cdot \mid o)\|_{\mathrm{TV}} q_{2,O}(do) \le 2\|q_1 - q_2\|_{\mathrm{TV}}. \tag{60}$$

Proof. Write $q_j(d\mathbf{u}, do) = K_j(d\mathbf{u} \mid o) q_{j,O}(do)$. Using densities for readability, the triangle inequality gives

$$\begin{aligned}
&\int q_{2,O}(o) \int |k_1(\mathbf{u} \mid o) - k_2(\mathbf{u} \mid o)| \, d\mathbf{u} \, do \\
&\quad \le \int |q_{2,O}(o) - q_{1,O}(o)| \, do + \int |q_1(\mathbf{u}, o) - q_2(\mathbf{u}, o)| \, d\mathbf{u} \, do \\
&\quad \le 2 \int |q_1(\mathbf{u}, o) - q_2(\mathbf{u}, o)| \, d\mathbf{u} \, do,
\end{aligned} \tag{61}$$

where marginalization is an L^1 contraction. Dividing by 2 on the left and using $\|q_1 - q_2\|_1 = 2\|q_1 - q_2\|_{\mathrm{TV}}$ proves (60). The argument extends verbatim to general probability measures by disintegration. □

Lemma A4 (Posterior-mean risk under the moment envelope). Under Assumption A7 and the conclusions of Theorem 1,

$$\int \|\hat{\mu}_n(o) - \mu^*_\beta(o)\|_2^2 q^*_{\beta,O}(do) = o_p(1), \qquad \mathcal{R}^*_\beta(\hat{\mu}_n) - \mathcal{R}^*_\beta(\mu^*_\beta) = o_p(1). \tag{62}$$

Proof. Let

$$\eta_n(o) := \left\| K^{\beta}_{\hat{\theta}_n,\hat{\phi}_n}(\cdot \mid o) - K^*_\beta(\cdot \mid o) \right\|_{\mathrm{TV}}.$$

By Lemma A1, $(\hat{\theta}_n, \hat{\phi}_n)$ lies in the ρ-near-maximizer set of A7 with probability tending to one. On this event, take a maximal coupling $(\mathbf{U}_n, \mathbf{U}_*)$ of the two conditional laws at o, so that $\mathbb{P}(\mathbf{U}_n \ne \mathbf{U}_* \mid o) = \eta_n(o)$. Conditional Cauchy–Schwarz and (45) give

$$\begin{aligned}
\|\hat{\mu}_n(o) - \mu^*_\beta(o)\|_2^2 &\le \mathbb{E}[\|\mathbf{U}_n - \mathbf{U}_*\|_2 \mathbf{1}\{\mathbf{U}_n \ne \mathbf{U}_*\} \mid o]^2 \\
&\le 2G(o)\eta_n(o).
\end{aligned} \tag{63}$$

For any $M > 0$,

$$\int G(o)\eta_n(o) q^*_{\beta,O}(do) \le M \int \eta_n(o) q^*_{\beta,O}(do) + \int_{\{G > M\}} G(o) q^*_{\beta,O}(do).$$

The first term is $o_p(1)$ by Lemmas A2–A3, and the second tends to zero as $M \to \infty$ because G is integrable. This proves the first claim in (62). Finally, conditional-mean orthogonality under q^*_β gives

$$\mathcal{R}^*_\beta(\hat{\mu}_n) - \mathcal{R}^*_\beta(\mu^*_\beta) = \int \|\hat{\mu}_n(o) - \mu^*_\beta(o)\|_2^2 q^*_{\beta,O}(do),$$

which proves the second claim. □

Proof of Theorem 1 and the corollaries. Lemma A1 proves (48); Lemma A2 proves (49) and Corollary A1; and Lemma A3 applied to $(q^\beta_{\hat{\theta}_n,\hat{\phi}_n}, q^*_\beta)$ proves (50). Lemma A4 proves Corollary A2. Under A8, every maximizer at $\beta = 1$ induces q_0, so A6 has $q^*_1 = q_0$ and Corollary A3 follows. □

Scope of the result. The theorem concerns the normalized target in (36), not the empirical distribution of a fixed number of MCMC draws. It also treats A5 as an assumption rather than a proved property of alternating Adam updates. For $\beta > 0$ and an unpenalized deterministic ϕ block, multiplying its log-likelihood by β does not change its exact block maximizers; nevertheless, it can change finite optimization trajectories and does not establish A5. In the BNN variant, λ replaces θ as the optimized generator quantity and the KL-weighted stochastic decoder induces a different law; a separate variational asymptotic analysis is therefore required. Most importantly, correct specification at $\beta = 1$ places q_0 in the model but does not establish A8: profiling observation-specific latent and missing variables need not be Fisher-consistent for the integrated law.

C Datasets

We evaluate MissBGM on a synthetic benchmark and four real-world tabular datasets that span a wide range of sample sizes and feature dimensions. This appendix details the data-generating processes, the missingness mechanisms, and the preprocessing protocol shared across all methods.

C.1 Synthetic dataset

The synthetic benchmark uses a conditional Gaussian complete-data model with closed-form conditional means, variances, and prediction intervals. We combine this model with self-masked MNAR for point-imputation experiments and an independent MCAR mask for the primary uncertainty-calibration experiment.

Anchor and target partition. Each sample $\mathbf{x}_i \in \mathbb{R}^p$ is partitioned into a fully observed *anchor* block of size p_a and a potentially missing *target* block of size $p_t = p - p_a$. We index the two blocks by the disjoint coordinate sets

$$\mathcal{J}_a = \{1, \ldots, p_a\}, \qquad \mathcal{J}_t = \{p_a + 1, \ldots, p\}, \tag{64}$$

and write the corresponding subvectors as $\mathbf{x}_{i,\mathcal{J}_a} = (x_{ij})_{j\in\mathcal{J}_a} \in \mathbb{R}^{p_a}$ and $\mathbf{x}_{i,\mathcal{J}_t} = (x_{ij})_{j\in\mathcal{J}_t} \in \mathbb{R}^{p_t}$. The full sample is then assembled by concatenation,

$$\mathbf{x}_i = \left(\mathbf{x}_{i,\mathcal{J}_a}, \mathbf{x}_{i,\mathcal{J}_t}\right) \in \mathbb{R}^p. \tag{65}$$

Throughout the synthetic experiments we set $p_a = 5$, so that $p_t = p - 5$.

Global parameters. The model has three population-level parameters $(B, \mathbf{b}, \boldsymbol{\sigma})$ that are sampled *once* per simulation replicate from a fixed seed and then shared across all n data points,

$$B \in \mathbb{R}^{p_a \times p_t}, \;\; B_{kl} \overset{\text{iid}}{\sim} \mathcal{N}(0,\, 0.4^2); \quad \mathbf{b} \in \mathbb{R}^{p_t}, \;\; b_l \overset{\text{iid}}{\sim} \mathcal{N}(0,\, 0.3^2); \quad \sigma_l \overset{\text{iid}}{\sim} \text{Uniform}(0.6,\, 1.2), \tag{66}$$

for $k = 1, \ldots, p_a$ and $l = 1, \ldots, p_t$, with $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_{p_t})^\top$. To match the per-target-coordinate notation used below, we define $\sigma_{(j)} \equiv \sigma_{j-p_a}$ and $b_{(j)} \equiv b_{j-p_a}$ for any target index $j \in \mathcal{J}_t$, and let $B_{:,(j)} \in \mathbb{R}^{p_a}$ denote the column of B corresponding to target coordinate j.

Per-sample data generation. Conditional on $(B, \mathbf{b}, \boldsymbol{\sigma})$, each of the n samples is generated independently as

$$\mathbf{x}_{i,\mathcal{J}_a} \sim \mathcal{N}(\mathbf{0},\, I_{p_a}), \qquad \mathbf{x}_{i,\mathcal{J}_t} \mid \mathbf{x}_{i,\mathcal{J}_a} \sim \mathcal{N}\big(B^\top \mathbf{x}_{i,\mathcal{J}_a} + \mathbf{b},\, \mathrm{diag}(\boldsymbol{\sigma}^2)\big)\,, \tag{67}$$

where $\mathrm{diag}(\boldsymbol{\sigma}^2) \in \mathbb{R}^{p_t \times p_t}$ is the diagonal matrix with entries $\sigma_1^2, \ldots, \sigma_{p_t}^2$. Equivalently, defining

$$\mu_j(\mathbf{x}_{i,\mathcal{J}_a}) \;=\; B_{:,(j)}^\top \mathbf{x}_{i,\mathcal{J}_a} + b_{(j)}, \qquad j \in \mathcal{J}_t, \tag{68}$$

the marginal conditional law of each target coordinate is the heteroscedastic Gaussian

$$x_{ij} \mid \mathbf{x}_{i,\mathcal{J}_a} \;\sim\; \mathcal{N}\Big(\mu_j(\mathbf{x}_{i,\mathcal{J}_a}),\, \sigma_{(j)}^2\Big)\,, \qquad j \in \mathcal{J}_t. \tag{69}$$

Self-masked MNAR rule. Anchor coordinates are always observed: $r_{ij} = 1$ for all $j \in \mathcal{J}_a$. For each target coordinate $j \in \mathcal{J}_t$, the entry is observed if and only if its realization falls below a quantile threshold of its own conditional distribution,

$$r_{ij} = \mathbb{I}\big\{x_{ij} \le \mu_j(\mathbf{x}_{i,\mathcal{J}_a}) + \sigma_{(j)}\kappa\big\}\,, \qquad \kappa = \Phi^{-1}(1-r), \tag{70}$$

where Φ is the standard normal cumulative distribution function (CDF) and $r \in (0,1)$ is the targeted per-column missing rate on the target block. Because the threshold depends on the value of x_{ij} itself, larger realizations of x_{ij} are systematically more likely to be missing, so the masking mechanism is non-ignorable (MNAR).

Effect of missing rate r. This self-masked construction also explains why the RMSE in Figure 2(b) is non-monotonic in r. When r is small, $\kappa = \Phi^{-1}(1-r)$ is large, so only the most extreme right-tail realizations of x_{ij} are masked; these atypical values lie far from the conditional mean $\mu_j(\mathbf{x}_{i,\mathcal{J}_a})$ and are intrinsically hard to recover even when abundant observed entries are available for training. When r is large, the threshold falls into the bulk of the conditional distribution and a large fraction of target entries are missing, leaving too few observations to fit the conditional model reliably. The hardest regimes therefore arise at both ends of the missing-rate spectrum, while moderate missing rates strike a balance between informativeness of the observed entries and severity of the tail truncation, which yields the lowest imputation RMSE in Figure 2(b).

Oracle posterior and prediction interval. For the uncertainty-quantification experiment in Table 2, we retain the complete-data process in Eq. (67) but replace the self-masked rule with independent MCAR masking. Anchor coordinates remain fully observed, while each target indicator is generated as

$$r_{ij} \overset{\text{iid}}{\sim} \mathrm{Bernoulli}(1-r), \qquad j \in \mathcal{J}_t, \qquad \mathbf{R} \perp\!\!\!\perp \mathbf{X}. \tag{71}$$

Consequently, conditioning on a target entry being missing does not truncate its Gaussian conditional distribution:

$$x_{ij} \mid \mathbf{x}_{i,\mathcal{J}_a},\, r_{ij} = 0 \sim \mathcal{N}\Big(\mu_j(\mathbf{x}_{i,\mathcal{J}_a}),\, \sigma_{(j)}^2\Big)\,. \tag{72}$$

The oracle posterior mean and standard deviation are therefore $\mu_j(\mathbf{x}_{i,\mathcal{J}_a})$ and $\sigma_{(j)}$, respectively. The oracle $(1-\alpha)$ prediction interval is

$$\mathrm{PI}_{ij}^{1-\alpha,\star} = \Big[\mu_j + \sigma_{(j)}\Phi^{-1}\Big(\frac{\alpha}{2}\Big)\,,\; \mu_j + \sigma_{(j)}\Phi^{-1}\Big(1-\frac{\alpha}{2}\Big)\Big]\,, \tag{73}$$

where $\mu_j = \mu_j(\mathbf{x}_{i,\mathcal{J}_a})$. Its width is $2\sigma_{(j)}\Phi^{-1}(1-\alpha/2)$, and the average oracle width is the denominator of WR in Table 2. We use $n = 5000$, $p = 50$, target-block missing rate $r = 0.2$, and $\alpha = 0.05$ for this experiment. This MCAR benchmark separates posterior calibration from the identification and tail-extrapolation difficulties associated with deterministic MNAR. Under the self-masked MNAR rule in Eq. (70), all evaluated methods substantially undercover, although MissBGM attained the highest coverage. We therefore treat the MNAR result as a stress-test boundary rather than evidence of calibrated uncertainty under general MNAR.

Table 5: Summary of the four real-world tabular datasets used in our benchmark. All datasets are obtained from the UCI Machine Learning Repository. The reported #Samples and #Features are after dropping ID columns, target columns, and rows with native missing values where applicable.

Dataset	UCI ID	#Samples	#Features	Domain
Wine	109	178	13	Chemistry / agriculture
Concrete	165	1,030	8	Civil engineering
Breast	15	683	9	Biomedical
Gisette	170	13,500	5,000	Computer vision (digits)

C.2 Real-world datasets

We use four publicly available tabular datasets from the UCI Machine Learning Repository (`https://archive.ics.uci.edu/`) as our real-world benchmark. They vary in sample size and feature dimension from small-n low-dimensional regression problems to a very high-dimensional dataset. A summary of the datasets is given in Table 5.

Wine contains $n = 178$ chemical analyses of wines grown in the same region in Italy but derived from three different cultivars. We drop the class label column and use the remaining $p = 13$ continuous chemical attributes (alcohol, malic acid, flavanoids, color intensity, etc.) as the imputation target.

Concrete Compressive Strength contains $n = 1,030$ concrete mixture records with $p = 8$ continuous predictors describing the mixture composition (cement, slag, fly ash, water, superplasticizer, coarse and fine aggregates, age). We drop the target compressive strength column and treat the remaining columns as features.

Breast Cancer Wisconsin (Original) contains digitized cytology measurements with $p = 9$ ordinal features on a 1–10 scale (clump thickness, uniformity of cell size and shape, marginal adhesion, etc.). We drop the sample-code-number ID column, remove the 16 rows that contain native "?" values, and use the resulting $n = 683$ samples.

Gisette is a high-dimensional binary classification benchmark constructed from handwritten digit images. We stack the official train, validation, and unlabeled test splits to obtain $n = 13{,}500$ samples with $p = 5{,}000$ integer-valued pixel/aggregate features. Gisette serves as an ideal stress test for whether imputers remain accurate and scalable when the feature dimension is large.

Preprocessing. Before applying any imputation method, the datasets are normalized feature-wise using `StandardScaler` from `scikit-learn`. All RMSE numbers reported in the main paper are therefore on the standardized scale, which is comparable across features and across datasets.

MNAR mask generation. Real-world datasets do not come with ground-truth missing entries that allow us to evaluate imputation accuracy. We therefore inject controlled MNAR missingness on top of the full data matrix, so that the original observed values can serve as the ground truth for the masked entries. Specifically, for each dataset we form a self-masked logistic mechanism that depends on both each feature's own value and on a nonlinear summary of the full feature vector. For each row i and feature j, the score is

$$s_{ij} = 0.6\,\widetilde{x}_{ij} + 0.4\,\tanh\big(\widetilde{\mathbf{x}}_i^\top W\big)_j, \qquad W \in \mathbb{R}^{p\times p},\;\; W_{kj} \sim \mathcal{N}(0,\, 0.3^2/p), \tag{74}$$

where $\widetilde{\mathbf{x}}_i$ is the standardized row. The observation probability is $\mathbb{P}(r_{ij} = 1 \mid \mathbf{x}_i) = \mathrm{sig}(s_{ij} + \beta_0)$, with the intercept β_0 calibrated by bisection so that the average per-entry observation rate matches the target (e.g., 80% for missing rate 0.2). To avoid degenerate corner cases we additionally enforce that every row has at least one observed entry and every column has at least one observed entry. The resulting mask is non-ignorable (MNAR) because the observation probability depends directly on the target value itself. The positive self-term induces a lower-tail missingness tendency, while the nonlinear term modulates the observation probability according to the multivariate context.

D BASELINE METHODS

We compare MissBGM against nine imputation baselines that span classical statistical methods, optimal-transport-based methods, ensemble multiple imputation, modern deep generative models, and MNAR-aware latent-variable models. To ensure a fair comparison, every baseline operates on exactly the same standardized incomplete matrix described in Appendix C.2, sees exactly the same missingness mask as MissBGM, and uses the same random seed for shared randomness from the same data realization. All baselines are evaluated on the standardized scale, so that RMSE numbers are directly comparable across methods.

Mean. Mean imputation replaces each missing entry by the mean of the corresponding feature computed over its observed entries. It is parameter-free, computationally efficient, and serves as the simplest baseline for imputation. We use a `numpy`-based implementation that operates directly on the standardized matrix.

Optimal Transport (OT) OT-based imputation casts missing data completion as the problem of minimizing the entropy-regularized Sinkhorn divergence between random mini-batches of the (partially imputed) data, so that the imputed values produce a self-consistent empirical distribution. We use the `OTimputer` implementation from the ForestDiffusion benchmark codebase. The regularization parameter ε is selected automatically using the authors' `pick_epsilon` heuristic with quantile 0.5 and scale factor 0.05. Each imputation runs the Sinkhorn loop for 3000 iterations with batch size 128 and Adam learning rate 10^{-2}. To produce K multiple imputations ($K = 5$ as default unless otherwise specified), we re-run the procedure K times with independent random seeds and aggregate the results by averaging as the point estimate.

Iterative Conditional Equations (ICE) ICE (also known as MICE-PMM in its multiple-imputation form) repeatedly regresses each partially observed feature on all other features in a round-robin fashion, drawing imputed values from the posterior predictive distribution of the regressor. We use `IterativeImputer` from `scikit-learn` library, a maximum of 10 outer round-robin iterations, and Bayesian Ridge regressors as the per-feature conditional models.

Miceforest Miceforest implements multiple imputation by chained equations using LightGBM gradient-boosted trees as the per-feature conditional models, which makes it well suited to nonlinear feature dependencies in tabular data. We use the official `miceforest.ImputationKernel` as called from the ForestDiffusion driver: `datasets=K` parallel imputation chains seeded by a shared `random_state`. The K completed datasets are then aggregated by averaging as the point estimate. $K = 5$ in point estimate experiments and $K = 1000$ for uncertainty quantification on the synthetic benchmark.

GAIN GAIN formulates imputation as a conditional generative adversarial learning problem: a generator network completes the missing entries while a discriminator, given a hint mask, tries to identify which entries were imputed. We use the original GAIN implementation with the default hyperparameters: batch size 128, hint rate 0.9, reconstruction-loss weight $\alpha = 100$, and $10{,}000$ training iterations. The trained generator is then sampled K times based on different latent variables to produce K multiple imputations. $K = 5$ in point estimate experiments and $K = 1000$ for uncertainty quantification on the synthetic benchmark.

ForestDiffusion ForestDiffusion adapts score-based diffusion to tabular data by replacing the score neural network with a tree-based ensemble that learns to predict the noise injected at each diffusion step. Imputation is performed by running the reverse-time diffusion from Gaussian noise while clamping the observed entries to their true values at every denoising step. We use the default tabular configuration: an XGBoost backbone (`forest_model=xgboost`) under the variance-preserving SDE (`diffusion_type=vp`), with $n_t = 50$ diffusion time steps for both training and sampling, $\beta_{\min} = 0.1$, $\beta_{\max} = 8.0$, $\varepsilon = 10^{-3}$, max tree depth 7, 100 trees per step, learning rate 0.3, and a duplication factor of 100 for the noise/time pair augmentation that ForestDiffusion uses to stabilize training. Sampling produces K completed matrices that are then averaged. $K = 5$ in point estimate experiments and $K = 1000$ for uncertainty quantification on the synthetic benchmark.

TabCSDI TabCSDI is a conditional score-based diffusion model adapted to tabular data, in which the score network is a transformer trained to denoise the missing entries conditioned on the observed ones. Imputation is performed by running the conditional reverse diffusion to draw posterior samples of the missing entries. We use the official TabCSDI implementation, retaining the default network architecture, training schedule, and noise schedule, and configure the model with one numerical-feature group covering all standardized columns. K posterior completions are sampled per test matrix and averaged. $K = 5$ in point estimate experiments and $K = 1000$ for uncertainty quantification on the synthetic benchmark.

not-MIWAE not-MIWAE augments MIWAE with a learned Bernoulli model for the observation mask and optimizes an importance-weighted bound on the joint data–mask likelihood. We use the authors' TensorFlow implementation with a standard-normal latent prior, a diagonal-Gaussian encoder, and a Gaussian observation model. The encoder and decoder each contain two 128-unit tanh hidden layers, and the latent dimension is $p - 1$. We use the implementation's nonlinear missingness option, which maps the completed feature vector through a 128-unit tanh layer to feature-wise Bernoulli logits. Missing entries are set to zero at the encoder input, while the observed mask enters the joint likelihood explicitly. We train with Adam at learning rate 10^{-3} for $100{,}000$ mini-batch updates, using batch size 16 and 20 importance samples without regularization.

GINA GINA combines an identifiable VAE with a Bernoulli missingness network and an auxiliary-conditioned latent prior. We use the official `MNARPartialVAE` implementation from `project-azua` and retain its default continuous-data configuration. The observed mask is used as the auxiliary input to the prior network. The model uses 20-dimensional feature and set embeddings, a two-dimensional latent space, 10-unit tanh encoder and decoder layers, and a Gaussian decoder variance of 0.01. The missingness network has one 10-unit tanh hidden layer, receives both the completed data and latent representation, and has loss coefficient 1. We train the importance-weighted objective for $5{,}000$ epochs with Adam, batch size 100, and learning rate 10^{-3}.

E Experimental details

This appendix section provides additional experimental details regarding model training, testing, hyperparameters, and environment. To ensure a fair comparison across methods, we control the randomness in each dataset so that MissBGM and all baselines are trained and evaluated on exactly the same observed entries and missingness mask under the same random seed.

E.1 Model architecture

Data-generating network g_θ. The conditional distribution $p_\theta(\mathbf{x} \mid \mathbf{z})$ is parameterized by a fully connected network (FCN) that maps $\mathbf{z} \in \mathbb{R}^d$ to a diagonal Gaussian over $\mathbf{x} \in \mathbb{R}^p$: it outputs a mean vector $\boldsymbol{\mu}_\theta(\mathbf{z})$ and a positive variance vector $\boldsymbol{\sigma}^2_\theta(\mathbf{z})$ (softplus on the variance head plus a small floor). Hidden layers use LeakyReLU ($\alpha = 0.2$) after each dense layer, batch normalization on the input to the first hidden block, and L2 weight decay (10^{-4}) on kernels and biases. g_θ has five layers and each layer has 120 hidden units as default.

Missingness network f_ϕ. The Bernoulli logits for $\mathbf{r} \mid \mathbf{x}$ are produced by a separate FCN with the same LeakyReLU/L2 structure as above. f_ϕ has two layers and each layer has 64 hidden units as default.

Optional Bayesian neural networks for g_θ and f_ϕ. We provide an optional Bayesian neural network (BNN) implementation to explicitly account for model-parameter uncertainty for g_θ. In this variant, deterministic dense layers are replaced by `DenseFlipout` layers from TensorFlow Probability. `DenseFlipout` performs variational Bayes with weight posteriors by sampling pseudo-independent sign-perturbations per example (Flipout), which yields an unbiased stochastic gradient estimator with substantially reduced gradient variance compared with naive shared-weight reparameterization. In practice, this variance reduction makes optimization more stable on mini-batches, improves training signal-to-noise, and potentially provides better calibrated uncertainty in predictions when posterior weight samples are propagated through the imputation pipeline. Note that this option introduces variational weight uncertainty, which is not covered by Theorem 1.

E.2 EGM INITIALIZATION

Before the alternating stochastic optimization, we use an encoding generative modeling (EGM) stage (Liu et al., 2024) to obtain a stable warm start for both network parameters and latent representations. The key idea is to additionally introduce an encoder network for pretraining a coupled encoder-generator pair so that latent samples and reconstructed data are already approximately aligned in distribution, instead of starting from random weights and random latent spaces. Specifically, we first apply a simple KNN imputer with five nearest neighbors to fill the missing entries, then an auxiliary encoder network $e(\mathbf{x})$ is introduced to map from the complete data space to the latent space, and a generator network $g(\mathbf{z})$ is introduced to map the latent samples back to the data space $\mathbf{x}$. The encoder and generator are trained jointly to minimize the adversarial loss and the cycle-consistency loss. Through adversarial matching, we desire that $e(\mathbf{x})$ should match the prior distribution of the latent variables $\pi(\mathbf{z})$, which is a standard normal distribution. The cycle-consistency loss encourages that $g(e(\mathbf{x}))$ should be close to $\mathbf{x}$.

Conceptually, this EGM stage encourages the encoder and generator to learn a coherent inverse relationship, where the encoder can be seen as a shortcut function as opposed to Bayesian inference of latent variables. The EGM stage is only used to initialize the model parameters and latent variables, and the encoder $e(\mathbf{x})$ will be discarded during the alternating stochastic optimization.

We only run EGM initialization for $1{,}500$ mini-batches as default, which can be typically done within 1 minute. Then we initialize each sample-specific latent variable with a forward encoder pass,

$$\mathbf{Z}^{(0)} = e(\mathbf{X}), \tag{75}$$

and use these latents together with the pretrained generator as the starting point of Algorithm 1. Empirically, this warm start improves optimization stability, speeds up convergence of the alternating updates for $(\mathbf{Z}, \mathbf{X}_{\text{mis}}, \theta, \phi)$, and helps avoid poor local solutions that occur more often with purely random initialization.

E.3 ALTERNATING STOCHASTIC OPTIMIZATION

MissBGM is trained by blockwise stochastic optimization on mini-batches. The procedure alternates between sample-specific updates, which refine the latent variables and the missing entries for the current batch, and global updates, which refine the parameters of the data model and the missingness model. This alternating structure mirrors the factorization of the joint posterior and avoids the need to optimize all unknowns simultaneously. More importantly, the alternating stochastic optimization only requires a single sample or a mini-batch of samples in each updating step, which makes it computationally efficient and scalable for large datasets.

We begin by running the EGM warm-start stage described above, which provides the initial state for the alternating updates. For a given batch during training, we first perform several local refinement steps on the latent variables and missing entries while holding the global network parameters fixed. The latent update maximizes the conditional log-posterior likelihood of the latent variables, while the missing-data update maximizes the conditional log-posterior likelihood of the missing entries under the joint data and mask model. After each missing-data update, the observed coordinates are projected back to their original values so that only the unobserved entries are allowed to move.

After the local refinement stage, we update the global parameters of g_θ and f_ϕ using the current completed batch. All gradient-based updates of the latent variables, the missing entries, and the global model parameters are performed with the Adam optimizer (Kingma & Ba, 2014). All batch losses are normalized by the feature dimension for numerical stability, and gradient clipping is used in the sample-specific steps to stabilize optimization. When Bayesian neural networks are used, the objective additionally includes the KL regularization terms associated with the variational weight posterior. Otherwise the updates reduce to stochastic gradient descent. In all cases, training proceeds by repeating this local-global alternation until the number of epochs is reached. The detailed algorithm is summarized in Algorithm 1.

Algorithm 1 Alternating stochastic optimization for MissBGM.

Require: Incomplete data matrix $\mathbf{X}_{\text{obs}}$, observation mask $\mathbf{R}$, number of epochs T, mini-batch size B, number of inner refinement steps K, learning rates η_1 for latent-variable and missing-value updates and η_2 for global parameter updates

Run EGM warm-start to initialize the model parameters and $(\mathbf{Z}^{(0)}, \mathbf{X}^{(0)})$
Set the initial state $(\mathbf{Z}, \mathbf{X}) \leftarrow (\mathbf{Z}^{(0)}, \mathbf{X}^{(0)})$
for $t = 1, 2, \ldots, T$ **do**
  Randomly shuffle the data and split them into mini-batches $\mathcal{B}$ of size at most B
  for each mini-batch $\mathcal{B}$ **do**
    for $k = 1, 2, \ldots, K$ **do**
      Update the batch latent variables by

$$\mathbf{Z}_{\mathcal{B}} \leftarrow \mathbf{Z}_{\mathcal{B}} - \eta_1 \nabla_{\mathbf{Z}_{\mathcal{B}}} \mathcal{L}_z(\mathcal{B}),$$

      where $\mathcal{L}_z(\mathcal{B})$ is the mini-batch latent objective induced by Eq. (6)
      Update the batch missing entries by

$$\mathbf{X}_{\text{mis},\mathcal{B}} \leftarrow \mathbf{X}_{\text{mis},\mathcal{B}} - \eta_1 \nabla_{\mathbf{X}_{\text{mis},\mathcal{B}}} \mathcal{L}_{x_{\text{mis}}}(\mathcal{B}),$$

      where $\mathcal{L}_{x_{\text{mis}}}(\mathcal{B})$ is the mini-batch missing-data objective induced by Eq. (8)
    end for
    Update the data-model parameters by stochastic gradient ascent,

$$\theta \leftarrow \theta + \eta_2 \nabla_\theta \mathcal{L}_\theta(\mathcal{B}),$$

    Update the missingness-model parameters by

$$\phi \leftarrow \phi + \eta_2 \nabla_\phi \mathcal{L}_\phi(\mathcal{B}),$$

    where $\mathcal{L}_\theta$ and $\mathcal{L}_\phi$ are the corresponding mini-batch KL-weighted variational objectives in Eq. (10)
  end for
end for
return fitted parameters (θ, ϕ) and final state $(\mathbf{Z}, \mathbf{X})$

E.4 Posterior inference and uncertainty quantification

Inference on the training data. In our imputation setting, posterior inference is most often performed on the same incomplete dataset used for training. In this case, no additional optimization is required. The alternating stochastic optimization already returns a completed data matrix at convergence, and this terminal state serves as the default point estimate. If posterior uncertainty is further required, we directly launch MCMC from the fitted terminal state while keeping the global parameters fixed.

Inference on a new dataset. When the test data differ from the training data, we keep the fitted global parameters (θ, ϕ) unchanged and update only the sample-specific latent variables and missing entries for the new dataset. Starting from a simple initial completion, we run the same local alternating refinement steps as in training, namely latent updates and missing-data updates, but without modifying the global model parameters. This produces a dataset-specific MAP state, which serves both as the point estimate for the new incomplete data and as the starting point for posterior sampling when uncertainty quantification is requested.

Posterior sampling. Given either the fitted training state or the dataset-specific MAP state, we approximate the β-tempered conditional target in Eq. (11) by HMC within Gibbs. Each sweep alternates between updating the latent variables conditional on the completed data and updating the missing entries conditional on the latent state and observed coordinates.

Posterior summaries. After discarding burn-in iterations, we summarize retained completed-data draws through conditional means and empirical prediction intervals. In the main experiments, we use 1000 burn-in iterations and retain 1000 draws, with initial step size 0.1 and 5 leapfrog steps.

Table 6: MissBGM main hyperparameters.

Hyperparameter	Explanation	Default
`z_dim`	Dimension of the latent variable $\mathbf{z}$.	5
`epochs`	Number of outer training epochs.	200
`batch_size`	Minibatch size used for stochastic training updates.	16
`beta`	Weight on the missingness-model term in Eq. (8).	0.01
`g_units`	Hidden-layer widths of g_θ.	$[120, 120, 120, 120, 120]$
`missingness_units`	Hidden-layer widths of f_ϕ.	$[64, 64]$
`lr_theta`	Learning rate for updating θ.	0.005
`lr_phi`	Learning rate for updating ϕ.	0.005
`lr_z`	Learning rate for updating $\mathbf{Z}$.	0.002
`lr_x`	Learning rate for updating $\mathbf{X}_{\text{mis}}$.	0.002
`n_inner_steps`	Number of inner updates of $\mathbf{Z}$ and $\mathbf{X}_{\text{mis}}$ per minibatch.	3
`use_bnn`	Whether to use the Bayesian neural networks	`false`
`kl_weight`	Weight τ on the KL regularization	5×10^{-5}
`test_epochs`	Number of test epochs before MCMC refinement.	30
`egm_init.enabled`	Whether to apply EGM initialization.	`true`
`egm_init.n_iter`	Number of mini-batches in EGM initialization.	1500
`egm_init.e_units`	Hidden-layer widths of encoder in EGM.	$[120, 120, 120, 120, 120]$
`egm_init.dz_units`	Hidden-layer widths of latent discriminator in EGM.	$[64, 32, 8]$
`egm_init.dx_units`	Hidden-layer widths of data discriminator in EGM.	$[64, 32, 8]$
`posterior.alpha`	Significance level for posterior prediction intervals.	0.05
`posterior.n_mcmc`	Number of retained posterior draws after burn-in.	1000
`posterior.burn_in`	Number of warm-up iterations in MCMC.	1000
`posterior.step_size`	Initial step size in MCMC.	0.1

E.5 Model hyperparameters

Table 6 summarizes the main MissBGM hyperparameters used in the main experiments. Dataset-specific YAML files may override few entries for particular small or large datesets. For example, we use `z_dim=10`, `egm_init.n_iter=5000` for the largest Gisette dataset, and use `g_units=[120,120,120]` for the smallest Wine dataset to keep the model simple and stable. The `epochs` is set to 50 for all simulation benchmark experiments. Overall, the model is robust to the choice of hyperparameters for datasets with varying sample sizes and feature dimensions.

The EGM encoder architecture is chosen as a rough mirror of the generator g_θ (e.g., reversing the layer widths in `g_units`), which stabilizes initialization and reduces the need for manual tuning. When the underlying data distribution is complex (large p, strong nonlinearity, or multi-modality), using a larger latent dimension `z_dim` makes more sense to improve fit and downstream imputation quality, at the cost of additional compute. In our experiments, we find that setting $z_{\text{dim}} = 5$ by default is across-the-board effective for most datasets.

E.6 Experimental environment

MissBGM was implemented in `Python` using `TensorFlow` for neural network modeling and alternating stochastic optimization, and `TensorFlow Probability` (TFP) (Dillon et al., 2017) for Bayesian neural networks and HMC-within-Gibbs posterior sampling. Note that the posterior inference for one sample is independent of other samples, so the Markov chains of all test samples are run in parallel using TFP for efficiency. All benchmark experiments were conducted on a Linux-

based high-performance computing cluster managed by `Slurm`. The GPU compute nodes used in our experiments were equipped with NVIDIA RTX 5000 Ada GPUs and dual-socket CPUs (48 CPU cores per node in total). For each experiment, we requested a maximum wall-clock time of 3 days and 300 GB of memory. We only used one GPU per experiment for deep learning models.

F SOFTWARE USAGE FOR MISSBGM

This section provides an example showing how to use MissBGM on a synthetic dataset. The code first loads a YAML configuration file into `params`, which contains model and inference hyperparameters such as network widths, learning rates, and MCMC settings, etc. It then prepares an incomplete dataset `data`: the main input to MissBGM is the partially observed matrix `x_obs`$\in \mathbb{R}^{n\times p}$, with missing entries encoded as `np.nan`; the optional binary mask `mask`$\in \{0,1\}^{n\times p}$ records which entries are observed (will be inferred from `x_obs` if not provided), and the optional complete data matrix `x_full` for evaluation only. The call `MissBGM(params, random_seed=42)` initializes the model, and `model.fit(...)` trains it using `x_obs` and `mask`. After fitting, `model.x_map_imputed_` stores the MAP completed data matrix, while `model.predict(...)` returns posterior summaries, including posterior-mean imputations and prediction intervals for all the missing entries. Note that the argument `x_true` is only used for internal evaluation and can be set to `None`. The argument `mask` can be automatically inferred from `x_obs` by setting it to `None`.

The example below illustrates running MissBGM on the synthetic benchmark, including model instantialization, model training and posterior-sampling inference.

Example: run MissBGM on a synthetic dataset

```
import yaml
from missbgm.models import MissBGM
from missbgm.datasets import simulate_mnar_oracle_data

params = yaml.safe_load(open("configs/Sim_MNAR_oracle.yaml", "r"))
data = simulate_mnar_oracle_data(
   n_samples=500,
   x_dim=50
)

# Instantiate the MissBGM model with the configuration
model = MissBGM(params, random_seed=42)

# Train the MissBGM model
model.fit(data=data["x_obs"], mask=data["mask"], x_true=data["x_full"]
    ])

# Get the MAP imputation
map_imputed = model.x_map_imputed_

# Make posterior predictions with uncertainty quantification
mcmc_imputed, intervals = model.predict(
   data=data["x_obs"],
   mask=data["mask"],
   x_true=data["x_full"],
   alpha=0.05
)
```

In this example, `simulate_mnar_oracle_data` returns the observed matrix `x_obs` (the missing entries are all `np.nan`), the binary missingness mask `mask`, and the complete synthetic ground truth `x_full`. The call to `model.fit(...)` runs EGM initialization, followed by the alternating stochastic optimization procedure. The subsequent call to `model.predict(...)` returns the posterior mean imputation and the corresponding posterior prediction intervals for the missing

entries. Note that the argument `x_true` is only used for internal evaluation and can be set to `None`. The argument `mask` can be automatically inferred from `x_obs` by setting it to `None`.

When posterior inference is performed on the same incomplete dataset used for training, as in the example above, MissBGM directly reuses the fitted training state and does not re-optimize the global parameters. If uncertainty quantification is not needed, one may stop after training and use the fitted state as the default point estimate. If uncertainty quantification is desired, posterior sampling can then be launched through `predict(...)`.

G Effect of missing rate and sample size

To complement Figure 2(b), which reports the missing-rate experiment at $n = 5000$, we provide the corresponding curves for three additional sample sizes, namely $n \in \{500, 1000, 10000\}$. In all cases, we fix $p = 50$ and vary the missing rate only. These additional plots show that the qualitative conclusions are stable across sample sizes. MissBGM remains the strongest method overall.

A second consistent pattern is that the RMSE curves are not strictly monotone in the missing rate. Instead, most methods exhibit a broad U-shaped trend, with the lowest RMSE often attained at moderate missingness levels and larger errors reappearing at high missingness rates. This behavior is consistent with the synthetic data construction and the discussion in Appendix C.1. Overall, the figures below reinforce the conclusion from Figure 2(b): MissBGM is robust to substantial changes in missingness severity.

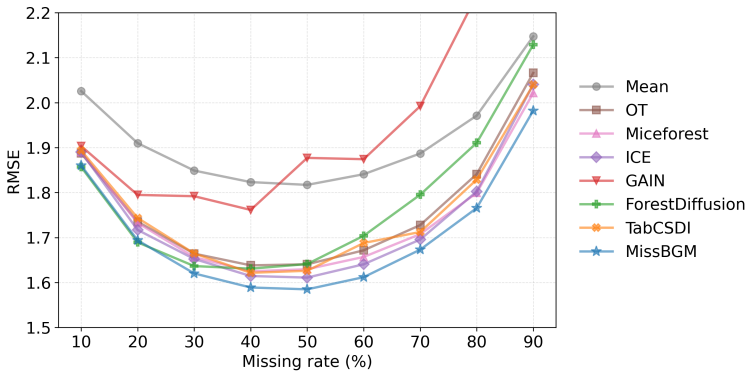

Figure 3: RMSE with different missing rate on the synthetic benchmark with $n = 500$ and $p = 50$.

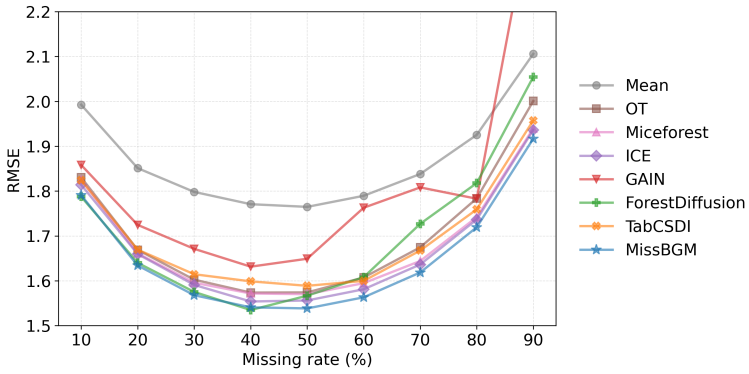

Figure 4: RMSE with different missing rate on the synthetic benchmark with $n = 1000$ and $p = 50$.

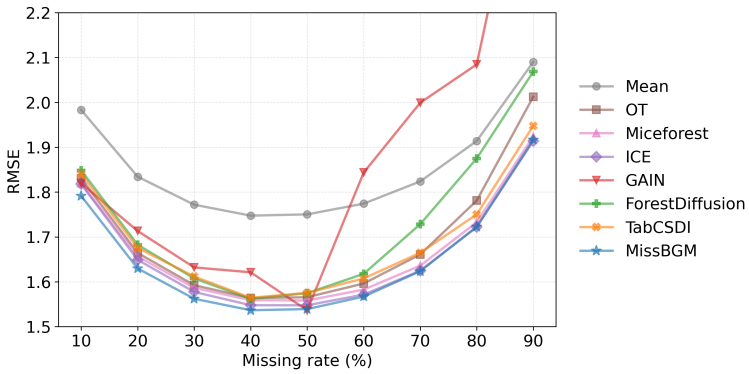

Figure 5: RMSE with different missing rate on the synthetic benchmark with $n = 10000$ and $p = 50$.

H RUNNING TIME

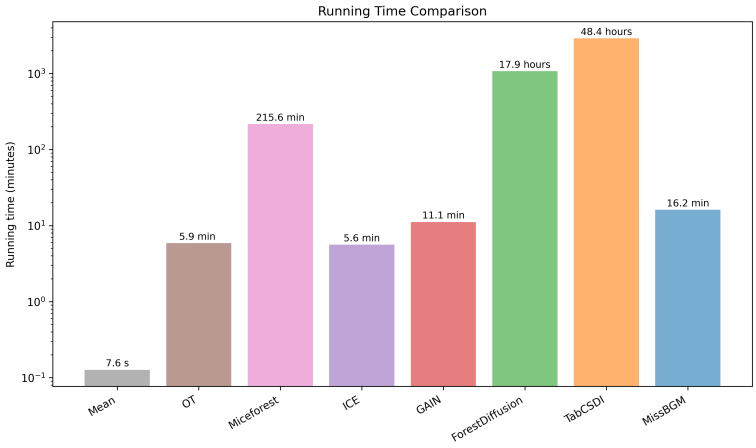

Figure 6: Running time comparison across imputation methods on a representative synthetic benchmark with moderate complexity settings ($n = 5000$, $p = 200$).

Figure 6 reports the running time of all compared methods on a representative synthetic benchmark with moderate complexity settings ($n = 5000$, $p = 200$, and missing rate $r = 0.2$). The y-axis is shown on a logarithmic scale as running time spans different orders of magnitude, from seconds to days. Mean imputation finishes in seconds while OT and ICE both complete in less than 10 minutes. Miceforest requires 215.6 minutes, which is already more than one order of magnitude slower than MissBGM due to the repeated fitting of conditional models for each feature across multiple iterations. The diffusion-based methods are substantially more expensive. ForestDiffusion requires 17.9 hours and TabCSDI requires 48.4 hours.

Therefore, although MissBGM jointly models both the data-generating process and the missingness mechanism and additionally supports uncertainty quantification, its computational cost remains moderate relative to other deep learning-based imputers. In particular, MissBGM is roughly comparable to GAIN, but substantially faster than miceforest and diffusion-based methods.

I EFFECT OF THE BAYESIAN NEURAL NETWORKS

To complement the main experiments, we replace the fully-connected neural networks of g_θ and f_ϕ with Bayesian neural networks to acount for model parameter uncertainty while keeping all the architecture and hyperparameters the same. We use the same synthetic protocol as in Table 1,

with missing rate $r = 0.3$ and sample sizes $n \in \{500, 1000, 5000, 10000\}$. The BNN version of MissBGM yields comparable or slightly worse RMSE than the default version.

Table 7: Effect of the Bayesian neural networks on the synthetic benchmark at missing rate $r = 0.3$. We report mean (s.d.) RMSE over 10 runs.

Method	**Size=500**	**Size=1000**	**Size=5000**	**Size=10000**
MissBGM (orig)	**1.620 (0.011)**	**1.568 (0.009)**	1.599 (0.010)	**1.562 (0.006)**
MissBGM (BNN)	1.632 (0.017)	1.570 (0.012)	**1.592 (0.012)**	1.564 (0.008)