CG-DIFF: CAUSAL-GRAPH-GUIDED DIFFUSION MODELS FOR TABULAR IMPUTATION UNDER SELF-MASKING MNAR

Anonymous authors
Paper under double-blind review

ABSTRACT

Missing-value imputation under Self-Masking Missing Not At Random (MNAR) is challenging because observed and missing values may follow different distributions, making it difficult to recover missing values from observed data alone. We propose a Causal-Graph-guided DIFFusion model (**CG-DIFF**) for tabular missing-value imputation under Self-Masking MNAR. **CG-DIFF** introduces an observation mask into the forward diffusion process to enable training on observed values only, and develops a Causal-Graph-guided Noise Estimator (CG-NE) to estimate diffusion noise from incomplete data. CG-NE combines Kolmogorov–Arnold network projections with graph-masked attention, using either parents plus self or the Markov blanket plus self as the graph support. At each reverse step, **CG-DIFF** replaces the values at observed positions with their forward-noised counterparts at the corresponding timestep, thereby preserving observed information throughout sampling. We establish sufficient conditions under which the population predictor based on the Markov blanket has lower missing-case squared-error risk than the predictor based only on the parent set, providing theoretical motivation for the Markov-blanket-plus-self design. We evaluate **CG-DIFF** against nine competing imputation methods on simulated datasets and a real-world dataset. Experimental results show that **CG-DIFF** with the Markov-blanket-plus-self graph support achieves better imputation performance than the competing methods in most settings.

1 INTRODUCTION

For tabular missing-value imputation, the missingness mechanism is crucial because it governs the relationship between observed and missing values and directly affects imputation reliability (Little & Rubin, 2019; Seaman et al., 2013). We study continuous tabular data under Self-Masking Missing Not At Random (MNAR), where the probability that a variable is observed depends on its underlying value (Rubin, 1976; Sportisse et al., 2020). As a result, observed and missing values generally follow different distributions, and missing values cannot in general be recovered from the observed data alone without additional assumptions or structural information (Mohan & Pearl, 2021). Motivated by this challenge, we use a supplied causal graph to encode structural relationships among variables and guide missing-value imputation through diffusion models.

Many existing imputation methods perform well on incomplete data but are not specifically designed for Self-Masking MNAR. MICE fits a conditional model using cases in which the target variable is observed and then uses this model to impute its missing values (van Buuren & Groothuis-Oudshoorn, 2011). Under Self-Masking MNAR, the distribution of the observed values may differ from that of the missing values. As a result, a model learned from the observed cases may not predict the missing cases accurately. GAIN uses adversarial learning to distinguish observed from imputed values (Yoon et al., 2018), while its main theoretical analysis assumes MCAR. MIWAE trains a latent-variable model with an importance-weighted objective (Mattei & Frellsen, 2019), but is developed under MAR. OT-Imputer uses optimal transport to match distributions (Muzellec et al., 2020); under Self-Masking MNAR, however, observed and missing values may follow different distributions, limiting the effectiveness of such distribution matching. HyperImpute automatically selects a prediction model and its hyperparameters for each incomplete variable (Jarrett et al., 2022), but is developed for

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.

MCAR or MAR settings. Under Self-Masking MNAR, improving predictive accuracy on observed data alone does not account for the difference between observed and missing values. MIRACLE jointly models the missingness mechanism and uses causal regularization to iteratively refine an initial imputation (Kyono et al., 2021). Although it is designed to handle MNAR, its final imputation depends on both the learned missingness mechanism and the initial imputed values, which may be difficult to estimate accurately when observed data provide limited information about missing values. GRIMP constructs a heterogeneous graph from an incomplete tabular dataset and uses graph representation learning with attribute-specific attention to predict missing values (Cappuzzo et al., 2024), but its published experiments focus primarily on MCAR rather than Self-Masking MNAR. Overall, these methods address different aspects of missing-value imputation, but none is specifically designed to exploit known structural relationships among variables to guide imputation under Self-Masking MNAR.

Recent diffusion-based methods have also been applied to missing-value imputation, but challenges remain under Self-Masking MNAR. CSDI uses conditional score-based diffusion to reconstruct missing values from observed data (Tashiro et al., 2021), but its artificial training masks may not reflect value-dependent missingness. TabDDPM applies diffusion models to tabular data (Kotelnikov et al., 2023), but is designed primarily for data generation rather than MNAR imputation. NewImp learns the joint data distribution through a Wasserstein gradient-flow framework (Chen et al., 2024), but does not explicitly model Self-Masking MNAR or exploit causal-graph information. Overall, existing diffusion-based methods do not jointly address Self-Masking MNAR and exploit known structural relationships among variables to guide imputation.

Diffusion models provide a natural framework for missing-value imputation by generating data through a reverse denoising process, while causal graphs provide structural information among variables. Building on DDIM (Song et al., 2021) and graph attention (Veličković et al., 2018), we propose the **Causal-Graph-Guided Diffusion model** (**CG-DIFF**) for tabular imputation. Since the rows in a tabular dataset are assumed to be independent and identically distributed, **CG-DIFF** processes the data row by row and applies the DDIM forward process to each row of complete data. **CG-DIFF** introduces a mask into the forward process so that the training loss is computed only on observed values. To exploit structural information, we develop a Causal-Graph-guided Noise Estimator (CG-NE), which combines Kolmogorov–Arnold network (KAN) projections (Liu et al., 2025) with graph-masked attention to estimate diffusion noise from incomplete data. We consider two graph supports: parents plus self and the Markov blanket plus self. During each reverse step, values at observed positions are replaced with their forward-noised counterparts at the corresponding timestep, preserving observed information throughout sampling. Starting from Gaussian noise, **CG-DIFF** generates each imputed row through the DDIM reverse process several times, and calculates their elementwise median. Finally, the resulting rows are stacked to form the final imputed table.

Our contributions are:

1. **Diffusion framework for imputation.** We develop a diffusion-based framework for missing-value imputation that incorporates the observation mask into the forward process, allowing the loss to be computed only on observed values while the reverse process generates complete data.
2. **Causal-Graph-guided Noise Estimator (CG-NE).** We introduce CG-NE, which combines Kolmogorov–Arnold network (KAN) projections (Liu et al., 2025) with graph-masked attention (Veličković et al., 2018) to estimate diffusion noise from incomplete data. The supplied causal graph guides information exchange among variables, thereby incorporating structural information into noise estimation.
3. **Theoretical analysis and empirical evaluation.** We establish sufficient conditions under which the population predictor based on the Markov blanket has lower missing-case squared-error risk than the parent-based predictor. We evaluate **CG-DIFF** against nine imputation methods on simulated datasets and a real dataset. **CG-DIFF** with the Markov-blanket-plus-self graph support achieves better imputation performance than the competing methods in most settings.

2 NOTATION

Let $\mathbf{x}_i^\star \in \mathbb{R}^d$ be the complete value vector for row i, and let $\mathbf{m}_i \in \{0,1\}^d$ be its observation mask, with $m_{i,j} = 1$ when $x_{i,j}^\star$ is observed and $m_{i,j} = 0$ otherwise. We assume that $\mathbf{x}_i^\star \overset{iid}{=} \mathbf{x}^\star$. Its observed and missing parts are represented as d-dimensional vectors:

$$\mathbf{x}_{i,\mathrm{obs}} = \mathbf{m}_i \odot \mathbf{x}_i^\star, \qquad \mathbf{x}_{i,\mathrm{mis}} = (\mathbf{1} - \mathbf{m}_i) \odot \mathbf{x}_i^\star,$$

where $\odot$ denotes elementwise multiplication and $\mathbf{1} \in \mathbb{R}^d$ is the all-ones vector. Thus $\mathbf{x}_{i,\mathrm{obs}}$ contains zero placeholders at missing positions, while $\mathbf{x}_{i,\mathrm{mis}}$ contains the unknown values at those positions and zeros elsewhere.

Superscripts train and imp distinguish the N_{train} rows used for training from the N_{imp} rows to be imputed. Their observed inputs are $(\mathbf{x}_{i,\mathrm{obs}}^{\mathrm{train}}, \mathbf{m}_i^{\mathrm{train}})$ and $(\mathbf{x}_{i,\mathrm{obs}}^{\mathrm{imp}}, \mathbf{m}_i^{\mathrm{imp}})$, respectively; the corresponding complete data vectors and missing vectors carry the same superscript. Both masks are known. We suppress the role superscript and row index in formulas that apply to any single vector.

Uppercase value matrices stack the transposed row vectors. In particular, $\mathbf{X}_{\mathrm{obs}}^{\mathrm{train}} \in \mathbb{R}^{N_{\mathrm{train}} \times d}$ and $\mathbf{M}^{\mathrm{train}} \in \{0,1\}^{N_{\mathrm{train}} \times d}$ stack the training observations and masks, while $\mathbf{X}_{\mathrm{obs}}^{\mathrm{imp}}$, $\mathbf{X}_{\mathrm{mis}}^{\mathrm{imp}}$, and $\mathbf{M}^{\mathrm{imp}}$ have N_{imp} rows. For example,

$$\mathbf{X}_{\mathrm{mis}}^{\mathrm{imp}} = \left[\mathbf{x}_{1,\mathrm{mis}}^{\mathrm{imp}}, \ldots, \mathbf{x}_{N_{\mathrm{imp}},\mathrm{mis}}^{\mathrm{imp}}\right]^\top \in \mathbb{R}^{N_{\mathrm{imp}} \times d}.$$

When the role is immaterial, N, $\mathbf{X}^\star$, $\mathbf{M}$, and $\mathbf{X}_{\mathrm{obs}} = \mathbf{M} \odot \mathbf{X}^\star$ denote the row count, complete data matrix, observation mask, and zero-filled observed matrix of the collection under discussion. The symbol $\mathbf{I}_d$ denotes the $d \times d$ identity.

3 PROBLEM FORMULATION AND CAUSAL GRAPH INTRODUCTION

3.1 PROBLEM FORMULATION

Given training observations $(\mathbf{x}_{i,\mathrm{obs}}^{\mathrm{train}}, \mathbf{m}_i^{\mathrm{train}})_{i=1}^{N_{\mathrm{train}}}$ and a supplied directed graph $\mathcal{G}$, CG-DIFF learns a graph-conditioned predictor. At imputation time, it receives $(\mathbf{x}_{i,\mathrm{obs}}^{\mathrm{imp}}, \mathbf{m}_i^{\mathrm{imp}})_{i=1}^{N_{\mathrm{imp}}}$ and estimates the missing vectors $(\mathbf{x}_{i,\mathrm{mis}}^{\mathrm{imp}})_{i=1}^{N_{\mathrm{imp}}}$ while preserving all observed values. It generates candidate complete data vectors $(\widetilde{\mathbf{x}}_i^{\mathrm{imp}})_{i=1}^{N_{\mathrm{imp}}}$ and returns

$$\widehat{\mathbf{x}}_i^{\mathrm{imp}} = \mathbf{x}_{i,\mathrm{obs}}^{\mathrm{imp}} + (\mathbf{1} - \mathbf{m}_i^{\mathrm{imp}}) \odot \widetilde{\mathbf{x}}_i^{\mathrm{imp}}, \qquad i = 1, \ldots, N_{\mathrm{imp}}.$$

We finally stack the returned vectors $(\widehat{\mathbf{x}}_i^{\mathrm{imp}})_{i=1}^{N_{\mathrm{imp}}}$ to get the completed value matrix:

$$\widehat{\mathbf{X}}^{\mathrm{imp}} = \left[\widehat{\mathbf{x}}_1^{\mathrm{imp}}, \ldots, \widehat{\mathbf{x}}_{N_{\mathrm{imp}}}^{\mathrm{imp}}\right]^\top \in \mathbb{R}^{N_{\mathrm{imp}} \times d}.$$

3.2 CAUSAL GRAPH PRIOR

Unrestricted MNAR recovery is non-identifiable without additional assumptions (Mohan & Pearl, 2021). We therefore study the Self-Masking mechanism in Definition 3.1 and use a supplied directed graph to specify which columns exchange information during prediction.

Let $\mathcal{G} = (V, E)$ be a directed acyclic graph (DAG), with node set $V = \{X_1^\star, \ldots, X_d^\star\}$, where $X_j^\star$ denotes the graph node associated with the jth data variable, and E is the set of directed edges. The graph notation $X_j^\star$ should be distinguished from $x_{i,j}^\star$, which denotes the corresponding random variable in row i in probability statements.

For a target variable $X_j^\star$ in $\mathcal{G}$, its parent and child index sets are $\mathrm{Pa}(j) = \{k : (X_k^\star \to X_j^\star) \in E\}$ and $\mathrm{Ch}(j) = \{k : (X_j^\star \to X_k^\star) \in E\}$, respectively. Its spouses and Markov blanket (MB) are defined in the usual graphical-model sense (Pearl, 2009):

$$\mathrm{Sp}(j) = \{k \neq j : \exists\, X_c^\star,\ X_j^\star \to X_c^\star,\ X_k^\star \to X_c^\star\}, \qquad \mathrm{MB}(j) = \mathrm{Pa}(j) \cup \mathrm{Ch}(j) \cup \mathrm{Sp}(j).$$

We next introduce the Self-Masking MNAR mechanism considered in this work.

Definition 3.1 (Self-Masking MNAR (Sportisse et al., 2020)). *Self-Masking MNAR assumes*

$$\Pr(m_{i,j} = 0 \mid \mathbf{x}_i^\star) = \Pr(m_{i,j} = 0 \mid x_{i,j}^\star).$$

Under Self-Masking MNAR, for any row i and any $j \neq k$, $m_{i,k} \perp\!\!\!\perp x_{i,j}^\star \mid x_{i,k}^\star$. Thus, once $x_{i,k}^\star$ is given, $m_{i,k}$ provides no additional information about another complete-data value $x_{i,j}^\star$. We therefore retain the observation mask as an explicit input to CG-DIFF.

We next consider which variables in the supplied graph should be used to predict each target variable. For each target $X_j^\star$, let $\mathcal{S}(j) \subseteq \{1, \ldots, d\} \setminus \{j\}$ denote the indices of the variables allowed to provide information for predicting $X_j^\star$. The corresponding graph support is defined by $A(\mathcal{S})$, with $(A(\mathcal{S}))_{jk} = \mathbb{1}[k \in \mathcal{S}(j) \cup \{j\}]$.

Two common choices are $\mathcal{S} \in \{\mathrm{Pa}, \mathrm{MB}\}$ (Tsamardinos & Aliferis, 2003; Brown & Tsamardinos, 2008; Pearl, 2009). We choose the Markov blanket rather than the parent set to use the additional predictive information that children and spouses may provide. Under Self-Masking MNAR, the observation probability depends on the target value, so its distribution among observed cases ($m_{i,j} = 1$) may differ from that among missing cases ($m_{i,j} = 0$). To justify our choice under this distribution shift, we compare the missing-case risks of predictors that minimize population squared error on observed cases.

For $\mathcal{S} \in \{\mathrm{Pa}, \mathrm{MB}\}$, define $\mathcal{I}_\mathcal{S}(i,j) = \sigma\big(\big\{\big((\mathbf{x}_{i,\mathrm{obs}})_k, m_{i,k}\big) : k \in \mathcal{S}(j)\big\}\big)$, where $\sigma(\cdot)$ denotes the generated σ-algebra. The population predictor that minimizes squared error on observed cases is

$$\mu_\mathcal{S}(i,j) = \mathbb{E}[x_{i,j}^\star \mid \mathcal{I}_\mathcal{S}(i,j), m_{i,j} = 1].$$

We evaluate this same predictor in both populations:

$$R_\mathcal{S}^{(r)}(j) = \mathbb{E}\big[\{x_{i,j}^\star - \mu_\mathcal{S}(i,j)\}^2 \mid m_{i,j} = r\big], \qquad r \in \{0,1\}.$$

Here $R_\mathcal{S}^{(1)}(j)$ and $R_\mathcal{S}^{(0)}(j)$ are the observed-case and missing-case risks, respectively. Define

$$\pi_{i,j} = \Pr\big(m_{i,j} = 1 \mid x_{i,j}^\star, \mathcal{I}_\mathrm{MB}(i,j)\big), \qquad w_{i,j} = \frac{1 - \pi_{i,j}}{\pi_{i,j}}.$$

Thus $w_{i,j}$ gives the missingness odds conditional on the target value and the available Markov-blanket information. Let

$$H_{i,j} = \{x_{i,j}^\star - \mu_\mathrm{Pa}(i,j)\}^2 - \{x_{i,j}^\star - \mu_\mathrm{MB}(i,j)\}^2.$$

Theorem 3.1. *Fix a target j. Under Self-Masking MNAR, assume that $\mathbb{E}[(x_{i,j}^\star)^2] < \infty$, $0 < \pi_{i,j} < 1$ almost surely, and $R_\mathcal{S}^{(0)}(j) < \infty$ for $\mathcal{S} \in \{\mathrm{Pa}, \mathrm{MB}\}$. Then*

$$\Delta_j = R_\mathrm{Pa}^{(1)}(j) - R_\mathrm{MB}^{(1)}(j) = \mathbb{E}\big[\{\mu_\mathrm{MB}(i,j) - \mu_\mathrm{Pa}(i,j)\}^2 \mid m_{i,j} = 1\big] \geq 0. \tag{1}$$

Moreover,

$$R_\mathrm{Pa}^{(0)}(j) - R_\mathrm{MB}^{(0)}(j) = \Delta_j + \frac{\mathrm{Cov}(w_{i,j}, H_{i,j} \mid m_{i,j} = 1)}{\mathbb{E}[w_{i,j} \mid m_{i,j} = 1]}, \tag{2}$$

where

$$\mathbb{E}[w_{i,j} \mid m_{i,j} = 1] = \frac{\Pr(m_{i,j} = 0)}{\Pr(m_{i,j} = 1)} \in (0, \infty). \tag{3}$$

If

$$\mathrm{Cov}(w_{i,j}, H_{i,j} \mid m_{i,j} = 1) \geq 0, \tag{4}$$

then $R_\mathrm{Pa}^{(0)}(j) - R_\mathrm{MB}^{(0)}(j) \geq \Delta_j \geq 0$.

The proof is provided in Supplementary Materials A. Theorem 3.1 first shows that the Markov-blanket population predictor has no greater squared-error risk on observed cases than the parent-only predictor, with the improvement measured by Δ_j. Here $H_{i,j} > 0$ means that the Markov-blanket predictor has smaller squared error on a case. The covariance term in (2) describes whether this advantage tends to be larger or smaller for cases with larger conditional missingness odds. If the covariance is nonnegative, the missing-case improvement is at least Δ_j.

A sufficient condition for (4) is that, among observed cases, the average error reduction from using the Markov blanket rather than the parent set is nondecreasing in the conditional missingness odds; that is, for $u_1 \le u_2$, $\mathbb{E}[H_{i,j} \mid w_{i,j} = u_1, m_{i,j} = 1] \le \mathbb{E}[H_{i,j} \mid w_{i,j} = u_2, m_{i,j} = 1]$. Hence, if the additional information from children and spouses is more useful for cases that are more likely to be missing, the Markov-blanket predictor tends to achieve larger error reductions on such cases. In this case, any positive observed-case improvement carries over to missing cases.

For real datasets, we construct the graph prior from domain knowledge documented in the literature or curated databases. For example, in our controlled imputation analysis using the Alzheimer's Disease Neuroimaging Initiative (ADNI) dataset, the working DAG is oriented from genetic ordering, biomarker-stage models, and longitudinal evidence (Jack et al., 2010; Peskind et al., 2006; Risacher et al., 2017).

4 CG-DIFF: Training and Imputation

DDIM is a useful diffusion model, which is detailed in section 4.1 (Song et al., 2021). But it is not directly designed for missing-value imputation. If diffusion is applied to the complete data, the loss function cannot be computed at missing positions because their values in the complete data are unknown. If diffusion is applied only to observed values, the reverse process cannot directly generate a sample of complete data.

To address this issue, we propose Causal-Graph-Guided Diffusion model (CG-DIFF), a new diffusion framework specifically for missing-value imputation. It contains three main components. First, we introduce a mask into the forward process. This allows diffusion to be defined over the complete data while computing the loss function only at observed positions, and the reverse process can still generate complete data, which is detailed in Section 4.2. Second, we design a Causal-Graph-guided Noise Estimator (CG-NE) that uses causal-graph to improve noise estimation, which is detailed in Section 4.3 and algorithm 1. Third, we make full use of the observed values during sampling. At each reverse step, the generated values at observed positions are replaced with the corresponding forward-noised observed values. During imputation, we run the diffusion reverse process multiple times to generate candidate complete-value vectors and use their elementwise median as the final imputed vector. This improves the quality of the generated data, which is detailed in Section 4.4 and algorithm 2.

4.1 DDIM Framework

For a sample of complete data $\mathbf{x}_0 \sim q_{\text{data}}$, DDIM uses the following forward process (Song et al., 2021):

$$\mathbf{x}_t = \sqrt{\alpha_t}\,\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t^{\text{fwd}}, \quad \boldsymbol{\epsilon}_t^{\text{fwd}} \overset{\text{iid}}{\sim} \mathcal{N}(0, \mathbf{I}_d), \quad t = 1, \dots, T,$$

where T is the number of diffusion steps, $\{\beta_t\}_{t=1}^T$ is the variance schedule. The resulting marginal expresses $\mathbf{x}_t$ as a linear combination of $\mathbf{x}_0$ and an aggregate noise variable $\boldsymbol{\epsilon}$:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}_d), \quad t = 1, \dots, T, \tag{5}$$

where $\bar{\alpha}_t = \prod_{u=1}^t \alpha_u$, $\alpha_t = 1 - \beta_t$, and $\bar{\alpha}_0 = 1$. DDIM calculate noise estimator $\boldsymbol{\epsilon}_\theta^{(t)}$ parameterized by θ through minimizing (Song et al., 2021):

$$\mathcal{L}(\theta) = \mathbb{E}_{t, \mathbf{x}_0, \boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta^{(t)}(\mathbf{x}_t, t)\right\|_2^2\right],$$

where $\mathbf{x}_0 \sim q_{\text{data}}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}_d)$, and t is sampled uniformly from $\{1, \dots, T\}$. For two selected sampling times $t > s \ge 0$, DDIM uses the following reverse process:

$$\mathbf{x}_s = \underbrace{\sqrt{\bar{\alpha}_s}\left(\frac{\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}_\theta^{(t)}(\mathbf{x}_t, t)}{\sqrt{\bar{\alpha}_t}}\right)}_{\text{predicted } \mathbf{x}_0} + \underbrace{\sqrt{1 - \bar{\alpha}_s - \sigma_{t,s}^2}\,\boldsymbol{\epsilon}_\theta^{(t)}(\mathbf{x}_t, t)}_{\text{direction pointing to } \mathbf{x}_t} + \underbrace{\sigma_{t,s}\boldsymbol{\epsilon}_t}_{\text{random noise}},$$

where $\boldsymbol{\epsilon}_\theta^{(t)}(\mathbf{x}_t, t)$ is the estimated noise and $\boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I}_d)$ is standard Gaussian noise independent of $\mathbf{x}_t$. The transition scale $\sigma_{t,s}$ is specified in Appendix C.1.

4.2 Masked Forward Process for Incomplete Values

For one incomplete value vector, the zero-filled observation $\mathbf{x}_{\text{obs}} = \mathbf{m} \odot \mathbf{x}^\star$, we use the following forward process:

$$\mathbf{x}_t = \mathbf{m} \odot \mathbf{x}_t^\star = \mathbf{m} \odot \left(\sqrt{\alpha_t}\, \mathbf{x}_{t-1}^\star + \sqrt{\beta_t} \boldsymbol{\epsilon}_t^{\text{fwd}} \right), \quad \boldsymbol{\epsilon}_t^{\text{fwd}} \overset{\text{iid}}{\sim} \mathcal{N}(0, \mathbf{I}_d), \quad t = 1, \ldots, T,$$

where $\mathbf{x}_0^\star \overset{d}{=} \mathbf{x}^\star$.

The resulting marginal expresses $\mathbf{x}_t$ as a linear combination of $\mathbf{x}_0^\star$ and an aggregate noise variable $\boldsymbol{\epsilon}$:

$$\begin{aligned} \mathbf{x}_t = \mathbf{m} \odot \mathbf{x}_t^\star &= \mathbf{m} \odot \left(\sqrt{\bar{\alpha}_t} \mathbf{x}_0^\star + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon} \right) \\ &= \mathbf{m} \odot \left(\sqrt{\bar{\alpha}_t} \mathbf{x}_{\text{obs}} + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon} \right), \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}_d), \quad t = 1, \ldots, T. \end{aligned} \tag{6}$$

The mask $\mathbf{m}$ is supplied as a separate conditioning channel and identifies the observed values used in the loss. The batched implementation is given in Appendix C.1.

4.3 Causal-Graph-guided Noise Estimator (CG-NE)

Figure 1 shows the Causal-Graph-guided Noise Estimator (CG-NE) architecture. CG-NE receives $(\theta_{A(\mathcal{S})}, t, \mathbf{x}_t; A(\mathcal{S}))$, and $\mathbf{x}_t$ is first mapped by the row-wise KAN projection to $\Phi_{\theta_{A(\mathcal{S})}}(\mathbf{x}_t) \in \mathbb{R}^{d \times d_m}$, where the jth row of $\Phi_{\theta_{A(\mathcal{S})}}(\mathbf{x}_t)$ is the d_m-dimensional feature representation associated with the jth entry of $\mathbf{x}_t$ (Liu et al., 2025). The resulting features are then combined with the learned variable parameters $C_{A(\mathcal{S})}$ and the time representation $\tau_{\theta_{A(\mathcal{S})}}(t)$ to form the initial hidden representation $H_{A(\mathcal{S})}^{(0)}$. Subsequently, they are processed by L residual graph-masked attention layers with graph support $A(\mathcal{S})$ (Veličković et al., 2018), producing hidden representations $H_{A(\mathcal{S})}^{(\ell)}$, $\ell = 1, \ldots, L$. The final hidden representation $H_{A(\mathcal{S})}^{(L)}$ is mapped by the stream-specific affine output with parameters $\mathbf{w}_{A(\mathcal{S})}$ and $b_{A(\mathcal{S})}$ to the intermediate output $\mathbf{y}_{A(\mathcal{S})}(\theta_{A(\mathcal{S})}, t, \mathbf{x}_t; A(\mathcal{S}))$, which is then converted to the noise estimate $\widehat{\boldsymbol{\epsilon}}_{\theta_{A(\mathcal{S})}}(\theta_{A(\mathcal{S})}, t, \mathbf{x}_t; A(\mathcal{S}))$. The detailed process is provided in Appendices C.2–C.3.

CG-NE is trained by minimizing the loss function parameterized by $\theta_{A(\mathcal{S})}$ on the observed values:

$$\mathcal{L}(\theta_{A(\mathcal{S})}) = \mathbb{E}_{\mathbf{x}_0^\star, t, \boldsymbol{\epsilon}} \left[\left\| \mathbf{m} \odot \left(\widehat{\boldsymbol{\epsilon}}_{\theta_{A(\mathcal{S})}}(\theta_{A(\mathcal{S})}, t, \mathbf{x}_t; A(\mathcal{S})) - \boldsymbol{\epsilon} \right) \right\|_2^2 \right], \tag{7}$$

where $\mathbf{x}_t$ is given by Equation (6). The training process of CG-NE is described in algorithm 1.

Through minimizing Equation (7), we get the optimal parameters $\theta_{A(\mathcal{S})}^\star$ of CG-NE. To reduce the effect of extreme noise values on the backward process, we clip the noise estimate $\widehat{\boldsymbol{\epsilon}}_{\theta_{A(\mathcal{S})}^\star}$ with an adaptive bound $[l(\mathbf{x_t}, t), u(\mathbf{x_t}, t)]$:

$$\widetilde{\boldsymbol{\epsilon}}_{\theta_{A(\mathcal{S})}^\star}(\mathbf{x}_t, t; A(\mathcal{S})) = \text{clip}\left(l(\mathbf{x}_t, t), \widehat{\boldsymbol{\epsilon}}_{\theta_{A(\mathcal{S})}^\star}(\mathbf{x}_t, t; A(\mathcal{S})), u(\mathbf{x}_t, t) \right), \tag{8}$$

where $\text{clip}(\mathbf{a}, \mathbf{z}, \mathbf{b}) = \min\{\mathbf{b}, \max\{\mathbf{a}, \mathbf{z}\}\}$ is applied elementwise. The construction of $l(\mathbf{x}_t, t)$ and $u(\mathbf{x}_t, t)$ is provided in Appendix C.1.

4.4 Adaptive Backward Process for Imputation

The corresponding DDIM sampling step is

$$\widetilde{\mathbf{x}}_s = \underbrace{\sqrt{\bar{\alpha}_s} \frac{\mathbf{x}_t - \sqrt{1 - \bar{\alpha}_t}\, \widetilde{\boldsymbol{\epsilon}}_{\theta_{A(\mathcal{S})}^\star}^{(t)}}{\sqrt{\bar{\alpha}_t}}}_{\text{predicted } \mathbf{x}_0 \text{ contribution}} + \underbrace{\sqrt{1 - \bar{\alpha}_s - \sigma_{t,s}^2}\, \widetilde{\boldsymbol{\epsilon}}_{\theta_{A(\mathcal{S})}^\star}^{(t)}}_{\text{direction pointing to } \mathbf{x}_t} + \underbrace{\sigma_{t,s} \boldsymbol{\epsilon}_s}_{\text{random noise}}, \tag{9}$$

where $\boldsymbol{\epsilon}_s \sim \mathcal{N}(0, \mathbf{I}_d)$.

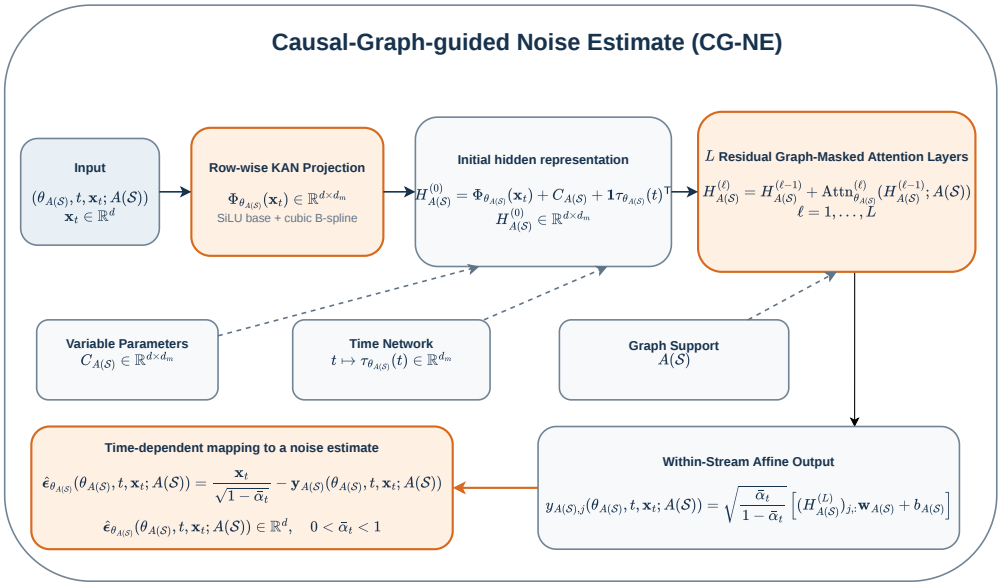


Figure 1: Causal-Graph-guided Noise Estimator (CG-NE)

At the final reverse step, where $s = 0$, we have $\bar{\alpha}_0 = 1$ and therefore $\sigma_{t,0} = 0$. Equation (9) then reduces to Equation (10):

$$\widetilde{\mathbf{x}}_0 = \underbrace{\frac{\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\,\widetilde{\boldsymbol{\epsilon}}^{(t)}_{\theta^\star_{A(\mathcal{S})}}}{\sqrt{\bar{\alpha}_t}}}_{\text{predicted } \mathbf{x}_0 \text{ contribution}}. \tag{10}$$

Considering we have observed values, we use noisy samples to replace observed values at each step of our backward process. Hence, we can fully use observed values.

$$\mathbf{x}_s = \begin{cases} \mathbf{m} \odot \left(\sqrt{\bar{\alpha}_s}\mathbf{x}_{\text{obs}} + \sqrt{1-\bar{\alpha}_s}\boldsymbol{\xi}_s\right) + (\mathbf{1}-\mathbf{m}) \odot \widetilde{\mathbf{x}}_s, & s > 0, \\ \mathbf{x}_{\text{obs}} + (\mathbf{1}-\mathbf{m}) \odot \widetilde{\mathbf{x}}_0, & s = 0, \end{cases}$$

where $\boldsymbol{\xi}_s \sim \mathcal{N}(0, \mathbf{I}_d)$ is sampled independently to supply the noise in the time-matched observed values for $s > 0$.

We generate N_{gen} complete value vectors $\{\mathbf{x}_0^{(i)}\}_{i=1}^{N_{\text{gen}}}$ through the backward process, and use their elementwise median as the final imputed vector:

$$\bar{x}_{0,j} = \text{median}\left\{x_{0,j}^{(1)}, \ldots, x_{0,j}^{(N_{\text{gen}})}\right\}, \qquad j = 1, \ldots, d$$
$$\bar{\mathbf{x}}_0 = (\bar{x}_{0,1}, \ldots, \bar{x}_{0,d})^\top.$$

For the ith row to be imputed, we use the previous backward process to generate the final imputed vector:

$$\widehat{\mathbf{x}}_i^{\text{imp}} = \bar{\mathbf{x}}_0^{(i)}, \quad i = 1, \ldots, N_{\text{imp}}.$$

We finally stack the returned vectors $(\widehat{\mathbf{x}}_i^{\text{imp}})_{i=1}^{N_{\text{imp}}}$ to get the completed value matrix:

$$\widehat{\mathbf{X}}^{\text{imp}} = \left[\widehat{\mathbf{x}}_1^{\text{imp}}, \ldots, \widehat{\mathbf{x}}_{N_{\text{imp}}}^{\text{imp}}\right]^\top \in \mathbb{R}^{N_{\text{imp}} \times d}.$$

5 Experiments

5.1 Controlled-masking Protocol

Synthetic data. We use 15 preselected initial-DAG seeds for each of four structural-equation families (linear Gaussian, nonlinear MLP, nonlinear MIM, and nonlinear quadratic), each with $N = 1000$

Table 1: Synthetic imputation RMSE (lower is better). Sample mean ± sample SD. Bold marks the lowest mean.

	Linear Gaussian			**Nonlinear MIM**		
Method	$\rho = 0.1$	$\rho = 0.3$	$\rho = 0.5$	$\rho = 0.1$	$\rho = 0.3$	$\rho = 0.5$
CG-DIFF ($A(\mathrm{MB})$)	**1.060±0.082**	**1.168±0.086**	**1.365±0.143**	**1.078±0.112**	1.171±0.087	1.421±0.143
CG-DIFF ($A(\mathrm{Pa})$)	1.145±0.087	1.219±0.080	1.437±0.119	1.112±0.102	**1.153±0.105**	**1.351±0.131**
Mean	2.104±0.030	2.104±0.021	2.282±0.025	1.978±0.073	1.998±0.069	2.199±0.066
MICE	1.283±0.398	1.538±0.386	2.323±0.132	1.971±0.182	2.089±0.157	2.326±0.115
GAIN	1.750±0.288	2.632±0.159	2.877±0.134	1.979±0.212	2.188±0.065	2.723±0.152
MIWAE	1.359±0.414	1.667±0.279	2.275±0.109	1.772±0.192	1.871±0.150	2.141±0.120
MIRACLE	1.146±0.364	1.569±0.260	2.283±0.060	1.662±0.191	1.879±0.185	2.167±0.120
OT-Imputer	1.522±0.276	1.821±0.200	2.156±0.093	1.791±0.207	1.875±0.177	2.134±0.122
NewImp	1.342±0.320	1.715±0.183	2.272±0.061	1.756±0.153	1.839±0.147	2.164±0.111
TabDDPM	1.827±0.322	2.645±0.834	3.279±0.607	2.034±0.175	2.226±0.263	2.535±0.319
GRIMP	2.082±0.042	1.934±0.116	2.127±0.119	1.978±0.075	1.989±0.082	2.170±0.085
	Nonlinear MLP			**Nonlinear Quadratic**		
Method	$\rho = 0.1$	$\rho = 0.3$	$\rho = 0.5$	$\rho = 0.1$	$\rho = 0.3$	$\rho = 0.5$
CG-DIFF ($A(\mathrm{MB})$)	**1.069±0.188**	**1.200±0.185**	**1.521±0.226**	1.823±1.820	**1.638±1.230**	**1.760±1.016**
CG-DIFF ($A(\mathrm{Pa})$)	1.124±0.157	1.298±0.187	1.538±0.209	**1.821±1.847**	1.639±1.232	1.763±1.010
Mean	2.097±0.171	2.161±0.153	2.420±0.201	2.674±1.805	2.427±1.205	2.497±0.981
MICE	1.687±0.310	1.871±0.300	2.548±0.204	2.799±1.760	2.602±1.167	2.663±0.965
GAIN	1.828±0.274	2.562±0.247	2.954±0.244	2.855±1.739	2.635±1.186	2.947±0.966
MIWAE	1.514±0.309	1.752±0.311	2.444±0.204	2.604±1.800	2.406±1.192	2.496±0.966
MIRACLE	1.448±0.301	1.776±0.302	2.418±0.193	2.511±1.785	2.433±1.163	2.514±0.957
OT-Imputer	1.684±0.315	1.990±0.235	2.359±0.205	2.635±1.788	2.391±1.194	2.466±0.980
NewImp	1.634±0.246	1.904±0.232	2.417±0.207	2.574±1.785	2.362±1.186	2.498±0.953
TabDDPM	1.920±0.344	2.255±0.304	2.757±0.360	2.892±1.751	3.011±1.344	3.600±1.382
GRIMP	2.054±0.209	2.034±0.195	2.297±0.198	2.675±1.804	2.433±1.201	2.492±0.983

rows and $d = 10$ variables. The same initial-DAG seeds and run seeds are shared across the four families. The structural equations and simulation details are given in Appendix C.4. For missingness rates $\rho \in \{0.1, 0.3, 0.5\}$, we generate Self-Masking missingness using

$$s_{i,j} = z_j(x^\star_{i,j}) + 0.5\xi_{i,j}, \qquad \xi_{i,j} \sim \mathcal{N}(0,1), \qquad m_{i,j} = \mathbb{1}[s_{i,j} \le q_{j,1-\rho}], \tag{11}$$

where z_j standardizes column j of the complete data and $q_{j,1-\rho}$ is the empirical $(1-\rho)$ quantile of the scores in that column. Thus, values with larger scores are more likely to be missing. At each missingness rate, all methods are trained and evaluated on the same mask seed.

Real data. We generate Self-Masking missingness in the complete ADNI dataset (414 rows, 8 variables) using the same masking procedure as in the synthetic experiments. The supplied graphs contain 9 literature-supported edges for ADNI, which is detailed in Appendix C.5. At each missingness rate, all methods are trained and evaluated on the same 10 mask seeds.

Comparisons and evaluation. We compare CG-DIFF with Mean, MICE (van Buuren & Groothuis-Oudshoorn, 2011), GAIN (Yoon et al., 2018), MIWAE (Mattei & Frellsen, 2019), MIRACLE (Kyono et al., 2021), OT-Imputer (Muzellec et al., 2020), NewImp (Chen et al., 2024), and TabDDPM (Kotelnikov et al., 2023) and GRIMP (Cappuzzo et al., 2024). For all datasets, the training/validation/evaluation split follows 3:1:1. Tunable methods select hyperparameters on the validation set, and the selected hyperparameters are then fixed when computing RMSE on the evaluation set. We compute the RMSE for the artificially hidden entries on the evaluation set. For synthetic data, we calculate the sample mean and SD across 25 initial-DAG seeds. For ADNI, we compute the sample mean and SD over 10 mask seeds. Training and sampling details are provided in Appendix C.1.

5.2 Results

Synthetic data. As reported in Table 1, CG-DIFF ($A(\mathrm{MB})$) achieves the best sample mean in most settings, with only a few cases where CG-DIFF ($A(\mathrm{Pa})$) performs slightly better. Overall, the

results demonstrate the usefulness of causal guidance in diffusion-based imputation and favor the Markov-blanket-plus-self structure in most cases.

Real data. As shown in Tables 2, CG-DIFF ($A(\mathrm{MB})$) achieves the lowest sample mean at all missingness rates. These results suggest that incorporating causal-graph information into diffusion-based imputation can improve imputation performance. Moreover, using the Markov-blanket-plus-self structure generally leads to better performance.

Table 2: ADNI imputation RMSE. Sample mean ± sample SD. Bold marks the lowest mean.

Method	$\rho = 0.1$	$\rho = 0.3$	$\rho = 0.5$
CG-DIFF ($A(\mathrm{MB})$)	**1.282±0.059**	**1.485±0.048**	**2.072±0.149**
CG-DIFF ($A(\mathrm{Pa})$)	1.324±0.092	1.529±0.077	2.243±0.157
Mean	2.338±0.034	2.525±0.026	3.146±0.116
MICE	2.097±0.046	2.435±0.031	3.250±0.122
GAIN	2.052±0.130	2.556±0.121	3.375±0.218
MIWAE	2.026±0.075	2.341±0.057	3.129±0.125
MIRACLE	1.891±0.054	2.317±0.086	3.056±0.158
OT-Imputer	2.196±0.112	2.494±0.024	3.146±0.115
NewImp	2.029±0.056	2.426±0.045	3.143±0.139
TabDDPM	2.195±0.042	2.573±0.080	3.243±0.164
GRIMP	2.266±0.043	2.427±0.050	3.075±0.120

6 Conclusion

In this work, we propose CG-DIFF, a causal-graph-guided diffusion model for tabular missing-value imputation under Self-Masking MNAR. CG-DIFF introduces a mask into the forward process so that the model can be trained using observed values while still generating complete data through the reverse process. To incorporate structural information from the causal graph, we introduce a Causal-Graph-guided Noise Estimator (CG-NE), which combines Kolmogorov–Arnold network (KAN) projections with graph-masked attention to estimate the diffusion noise. During the reverse process, the values at observed positions are replaced with their forward-noised values at the corresponding timestep, allowing the observed information to be preserved throughout sampling. We further establish sufficient conditions under which the population predictor based on the Markov blanket has lower missing-case squared-error risk than the parent-based predictor, providing theoretical motivation for the Markov-blanket-plus-self graph support. Experiments on simulated datasets and one real-world dataset show that CG-DIFF with the Markov-blanket-plus-self graph support outperforms competing methods in most settings.

AI use statement

Generative-AI tools assisted with software revisions, experimental-protocol checks and orchestration, result auditing and interpretation, literature retrieval, scientific-figure layout, and manuscript drafting and editing. Procedural checks for scientific writing followed the Scientific Agent Skills framework (Kassis et al., 2026). The AI-assisted audit checked the implementation, raw result files, cited sources, and compiled paper. The authors retain responsibility for verifying and approving the final text, claims, code, and figures before submission.

Reproducibility statement

The manuscript specifies the training and imputation algorithms, parameter initialization, mask construction, model-selection protocol, and aggregation units. The supplementary material records every frozen synthetic and empirical configuration together with reconstruction, provenance, and hash checks. The reported results in the main text and appendix retain all audited method–condition values.

ETHICS STATEMENT

The experiments are retrospective analyses of synthetic or secondary research values and involve no new human-participant value collection.

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Laura E. Brown and Ioannis Tsamardinos. A strategy for making predictions under manipulation. In *Proceedings of the Workshop on the Causation and Prediction Challenge at WCCI 2008*, pp. 35–52, 2008.

Riccardo Cappuzzo, Saravanan Thirumuruganathan, and Paolo Papotti. Relational data imputation with graph neural networks. In *International Conference on Extending Database Technology*, pp. 221–233, 2024.

Zhichao Chen, Haoxuan Li, Fangyikang Wang, Odin Zhang, Hu Xu, Xiaoyu Jiang, Zhihuan Song, and Hao Wang. Rethinking the diffusion models for missing data imputation: A gradient flow perspective. In *Advances in Neural Information Processing Systems*, pp. 112050–112103, 2024.

Daniel Ferreira, Oskar Hansson, José Barroso, Yaiza Molina, Alejandra Machado, Juan Andrés Hernández-Cabrera, J.-Sebastian Muehlboeck, Erik Stomrud, Katarina Nägga, Olof Lindberg, David Ames, Grégoria Kalpouzos, Laura Fratiglioni, Lars Bäckman, Caroline Graff, Patrizia Mecocci, Bruno Vellas, Magda Tsolaki, Iwona Kłoszewska, Hilkka Soininen, Simon Lovestone, Håkan Ahlström, Lars Lind, Elna-Marie Larsson, and Lars-Olof Wahlund. The interactive effect of demographic and clinical factors on hippocampal volume: A multicohort study on 1958 cognitively normal individuals. *Hippocampus*, 27(6):653–667, 2017.

Guia Guffanti, Federica Torri, Jerod Rasmussen, Andrew P. Clark, Anita Lakatos, Jessica A. Turner, James H. Fallon, Andrew J. Saykin, Michael Weiner, ADNI the Alzheimer's Disease Neuroimaging Initiative, Marquis P. Vawter, James A. Knowles, Steven G. Potkin, and Fabio Macciardi. Increased CNV-region deletions in mild cognitive impairment (MCI) and Alzheimer's disease (AD) subjects in the ADNI sample. *Genomics*, 102(2):112–122, 2013.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *IEEE International Conference on Computer Vision*, pp. 1026–1034, 2015.

W. J. P. Henneman, H. Vrenken, J. Barnes, I. C. Sluimer, N. A. Verwey, M. A. Blankenstein, M. Klein, N. C. Fox, P. Scheltens, F. Barkhof, and W. M. van der Flier. Baseline CSF p-tau levels independently predict progression of hippocampal atrophy in Alzheimer disease. *Neurology*, 73 (12):935–940, 2009.

Xue Hua, Derrek P. Hibar, Suh Lee, Arthur W. Toga, Clifford R. Jack, Michael W. Weiner, Paul M. Thompson, and Alzheimer's Disease Neuroimaging Initiative. Sex and age differences in atrophic rates: An ADNI study with $n = 1368$ MRI scans. *Neurobiology of Aging*, 31(8):1463–1480, 2010.

Clifford R. Jack and David M. Holtzman. Biomarker modeling of Alzheimer's disease. *Neuron*, 80 (6):1347–1358, 2013.

Clifford R. Jack, Jr., David S. Knopman, William J. Jagust, Leslie M. Shaw, Paul S. Aisen, Michael W. Weiner, Ronald C. Petersen, and John Q. Trojanowski. Hypothetical model of dynamic biomarkers of the Alzheimer's pathological cascade. *The Lancet Neurology*, 9(1):119–128, 2010.

Daniel Jarrett, Bogdan C. Cebere, Tennison Liu, Alicia Curth, and Mihaela van der Schaar. HyperImpute: Generalized iterative imputation with automatic model selection. In *International Conference on Machine Learning*, pp. 9916–9937, 2022.

Timothy Kassis, Vinayak Agarwal, Yuhuan He, Darshil Patel, and Aubrey M Brueckner. Scientific Agent Skills: A library of procedural knowledge for research agents. *arXiv preprint arXiv:2609.00065*, 2026.

Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. TabDDPM: Modelling tabular data with diffusion models. In *International Conference on Machine Learning*, pp. 17564–17579, 2023.

Trent Kyono, Yao Zhang, Alexis Bellot, and Mihaela van der Schaar. MIRACLE: Causally-aware imputation via learning missing data mechanisms. In *Advances in Neural Information Processing Systems*, pp. 23806–23817, 2021.

Roderick J. A. Little and Donald B. Rubin. *Statistical Analysis with Missing Data*. Wiley, 2019.

Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljacic, Thomas Hou, and Max Tegmark. KAN: Kolmogorov–arnold networks. In *International Conference on Learning Representations*, 2025.

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, 2019.

Karthika Mohan and Judea Pearl. Graphical models for processing missing data. *Journal of the American Statistical Association*, 116(534):1023–1037, 2021.

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

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2009.

Elaine R. Peskind, Ge Li, Jane Shofer, Joseph F. Quinn, Jeffrey A. Kaye, Chris M. Clark, Martin R. Farlow, Charles DeCarli, Murray A. Raskind, Gerard D. Schellenberg, Virginia M.-Y. Lee, and Douglas R. Galasko. Age and apolipoprotein E*4 allele effects on cerebrospinal fluid beta-amyloid 42 in adults with normal cognition. *Archives of Neurology*, 63(7):936–939, 2006.

Shannon L. Risacher, Wesley H. Anderson, Arnaud Charil, Peter F. Castelluccio, Sergey Shcherbinin, Andrew J. Saykin, Adam J. Schwarz, and Alzheimer's Disease Neuroimaging Initiative. Alzheimer disease brain atrophy subtypes are associated with cognition and rate of decline. *Neurology*, 89(21):2176–2186, 2017.

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

Shaun Seaman, John Galati, Dan Jackson, and John Carlin. What is meant by "missing at random"? *Statistical Science*, 28(2):257–268, 2013.

Jasper D. Sluimer, Wiesje M. van der Flier, Giorgos B. Karas, Nick C. Fox, Philip Scheltens, Frederik Barkhof, and Hugo Vrenken. Whole-brain atrophy rate and cognitive decline: Longitudinal MR study of memory clinic patients. *Radiology*, 248(2):590–598, 2008.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

Aude Sportisse, Claire Boyer, and Julie Josse. Estimation and imputation in probabilistic principal component analysis with missing not at random data. In *Advances in Neural Information Processing Systems*, pp. 7067–7077, 2020.

Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. CSDI: Conditional score-based diffusion models for probabilistic time series imputation. In *Advances in Neural Information Processing Systems*, pp. 24804–24816, 2021.

Ioannis Tsamardinos and Constantin F. Aliferis. Towards principled feature selection: Relevancy, filters and wrappers. In *International Workshop on Artificial Intelligence and Statistics*, pp. 300–307, 2003.

Stef van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in R. *Journal of Statistical Software*, 45(3):1–67, 2011.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Jinsung Yoon, James Jordon, and Mihaela van der Schaar. GAIN: Missing data imputation using generative adversarial nets. In *International Conference on Machine Learning*, pp. 5689–5698, 2018.

Keli Zhang, Shengyu Zhu, Marcus Kalander, Ignavier Ng, Junjian Ye, Zhitang Chen, and Lujia Pan. gCastle: A python toolbox for causal discovery. *arXiv preprint arXiv:2111.15155*, 2021.

Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. DAGs with NO TEARS: Continuous optimization for structure learning. In *Advances in Neural Information Processing Systems*, pp. 9472–9483, 2018.

A Proof of Theorem 3.1

Proof. By the definition of $\pi_{i,j}$,

$$\begin{aligned}
&\Pr(m_{i,j}=1) = \mathbb{E}\big[\mathbb{E}[m_{i,j} \mid x^\star_{i,j}, \mathcal{I}_{\mathrm{MB}}(i,j)]\big] = \mathbb{E}[\pi_{i,j}] \in (0,1),\\
&\Pr(m_{i,j}=0) = 1 - \Pr(m_{i,j}=1) \in (0,1),\\
&\qquad 0 < w_{i,j} = \frac{1-\pi_{i,j}}{\pi_{i,j}} < \infty \qquad \text{almost surely.}
\end{aligned}$$

For any nonnegative $\sigma(x^\star_{i,j}, \mathcal{I}_{\mathrm{MB}}(i,j))$-measurable random variable U,

$$\begin{aligned}
\Pr(m_{i,j}=1)\mathbb{E}[w_{i,j}U \mid m_{i,j}=1] &= \mathbb{E}[m_{i,j}w_{i,j}U]\\
&= \mathbb{E}\big[w_{i,j}U\,\mathbb{E}[m_{i,j} \mid x^\star_{i,j}, \mathcal{I}_{\mathrm{MB}}(i,j)]\big]\\
&= \mathbb{E}[\pi_{i,j}w_{i,j}U]\\
&= \mathbb{E}[(1-\pi_{i,j})U]\\
&= \mathbb{E}\big[U\,\mathbb{E}[1-m_{i,j} \mid x^\star_{i,j}, \mathcal{I}_{\mathrm{MB}}(i,j)]\big]\\
&= \mathbb{E}[(1-m_{i,j})U]\\
&= \Pr(m_{i,j}=0)\mathbb{E}[U \mid m_{i,j}=0].
\end{aligned}$$

Taking $U=1$ gives

$$\mathbb{E}[w_{i,j} \mid m_{i,j}=1] = \frac{\Pr(m_{i,j}=0)}{\Pr(m_{i,j}=1)} \in (0,\infty),$$

which proves (3). Hence

$$\mathbb{E}[U \mid m_{i,j}=0] = \frac{\mathbb{E}[w_{i,j}U \mid m_{i,j}=1]}{\mathbb{E}[w_{i,j} \mid m_{i,j}=1]}. \tag{12}$$

For every $A \in \sigma(x^\star_{i,j}, \mathcal{I}_{\mathrm{MB}}(i,j))$, let $\mathbb{1}_A$ denote its indicator. Then

$$\begin{aligned}
\Pr(A \mid m_{i,j}=0) = 0 &\iff \mathbb{E}[w_{i,j}\mathbb{1}_A \mid m_{i,j}=1] = 0\\
&\iff \Pr(A \mid m_{i,j}=1) = 0.
\end{aligned}$$

Thus the observed-case versions of $\mu_{\mathcal{S}}(i,j)$ are also unique almost surely under the missing-case law.

For $\mathcal{S} \in \{\mathrm{Pa}, \mathrm{MB}\}$,

$$\mathbb{E}[(x^\star_{i,j})^2 \mid m_{i,j}=1] = \frac{\mathbb{E}[m_{i,j}(x^\star_{i,j})^2]}{\Pr(m_{i,j}=1)} \le \frac{\mathbb{E}[(x^\star_{i,j})^2]}{\Pr(m_{i,j}=1)} < \infty.$$

Conditional Jensen's inequality gives

$$\begin{aligned}\mathbb{E}[\mu_{\mathcal{S}}(i,j)^2 \mid m_{i,j}=1] &= \mathbb{E}\Big[\big\{\mathbb{E}[x^\star_{i,j} \mid \mathcal{I}_{\mathcal{S}}(i,j), m_{i,j}=1]\big\}^2 \;\Big|\; m_{i,j}=1\Big] \\ &\le \mathbb{E}\Big[\mathbb{E}[(x^\star_{i,j})^2 \mid \mathcal{I}_{\mathcal{S}}(i,j), m_{i,j}=1] \;\Big|\; m_{i,j}=1\Big] \\ &= \mathbb{E}[(x^\star_{i,j})^2 \mid m_{i,j}=1].\end{aligned}$$

Consequently,

$$\begin{aligned}R^{(1)}_{\mathcal{S}}(j) &\le 2\mathbb{E}[(x^\star_{i,j})^2 \mid m_{i,j}=1] + 2\mathbb{E}[\mu_{\mathcal{S}}(i,j)^2 \mid m_{i,j}=1] \\ &\le 4\mathbb{E}[(x^\star_{i,j})^2 \mid m_{i,j}=1] < \infty,\end{aligned}$$

and

$$\begin{aligned}&\mathbb{E}\big[\{\mu_{\mathrm{MB}}(i,j)-\mu_{\mathrm{Pa}}(i,j)\}^2 \mid m_{i,j}=1\big] \\ &\le 2\mathbb{E}[\mu_{\mathrm{MB}}(i,j)^2 \mid m_{i,j}=1] + 2\mathbb{E}[\mu_{\mathrm{Pa}}(i,j)^2 \mid m_{i,j}=1] < \infty.\end{aligned}$$

Cauchy–Schwarz gives

$$\begin{aligned}&\mathbb{E}\big[\big|\{x^\star_{i,j}-\mu_{\mathrm{MB}}(i,j)\}\{\mu_{\mathrm{MB}}(i,j)-\mu_{\mathrm{Pa}}(i,j)\}\big| \mid m_{i,j}=1\big] \\ &\le \Big\{R^{(1)}_{\mathrm{MB}}(j)\mathbb{E}\big[\{\mu_{\mathrm{MB}}(i,j)-\mu_{\mathrm{Pa}}(i,j)\}^2 \mid m_{i,j}=1\big]\Big\}^{1/2} < \infty.\end{aligned}$$

Moreover,

$$\mathcal{I}_{\mathrm{Pa}}(i,j) \subseteq \mathcal{I}_{\mathrm{MB}}(i,j), \qquad \mathbb{E}[x^\star_{i,j}-\mu_{\mathrm{MB}}(i,j) \mid \mathcal{I}_{\mathrm{MB}}(i,j), m_{i,j}=1] = 0.$$

Therefore,

$$\begin{aligned}&\mathbb{E}\big[\{x^\star_{i,j}-\mu_{\mathrm{MB}}(i,j)\}\{\mu_{\mathrm{MB}}(i,j)-\mu_{\mathrm{Pa}}(i,j)\} \mid m_{i,j}=1\big] \\ &= \mathbb{E}\Big[\{\mu_{\mathrm{MB}}(i,j)-\mu_{\mathrm{Pa}}(i,j)\}\mathbb{E}[x^\star_{i,j}-\mu_{\mathrm{MB}}(i,j) \mid \mathcal{I}_{\mathrm{MB}}(i,j), m_{i,j}=1] \;\Big|\; m_{i,j}=1\Big] \\ &= 0.\end{aligned}$$

Since

$$\begin{aligned}H_{i,j} &= \big\{x^\star_{i,j}-\mu_{\mathrm{MB}}(i,j)+\mu_{\mathrm{MB}}(i,j)-\mu_{\mathrm{Pa}}(i,j)\big\}^2 - \{x^\star_{i,j}-\mu_{\mathrm{MB}}(i,j)\}^2 \\ &= \{\mu_{\mathrm{MB}}(i,j)-\mu_{\mathrm{Pa}}(i,j)\}^2 + 2\{x^\star_{i,j}-\mu_{\mathrm{MB}}(i,j)\}\{\mu_{\mathrm{MB}}(i,j)-\mu_{\mathrm{Pa}}(i,j)\},\end{aligned}$$

we obtain

$$\begin{aligned}\Delta_j &= R^{(1)}_{\mathrm{Pa}}(j) - R^{(1)}_{\mathrm{MB}}(j) \\ &= \mathbb{E}[H_{i,j} \mid m_{i,j}=1] \\ &= \mathbb{E}\big[\{\mu_{\mathrm{MB}}(i,j)-\mu_{\mathrm{Pa}}(i,j)\}^2 \mid m_{i,j}=1\big] \ge 0,\end{aligned}$$

proving (1).

For $r \in \{0,1\}$,

$$\mathbb{E}[|H_{i,j}| \mid m_{i,j}=r] \le R^{(r)}_{\mathrm{Pa}}(j) + R^{(r)}_{\mathrm{MB}}(j) < \infty.$$

The variables $H_{i,j}$ and $|H_{i,j}|$ are $\sigma(x^\star_{i,j}, \mathcal{I}_{\mathrm{MB}}(i,j))$-measurable. Applying (12) to $|H_{i,j}|$ gives

$$\mathbb{E}[w_{i,j}|H_{i,j}| \mid m_{i,j}=1] = \mathbb{E}[w_{i,j} \mid m_{i,j}=1]\mathbb{E}[|H_{i,j}| \mid m_{i,j}=0] < \infty.$$

In particular,

$$\begin{aligned}&\mathbb{E}[|\{w_{i,j}-\mathbb{E}[w_{i,j} \mid m_{i,j}=1]\}(H_{i,j}-\Delta_j)| \mid m_{i,j}=1] \\ &\le \mathbb{E}[w_{i,j}|H_{i,j}| \mid m_{i,j}=1] + \mathbb{E}[w_{i,j} \mid m_{i,j}=1]\mathbb{E}[|H_{i,j}| \mid m_{i,j}=1] + 2\Delta_j\mathbb{E}[w_{i,j} \mid m_{i,j}=1] < \infty.\end{aligned}$$

Thus

$$\begin{aligned}&\mathrm{Cov}(w_{i,j}, H_{i,j} \mid m_{i,j}=1) \\ &= \mathbb{E}[w_{i,j}H_{i,j} \mid m_{i,j}=1] - \mathbb{E}[w_{i,j} \mid m_{i,j}=1]\mathbb{E}[H_{i,j} \mid m_{i,j}=1] \\ &= \mathbb{E}[w_{i,j}H_{i,j} \mid m_{i,j}=1] - \mathbb{E}[w_{i,j} \mid m_{i,j}=1]\Delta_j.\end{aligned}$$

Applying (12) to each squared error gives

$$R^{(0)}_{\mathcal{S}}(j) = \frac{\mathbb{E}\big[w_{i,j}\{x^{\star}_{i,j} - \mu_{\mathcal{S}}(i,j)\}^2 \mid m_{i,j}=1\big]}{\mathbb{E}[w_{i,j} \mid m_{i,j}=1]}, \qquad \mathcal{S} \in \{\mathrm{Pa}, \mathrm{MB}\}.$$

Subtracting these finite quantities yields

$$R^{(0)}_{\mathrm{Pa}}(j) - R^{(0)}_{\mathrm{MB}}(j) = \frac{\mathbb{E}[w_{i,j}H_{i,j} \mid m_{i,j}=1]}{\mathbb{E}[w_{i,j} \mid m_{i,j}=1]} = \Delta_j + \frac{\mathrm{Cov}(w_{i,j}, H_{i,j} \mid m_{i,j}=1)}{\mathbb{E}[w_{i,j} \mid m_{i,j}=1]},$$

which proves (2). Under (4),

$$R^{(0)}_{\mathrm{Pa}}(j) - R^{(0)}_{\mathrm{MB}}(j) - \Delta_j = \frac{\mathrm{Cov}(w_{i,j}, H_{i,j} \mid m_{i,j}=1)}{\mathbb{E}[w_{i,j} \mid m_{i,j}=1]} \geq 0.$$

This completes the proof. □

B ALGORITHMS

The training process of Causal-Graph-guided Noise Estimator (CG-NE) is presented in Algorithm 1. Noise estimate $\widehat{\boldsymbol{\epsilon}}_{\theta_{A(\mathcal{S})}}(\theta_{A(\mathcal{S})}, t, \mathbf{x}_t; A(\mathcal{S}))$ = CG-NE$(\theta_{A(\mathcal{S})}, t, \mathbf{x}_t; A(\mathcal{S}))$. The CG-DIFF imputation with DDIM sampling is summarized in Algorithm 2.

Algorithm 1 training process of Causal-Graph-guided Noise Estimator (CG-NE).

Input: Observed pairs $\{(\mathbf{x}^{\mathrm{train}}_{i,\mathrm{obs}}, \mathbf{m}^{\mathrm{train}}_{i})\}_{i=1}^{N_{\mathrm{train}}}$, graph support $A(\mathcal{S})$, bach size B, epoch E
Output: Optimal parameters of Causal-Graph-guided Noise Estimator (CG-NE) $\theta^{\star}_{A(\mathcal{S})}$

Calculate the number of mini-bach $Q = N_{\mathrm{train}}/B$
for $j = 1, \ldots, E$ **do**
  Randomly divide $\{(\mathbf{x}^{\mathrm{train}}_{i,\mathrm{obs}}, \mathbf{m}^{\mathrm{train}}_{i})\}_{i=1}^{N_{\mathrm{train}}}$ into Q mini-bach $\{(\mathbf{x}^{\mathrm{train}}_{n,\mathrm{obs}}, \mathbf{m}^{\mathrm{train}}_{n})\}_{n=1}^{B}$
  for $k = 1, \ldots, Q$ **do**
    Draw $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
    Draw $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}_d)$
    for $n = 1, \ldots, B$ **do**
      $\mathbf{x}_{n,t} = \mathbf{m}^{\mathrm{train}}_{n} \odot \left(\sqrt{\bar{\alpha}_t}\mathbf{x}^{\mathrm{train}}_{n,\mathrm{obs}} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}\right)$
      $\widehat{\boldsymbol{\epsilon}}_{n,\theta_{A(\mathcal{S})}} = \text{CG-NE}(\theta_{A(\mathcal{S})}, t, \mathbf{x}_{n,t}; A(\mathcal{S}))$
    end for
    Calculate $\mathcal{L}(\theta_{A(\mathcal{S})}) = \frac{1}{B}\sum_{n=1}^{B}\left\|\mathbf{m}^{\mathrm{train}}_{n} \odot \left(\widehat{\boldsymbol{\epsilon}}_{n,\theta_{A(\mathcal{S})}}(\theta_{A(\mathcal{S})}, t, \mathbf{x}_{n,t}; A(\mathcal{S})) - \boldsymbol{\epsilon}\right)\right\|_2^2$
    Calculate $\nabla\mathcal{L}(\theta_{A(\mathcal{S})})$ and update the parameter $\theta^{\star}_{A(\mathcal{S})}$
  end for
end for
return $\theta^{\star}_{A(\mathcal{S})}$

C IMPLEMENTATION DETAILS

C.1 IMPLEMENTATION AND TRAINING DETAILS

Each training update uses a minibatch of rows. Let $\mathbf{X}_{\mathrm{obs},b} \in \mathbb{R}^{B\times d}$ contain B rows sampled from $\mathbf{X}^{\mathrm{train}}_{\mathrm{obs}}$, and let $\mathbf{M}_b \in \{0,1\}^{B\times d}$ contain the corresponding rows of $\mathbf{M}^{\mathrm{train}}$. For any matrix $\mathbf{A}$, $(\mathbf{A})_{i,:}$ denotes its ith row. For each row i, we independently sample t_i uniformly from $\{1, \ldots, T\}$ and write $\mathbf{t} = (t_1, \ldots, t_B)$. We also draw an independent Gaussian noise vector for each row. Missing positions are set to zero before noising and remain zero after the mask is applied:

$$(\mathbf{X}_{\mathbf{t},b})^{\mathsf{T}}_{i,:} = (\mathbf{M}_b)^{\mathsf{T}}_{i,:} \odot \left[\sqrt{\bar{\alpha}_{t_i}}\,(\mathbf{X}_{\mathrm{obs},b})^{\mathsf{T}}_{i,:} + \sqrt{1-\bar{\alpha}_{t_i}}\,\boldsymbol{\epsilon}_{b,i}\right], \qquad \boldsymbol{\epsilon}_{b,i} \sim \mathcal{N}(0, \mathbf{I}_d). \tag{13}$$

The network also receives a conditioning mask that indicates which values are observed. During training, each observed entry in this conditioning mask is independently hidden with probability 0.2:

$$\mathbf{M}^{\mathrm{cond}}_b = \mathbf{M}_b \odot D_b, \qquad (D_b)_{ij} \sim \mathrm{Bernoulli}(0.8),$$

where $\mathrm{Bernoulli}(p)$ denotes a Bernoulli distribution that equals 1 with probability p and 0 with probability $1-p$. This modification is applied only to the conditioning mask supplied to the network;

Algorithm 2 CG-DIFF imputation with DDIM sampling.

Input: Imputed pairs$(\mathbf{x}_{i,\text{obs}}^{\text{imp}}, \mathbf{m}_i^{\text{imp}})_{i=1}^{N_{\text{imp}}}$, optimal parameters of Causal-Graph-guided Noise Estimator (CG-NE) $\theta^\star_{A(\mathcal{S})}$, graph support $A(\mathcal{S})$, sample time steps $\tau_0 > \cdots > \tau_{J-1} = 0$, random noise scales $\{\sigma_{t,s}\}$, estimator bound $[l(\mathbf{x}_t, t), u(\mathbf{x}_t, t)]$, generation number N_{gen}

Output: completed standardized table $\widehat{\mathbf{X}}^{\text{imp}}$

for $i = 1, \ldots, N_{\text{imp}}$ **do**
  for $j = 1, \ldots, N_{\text{gen}}$ **do**
    Draw $\mathbf{x}_{\tau_0}^{(j)} \sim \mathcal{N}(0, \mathbf{I}_d)$
    for $k = 0, \ldots, J-2$ **do**
      $(t, s) = (\tau_k, \tau_{k+1})$
      $\boldsymbol{\epsilon}_{\theta^\star_{A(\mathcal{S})}}^{(t,j)} = \text{CG-NE}(\theta^\star_{A(\mathcal{S})}, t, \mathbf{x}_t^{(j)}; A(\mathcal{S}))$
      $\widetilde{\boldsymbol{\epsilon}}_{\theta^\star_{A(\mathcal{S})}}^{(t,j)}(\mathbf{x}_t^{(j)}, t; A(\mathcal{S})) = \text{clip}\left(l\left(\mathbf{x}_t^{(j)}, t\right), \widehat{\boldsymbol{\epsilon}}_{\theta^\star_{A(\mathcal{S})}}^{(t)}(\mathbf{x}_t^{(j)}, t; A(\mathcal{S})), u\left(\mathbf{x}_t^{(j)}, t\right)\right)$
      Draw $\boldsymbol{\epsilon}_s^{(j)} \sim \mathcal{N}(0, \mathbf{I}_d)$
      $\sigma_{t,s} = 0$ if $s = 0$
      $\widetilde{\mathbf{x}}_s^{(j)} = \sqrt{\bar{\alpha}_s}\,\dfrac{\mathbf{x}_t^{(j)} - \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}_{\theta^\star_{A(\mathcal{S})}}^{(t,j)}}{\sqrt{\bar{\alpha}_t}} + \sqrt{1-\bar{\alpha}_s-\sigma_{t,s}^2}\,\widetilde{\boldsymbol{\epsilon}}_{\theta^\star_{A(\mathcal{S})}}^{(t,j)} + \sigma_{t,s}\boldsymbol{\epsilon}_s^{(j)}$
      Draw $\boldsymbol{\xi}_s^{(j)} \sim \mathcal{N}(0, \mathbf{I}_d)$ if $s > 0$
      $\mathbf{x}_s^{(j)} = \begin{cases} \mathbf{m} \odot \left(\sqrt{\bar{\alpha}_s}\mathbf{x}_{i,\text{obs}}^{\text{imp}} + \sqrt{1-\bar{\alpha}_s}\boldsymbol{\xi}_s^{(j)}\right) + (\mathbf{1}-\mathbf{m}) \odot \widetilde{\mathbf{x}}_s^{(j)}, & s > 0, \\ \mathbf{x}_{i,\text{obs}}^{\text{imp}} + (\mathbf{1}-\mathbf{m}) \odot \widetilde{\mathbf{x}}_0^{(j)}, & s = 0 \end{cases}$
    end for
  end for
  $\bar{x}_{0,p} = \text{median}\left\{x_{0,p}^{(1)}, \ldots, x_{0,p}^{(N_{\text{gen}})}\right\}, \quad p = 1, \ldots, d$
  $\bar{\mathbf{x}}_0 = (\bar{x}_{0,1}, \ldots, \bar{x}_{0,d})^\top$
  $\widehat{\mathbf{x}}_{i,\text{mis}}^{\text{imp}} = \bar{\mathbf{x}}_0$
end for
$\widehat{\mathbf{X}}^{\text{imp}} = \left[\widehat{\mathbf{x}}_{1,\text{mis}}^{\text{imp}}, \ldots, \widehat{\mathbf{x}}_{N_{\text{imp}},\text{mis}}^{\text{imp}}\right]^\top \in \mathbb{R}^{N_{\text{imp}} \times d}$
return $\widehat{\mathbf{X}}^{\text{imp}}$

the training loss still uses the original mask $\mathbf{M}_b$. Therefore, an observed value whose conditioning mask bit is hidden still contributes to the training loss. Originally missing positions remain zero and do not contribute to the loss.

For each graph support $A(\mathcal{S})$, we train a separate network. Let

$$\Omega_b = \{(i,j) : (\mathbf{M}_b)_{ij} = 1\}$$

denote the observed positions in the minibatch. For $|\Omega_b| > 0$, the training loss is

$$\mathcal{L}_b = \frac{1}{|\Omega_b|} \sum_{i=1}^{B} \frac{\bar{\alpha}_{t_i}}{1-\bar{\alpha}_{t_i}} \left\| (\mathbf{M}_b)_{i,:}^{\mathsf{T}} \odot \left(\widehat{\mathbf{x}}_{i,0,A(\mathcal{S})}^{(t_i)} - (\mathbf{X}_{\text{obs},b})_{i,:}^{\mathsf{T}}\right)\right\|_2^2. \tag{14}$$

Here, $\widehat{\mathbf{x}}_{i,0,A(\mathcal{S})}^{(t_i)}$ is the clean-value prediction for row i. For each row, the squared prediction error at observed positions is weighted by

$$\text{SNR}_{t_i} = \frac{\bar{\alpha}_{t_i}}{1-\bar{\alpha}_{t_i}}.$$

By Equation (5), this SNR-weighted clean-value loss is equal to the squared error of the noise estimate at observed positions. Thus, Equation (14) is the minibatch implementation of the observed-value objective in Equation (7). Each row is processed independently, and graph attention exchanges information only across variables within the same row.

Clipping values. In the reported experiments, we clip the noise estimate using the bounds in Equation (8). These bounds are derived from the fixed interval $[c_{\min}, c_{\max}] = [-5, 5]$ for the predicted clean values in standardized units.

Let $\mathbf{z}_0^{(t)}$ denote the clean-value prediction before clipping. For $0 < \bar{\alpha}_t < 1$, we first clip this prediction and then recompute the corresponding noise estimate:

$$\begin{aligned}
\mathbf{z}_0^{(t)} &= \frac{\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\,\widehat{\boldsymbol{\epsilon}}_{\theta^\star_{A(\mathcal{S})}}(\theta^\star_{A(\mathcal{S})}, t, \mathbf{x}_t; A(\mathcal{S}))}{\sqrt{\bar{\alpha}_t}}, \\
\widehat{\mathbf{x}}_0^{(t)} &= \min\Big\{c_{\max}\mathbf{1},\, \max\Big\{c_{\min}\mathbf{1},\, \mathbf{z}_0^{(t)}\Big\}\Big\}, \\
\widetilde{\boldsymbol{\epsilon}}_{\theta^\star_{A(\mathcal{S})}}(\theta^\star_{A(\mathcal{S})}, t, \mathbf{x}_t; A(\mathcal{S})) &= \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\,\widehat{\mathbf{x}}_0^{(t)}}{\sqrt{1-\bar{\alpha}_t}},
\end{aligned} \tag{15}$$

where $\mathbf{1} \in \mathbb{R}^d$ denotes the all-ones vector, and all minima and maxima are taken elementwise. For fixed $\mathbf{x}_t$ and t, the noise estimate decreases as the clean-value prediction increases. Therefore, the upper clean-value bound gives the lower noise bound, while the lower clean-value bound gives the upper noise bound:

$$\begin{aligned}
l(\mathbf{x}_t, t) &= \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\, c_{\max}\mathbf{1}}{\sqrt{1-\bar{\alpha}_t}} = \frac{\mathbf{x}_t - 5\sqrt{\bar{\alpha}_t}\,\mathbf{1}}{\sqrt{1-\bar{\alpha}_t}}, \\
u(\mathbf{x}_t, t) &= \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\, c_{\min}\mathbf{1}}{\sqrt{1-\bar{\alpha}_t}} = \frac{\mathbf{x}_t + 5\sqrt{\bar{\alpha}_t}\,\mathbf{1}}{\sqrt{1-\bar{\alpha}_t}}.
\end{aligned} \tag{16}$$

Thus, $l(\mathbf{x}_t, t), u(\mathbf{x}_t, t) \in \mathbb{R}^d$ give elementwise lower and upper bounds for the noise estimate. With the argument order used in Equation (8),

$$\mathrm{clip}(\mathbf{a}, \mathbf{z}, \mathbf{b}) = \min\{\mathbf{b}, \max\{\mathbf{a}, \mathbf{z}\}\}.$$

Substituting the bounds above into Equation (8) gives the clipped noise estimate in Equation (15).

Noise schedule and reverse updates. For each graph support, we optimize its network using AdamW and clip the gradient norm at 1. Following Section 4.1, we use $T = 1000$ diffusion steps, with $\alpha_t = 1 - \beta_t$, $\bar{\alpha}_t = \prod_{v=1}^{t} \alpha_v$, and $\bar{\alpha}_0 = 1$. The cosine noise schedule is

$$f(t) = \cos^2\left[\frac{\pi}{2}\frac{t/T + 0.008}{1.008}\right], \qquad \beta_t = \mathrm{clip}\left(1 - \frac{f(t)}{f(t-1)}, 10^{-4}, 0.999\right), \quad t = 1, \ldots, T.$$

For a reverse update from timestep t to s, where $t > s \geq 0$, we use the generalized DDIM sampling rule of Song et al. (2021) with

$$\sigma_{t,s} = \eta\sqrt{\frac{1-\bar{\alpha}_s}{1-\bar{\alpha}_t}\left(1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_s}\right)}, \qquad \eta = 1. \tag{17}$$

For $s > 0$, the reverse update includes transition noise $\sigma_{t,s}\boldsymbol{\epsilon}_s$, where $\boldsymbol{\epsilon}_s \sim \mathcal{N}(0, \mathbf{I}_d)$, as in Equation (9). For $s = 0$, $\bar{\alpha}_0 = 1$ implies $\sigma_{t,0} = 0$, and the reverse update reduces to the clean-value prediction $\widetilde{\mathbf{x}}_0 = \widehat{\mathbf{x}}_0^{(t)}$, as shown in Equation (10). Thus, $\eta = 1$ gives stochastic sampling, whereas $\eta = 0$ gives deterministic DDIM sampling.

Selected reverse timesteps. For a requested reverse-step setting $S_{\mathrm{rev}} \in \{1, \ldots, T\}$, let

$$h_{\mathrm{rev}} = \left\lfloor \frac{T}{S_{\mathrm{rev}}} \right\rfloor, \qquad J = \left\lceil \frac{T}{h_{\mathrm{rev}}} \right\rceil,$$

where $\lfloor x \rfloor$ and $\lceil x \rceil$ denote the floor and ceiling of x, respectively. We use the reverse timesteps

$$\tau_j = (J - 1 - j)h_{\mathrm{rev}} + 1, \qquad j = 0, \ldots, J-2,$$

together with the terminal timestep $\tau_{J-1} = 0$. Thus, the reverse process proceeds through $\tau_0 > \cdots > \tau_{J-1} = 0$. With $T = 1000$, the settings $S_{\mathrm{rev}} = 20, 30$, and 50 give $J = 20, 31$, and 50 network evaluations, respectively. We use $S_{\mathrm{rev}} = 50$ in the final experiments.

C.2 KAN Input Projection: From a Value–Mask Pair to Features

The KAN input projection maps the current value and conditioning mask bit of each variable to a d_m-dimensional feature vector. Each value–mask pair is processed independently before the resulting features are passed to graph-masked attention, where information is exchanged across variables according to the graph support.

Input: value–mask pair. Let R denote the number of rows processed together, with $R = B$ during minibatch training and $R = N_{\text{imp}}$ during imputation. Let $\mathbf{X_t} \in \mathbb{R}^{R\times d}$ contain the noisy rows, where $\mathbf{t} = (t_1, \ldots, t_R)$. For row i and variable j, the input pair is

$$\mathbf{u}_{i,j} = \big((\mathbf{X_t})_{i,j}, (\widetilde{\mathbf{M}})_{i,j}\big)^{\mathsf{T}} \in \mathbb{R}^2, \qquad U_{\mathbf{t}} \in \mathbb{R}^{R\times d\times 2}.$$

Thus, the two input channels ($c_{\text{in}} = 2$) are the current value and its conditioning mask bit. We use $\widetilde{\mathbf{M}} = \mathbf{M}_b^{\text{cond}}$ during training and $\widetilde{\mathbf{M}} = \mathbf{M}^{\text{imp}}$ during imputation. For example, $(0.7, 1)$ represents a current value of 0.7 whose conditioning mask bit indicates that it is observed. During training, an originally missing position has input $(0, 0)$ because missing values are set to zero. During imputation, the first channel at a missing position contains its current value from the reverse process, while the conditioning mask bit remains zero.

Step 1: normalize and clip the input. For each pair $\mathbf{u}$, we first apply LayerNorm (Ba et al., 2016) across its two channels and then clip the normalized values:

$$\bar{\mathbf{u}} = \text{clip}\left(\boldsymbol{\gamma}_{\text{LN}} \odot \frac{\mathbf{u} - \mu\mathbf{1}_2}{\sqrt{v + 10^{-5}}} + \boldsymbol{\beta}_{\text{LN}}, -2.99, 2.99\right),$$
$$\mu = (u_1 + u_2)/2, \qquad v = \big[(u_1 - \mu)^2 + (u_2 - \mu)^2\big]/2.$$

Here, $\mathbf{1}_2 = (1, 1)^{\mathsf{T}}$, and $\boldsymbol{\gamma}_{\text{LN}}$ and $\boldsymbol{\beta}_{\text{LN}}$ are learned gain and shift parameters. LayerNorm is applied separately to each value–mask pair and therefore does not use information from other rows or variables. We clip the normalized inputs to $[-2.99, 2.99]$ so that they remain well within the fixed spline grid rather than approaching its boundaries. This operation is different from clipping the clean-value prediction to $[-5, 5]$ during the reverse process.

Step 2: compute the KAN features. For output feature $h \in \{1, \ldots, d_m\}$, the KAN projection combines a base response and a spline response:

$$\big[\Phi_{\theta_{A(\mathcal{S})}}(U_{\mathbf{t}})\big]_{i,j,h} = \underbrace{\sum_{q=1}^{2} (W_{\text{base}})_{h,q}\, \text{SiLU}(\bar{u}_{i,j,q})}_{\text{base response}} + \underbrace{\sum_{q=1}^{2}\sum_{a=0}^{12} (W_{\text{spline}})_{h,q,a} \mathcal{B}_{q,a}^{(3)}(\bar{u}_{i,j,q})}_{\text{spline response}}.$$

The base response applies $\text{SiLU}(z) = z/(1+e^{-z})$ to each normalized input and combines the results using learned weights. The spline response evaluates 13 cubic B-spline basis functions for each input channel and combines them using learned spline weights. Each B-spline basis function is nonzero only within a limited interval and is zero outside that interval. Together, these two components provide a flexible nonlinear mapping from each value–mask pair to its feature representation.

The weight arrays have shapes $W_{\text{base}} \in \mathbb{R}^{d_m\times 2}$ and $W_{\text{spline}} \in \mathbb{R}^{d_m\times 2\times 13}$. Both input channels use the 17 fixed knots $\kappa_{q,a} = -4.8 + 0.6a$, $a = 0, \ldots, 16$. The basis functions $\mathcal{B}_{q,a}^{(3)}$ are cubic B-splines constructed by the Cox–de Boor recursion from the degree-zero interval indicators $\mathbb{1}[\kappa_{q,a} \le z < \kappa_{q,a+1}]$. The B-spline knot locations are fixed throughout training, while the base weights, spline weights, and LayerNorm parameters are learned.

Output: one feature vector per variable. The KAN output is

$$\Phi_{\theta_{A(\mathcal{S})}}(U_{\mathbf{t}}) \in \mathbb{R}^{R\times d\times d_m},$$

giving one d_m-dimensional feature vector for each variable in each row. For efficient computation, all Rd value–mask pairs are processed together and then reshaped back to their original row and variable positions. This reshaping only changes the tensor layout and does not exchange information across variables or rows. The resulting feature vectors are then passed to graph-masked attention, where variables exchange information according to the graph support.

C.3 Graph Attention and Conversion to a Noise Estimate

Add variable and time information. Each feature vector is combined with a learned variable vector from $C_{A(\mathcal{S})} \in \mathbb{R}^{1\times d\times d_m}$ and a time representation $\tau_{\theta_{A(\mathcal{S})}}(t_i)$. The variable vector identifies

the corresponding variable, while the time representation encodes the current diffusion timestep. Let $h_t = d_m/2$ and $\nu_q = 10000^{-q/(h_t-1)}$ for $q = 0, \dots, h_t - 1$. For $t_i \in \{1, \dots, T\}$, we set $k_i = t_i - 1$ and form the sinusoidal representation by concatenating $(\sin(k_i\nu_q))_q$ and $(\cos(k_i\nu_q))_q$. This vector is passed through an affine map from d_m to $2d_m$, followed by SiLU and a second affine map from $2d_m$ to d_m. The resulting time vector is added to all variable features in the same row. Here, $\tau_{\theta_{A(\mathcal{S})}}(\cdot)$ denotes the time representation, while τ_j denotes a selected reverse timestep.

Exchange information according to the graph support. For a selected graph support, let $A = A(\mathcal{S})$. For one row, let $H \in \mathbb{R}^{d\times d_m}$ contain the current feature vectors. The graph support determines which source variables are allowed to provide information to each target variable. Each attention layer first computes the query, key, and value matrices:

$$Q = \mathrm{Aff}_{W_Q,\mathbf{b}_Q}(H), \qquad K_{\mathrm{att}} = \mathrm{Aff}_{W_K,\mathbf{b}_K}(H), \qquad V = \mathrm{Aff}_{W_V,\mathbf{b}_V}(H),$$

where

$$\mathrm{Aff}_{W,\mathbf{b}}(H) = HW^\top + \mathbf{b}^\mathsf{T}.$$

The attention-score matrix is

$$S = \frac{QK_{\mathrm{att}}^\top}{\sqrt{d_m}}.$$

For target variable j, only source variables in $\mathcal{N}_A(j) = \{k : A_{jk} = 1\}$ receive nonzero attention weights:

$$\mathrm{msoftmax}_A(S)_{j,k} = \mathbb{1}[k \in \mathcal{N}_A(j)]\frac{\exp(S_{j,k})}{\sum_{u\in\mathcal{N}_A(j)}\exp(S_{j,u})}.$$

The attention weights over the allowed source variables sum to one. Because the graph support always includes the target variable itself, every target has at least one allowed source. In the implementation, this masked softmax is computed by applying the row-wise softmax

$$\mathrm{softmax}(B)_{j,k} = \frac{\exp(B_{j,k})}{\sum_{u=1}^{d}\exp(B_{j,u})}$$

to

$$B = S - 10^9(\mathbf{1}_{d\times d} - A),$$

where $\mathbf{1}_{d\times d}$ denotes the $d \times d$ all-ones matrix. The weighted messages are then added to the current features through a residual connection:

$$H' = H + \mathrm{Aff}_{W_O,\mathbf{b}_O}\left(\mathrm{msoftmax}_A(S)V\right).$$

All four affine maps have width d_m and use separate parameters in each layer. Thus, each layer applies single-head scaled dot-product attention (Vaswani et al., 2017), while the graph support restricts information exchange to the allowed source variables. Repeating the layer L times allows information to propagate through up to L successive connections in the graph support. Here, K_{att} denotes the key matrix.

Convert the clean-value prediction to a noise estimate. The final affine map $(\mathbf{w}_{A(\mathcal{S})}, b_{A(\mathcal{S})})$ produces one clean-value prediction for each variable. Let the resulting prediction for one row be $\widehat{\mathbf{x}}^{(t)}_{0,A(\mathcal{S})}$. The corresponding noise estimate used in the main text is

$$\widehat{\boldsymbol{\epsilon}}_{\theta_{A(\mathcal{S})}}(\theta_{A(\mathcal{S})}, t, \mathbf{x}_t; A(\mathcal{S})) = \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\,\widehat{\mathbf{x}}^{(t)}_{0,A(\mathcal{S})}}{\sqrt{1-\bar{\alpha}_t}}.$$

The implementation minimizes the SNR-weighted squared error of the clean-value prediction at observed positions. By the relation above, this is equivalent to minimizing the squared error of the corresponding noise estimate at those positions, as in the objective used in the main text.

Parameter initialization. Each graph support uses a separate network with its own parameters. The KAN base weights use Kaiming uniform initialization (He et al., 2015) with $a = \sqrt{5}$, which gives $\mathcal{U}[-1/\sqrt{c_{\mathrm{in}}},\ 1/\sqrt{c_{\mathrm{in}}}]$, where $\mathcal{U}(a, b)$ denotes the continuous uniform distribution on $[a, b]$. The spline weights are independently initialized from $\mathcal{N}(0, 0.1^2)$. The input LayerNorm is initialized with gain $\mathbf{1}_2$ and shift $\mathbf{0}_2$, and the variable vectors $C_{A(\mathcal{S})}$ are initialized from $\mathcal{N}(0, 1)$. For each affine layer in the time network, attention layers, and output layer, the weights and biases are initialized from $\mathcal{U}[-1/\sqrt{f_{\mathrm{in}}},\ 1/\sqrt{f_{\mathrm{in}}}]$, where f_{in} is the input width. The run seed fixes these initializations for reproducibility. The spline grid remains fixed during training.

C.4 SYNTHETIC SCM CONSTRUCTION

We generate the synthetic datasets using the IID simulator in gCastle (Zhang et al., 2021), whose linear and nonlinear simulation routines build on the NOTEARS simulation code (Zheng et al., 2018). For each preselected initial-DAG seed, the simulator generates an Erdős–Rényi DAG, that is, a random DAG obtained by randomly sampling edges under an acyclic node ordering, and assigns signed edge weights with magnitudes in $[0.5, 2]$. The variables are then generated in topological order with independent Gaussian noise $\varepsilon_j^{\mathrm{SCM}} \sim \mathcal{N}(0,1)$. Using the complete-data variables $X_j^\star$ and parent sets $\mathrm{Pa}(j)$ defined in Section 3.2, we consider four structural-equation families for variables with parents:

$$\begin{aligned}
\text{Linear Gaussian:} \quad & X_j^\star = \sum_{k\in\mathrm{Pa}(j)} W_{k,j} X_k^\star + \varepsilon_j^{\mathrm{SCM}}, \\
\text{Nonlinear MLP:} \quad & X_j^\star = \sum_{h=1}^{100} v_{j,h}\ \mathrm{sigmoid}\left(\sum_{k\in\mathrm{Pa}(j)} u_{j,k,h} X_k^\star\right) + \varepsilon_j^{\mathrm{SCM}}, \\
\text{Nonlinear MIM:} \quad & X_j^\star = \tanh(a_{j,1}) + \cos(a_{j,2}) + \sin(a_{j,3}) + \varepsilon_j^{\mathrm{SCM}}, \\
\text{Nonlinear quadratic:} \quad & X_j^\star = \frac{1}{\max(1, |\mathcal{T}_j|)} \sum_{g\in\mathcal{T}_j} c_{j,g}\, g\big((X_k^\star)_{k\in\mathrm{Pa}(j)}\big) + \varepsilon_j^{\mathrm{SCM}},
\end{aligned} \tag{18}$$

Here, in the linear Gaussian family, $W_{k,j}$ is the edge weight assigned to $X_k^\star \to X_j^\star$ as described above. For the nonlinear MLP family, $u_{j,k,h}$ and $v_{j,h}$ are the input-to-hidden and hidden-to-output coefficients, respectively, and are independently generated using the same random-sign and $[0.5, 2]$ magnitude rule. For the nonlinear MIM family, we define

$$a_{j,q} = \sum_{k\in\mathrm{Pa}(j)} w_{j,q,k} X_k^\star, \qquad q \in \{1,2,3\},$$

where the coefficients $w_{j,q,k}$ are independently generated using the same rule. The construction of $\mathcal{T}_j$ and the coefficients $c_{j,g}$ for the nonlinear quadratic family is described below. Variables with no parents are generated directly from $\varepsilon_j^{\mathrm{SCM}} \sim \mathcal{N}(0,1)$, whereas variables with at least one parent are generated according to Equation (18).

Terms retained in the quadratic family. The set $\mathcal{T}_j$ contains the linear, squared, and pairwise-product terms retained for the parents of $X_j^\star$. If a variable has a single parent, both its linear and squared terms are included, with their coefficients independently assigned random signs. If a variable has multiple parents, each candidate coefficient is set to zero with probability $1/2$; otherwise, its sign is sampled at random and its magnitude is drawn from $[0.5, 1]$. In this case, only terms with positive sampled coefficients are retained. If no term is retained, the structural contribution is zero and the variable is generated only from its Gaussian noise.

C.5 DATASET AND GRAPH-PROVENANCE SUMMARY

Table 3 summarizes the datasets and the graph used in CG-DIFF.

Table 3: Summary of the datasets and graphs used in CG-DIFF. "Controlled" means that missing positions are created from complete data, with their true values used only for evaluation. For empirical datasets, the supplied graph is treated as a working prior rather than as the ground-truth causal graph.

Dataset	N	d	Edges	Evaluation	Graph status
Four synthetic SCM families	1000	10	instance-specific	Controlled	Generator DAG; in the quadratic family, the recorded graph may contain edges that are not used in the final structural equations.
ADNI repository snapshot	414	8	9	Controlled	Literature-supported biomarker working DAG; constructed from prior evidence and not treated as the ground-truth causal graph (Guffanti et al., 2013).

Table 4 summarizes the literature evidence used to construct the graph of ADNI.

Table 4: Literature evidence used to define the nine directed edges in the ADNI working DAG. The directions are based on genetic ordering, biomarker-stage models, longitudinal evidence, and reported associations. The graph is used as a working prior; the cited studies do not establish every edge as a known causal direction.

Working edge(s)	Evidence for the working orientation	Source
AGE → Hippocampus	Age is associated with hippocampal volume in a multicohort study that includes ADNI.	Ferreira et al. (2017)
AGE → WholeBrain; WholeBrain → ADAS13	Longitudinal ADNI MRI data show age-related differences in brain atrophy and an association between brain atrophy and clinical progression.	Hua et al. (2010)
APOE4 → ABETA	APOE4 genotype is fixed before biomarker measurement and is associated with CSF amyloid levels.	Peskind et al. (2006)
ABETA → PTAU	The working direction follows an amyloid-first biomarker-stage model, in which amyloid abnormalities occur before later disease biomarkers.	Jack & Holtzman (2013)
PTAU → Hippocampus	Baseline CSF p-tau predicts subsequent hippocampal atrophy.	Henneman et al. (2009)
Hippocampus → MMSE; Hippocampus → ADAS13	ADNI studies associate hippocampal atrophy with cognition and subsequent clinical decline.	Risacher et al. (2017)
WholeBrain → MMSE	Longitudinal whole-brain atrophy is associated with decline in MMSE.	Sluimer et al. (2008)

C.6 CPU RUNTIME

Table 5 reports CPU runtime for nine baseline methods and two CG-DIFF models: the MB and Pa models. For the runtime comparison, we evaluate at the missingness rate of $\rho = 0.3$. Each method uses the same observation mask and its selected hyperparameters, giving 132 timing records in total.

We retain the 108 previously measured baseline timings and add 24 new measurements for the two CG-DIFF models. The baseline and CG-DIFF measurements were collected on different physical nodes, both equipped with an Intel Xeon Gold 6132 CPU at 2.60 GHz. All runs use two CPU cores and at most two threads per numerical library, without GPUs.

Runtime includes model construction, training, and imputation, but excludes package imports, standardization, hyperparameter selection, and evaluation. For CG-DIFF, imputation includes ten reverse trajectories and their elementwise median. We report the mean and sample SD over 15 initial-DAG seeds.

Table 5: CPU time in seconds for model construction, fitting and imputation; mean ± sample SD over seeds 42–44 on one representative instance per family at $\rho = 0.3$. The 96 baseline measurements are retained from node a116; the 24 CG-DIFF measurements were collected on node a131. Both nodes use Intel Xeon Gold 6132 CPUs at 2.60 GHz with two CPU cores and at most two threads per numerical library. CG-DIFF time includes ten reverse trajectories and median aggregation. Imports, standardization, tuning and scoring are excluded. GRIMP is not included in this timing comparison.

Method / variant	Linear Gaussian	Nonlinear MIM	Nonlinear MLP	Nonlinear Quadratic
CG-DIFF ($A(\mathrm{MB})$)	73.955±8.645	40.509±1.360	39.151±0.277	40.388±1.173
CG-DIFF ($A(\mathrm{Pa})$)	66.360±2.077	65.518±2.270	39.705±0.437	39.880±0.945
Mean	0.003±0.000	0.003±0.000	0.003±0.000	0.003±0.000
MICE	0.416±0.003	0.340±0.225	0.660±0.005	0.216±0.002
GAIN	3.906±0.012	4.474±0.717	3.958±0.010	3.927±0.013
MIWAE	51.797±1.238	30.811±0.096	20.999±0.323	25.898±9.094
MIRACLE	41.058±0.324	42.872±2.517	40.849±0.257	41.070±0.228
OT-Imputer	29.566±0.685	27.906±0.714	66.744±0.129	65.273±1.649
NewImp	481.776±9.276	485.425±8.016	431.638±0.715	394.253±46.016
TabDDPM	8.334±0.710	10.569±0.173	15.576±0.105	8.130±0.160

D PAIRED COMPONENT-ABLATION ANALYSIS

Table 6 compares the Markov-blanket-based CG-DIFF model with the parent-based model and five ablation variants. All comparisons use the same datasets, mask seed and run seeds.

- **w/o KAN** replaces the KAN input projection with a linear projection, removing the spline response, SiLU response, input LayerNorm, and input clipping.

- **w/o mask channel** sets the conditioning mask channel to zero while retaining the two-channel KAN input module.
- **w/o graph attention** removes all graph-attention layers and therefore prevents information exchange across variables through the graph support.
- **Final-only restoration** uses the same trained model and paired sampling noise as the full model, but restores the observed values only at the final sampling step.

For each instance and missingness rate, we first average the results over the run seeds. The paired differences are then averaged across the missingness rates and the four families sharing the same initial-DAG seed, giving 25 paired blocks. The Markov-blanket-based model has lower average RMSE than the parent-based model in 22 of the 25 blocks and lower average RMSE than each of the five ablation variants in all 25 blocks. These comparisons reflect the full changes defined above; in particular, removing KAN or graph attention changes multiple operations and may also change model capacity.

Table 6: Independently tuned component ablations over 25 graph-seed blocks; each block averages four families. Difference is variant minus MB. MB wins counts blocks with lower MB RMSE.

Variant	Mean RMSE $\pm$ SD	Difference	MB wins
CG-DIFF ($A(\mathrm{MB})$)	**1.350±0.271**	–	–
CG-DIFF ($A(\mathrm{Pa})$)	1.374±0.279	+0.0236	18/25
w/o diffusion	1.483±0.278	+0.1322	24/25
w/o KAN	2.144±0.268	+0.7931	25/25
w/o mask channel	2.239±0.272	+0.8888	25/25
w/o graph attention	1.389±0.293	+0.0390	21/25

The parent-based model uses $A(\mathrm{Pa})$ instead of $A(\mathrm{MB})$ and is tuned separately. Its results are also reported in the main comparison tables.

These controls use all synthetic datasets and the controlled ADNI and benchmarks, with the same missingness rates, mask seed, and run seeds. They examine the contribution of diffusion and the choice of nonlinear input projection, respectively.

Graph predictor without diffusion. This control retains the KAN input projection, variable embeddings, Markov-blanket-plus-self attention, and output layer, while removing the time-embedding network and the diffusion process. It receives standardized observed values with zeros at missing positions and minimizes reconstruction mean squared error over observed values only. The original mask-channel dropout is retained: each observed conditioning bit is set to zero with probability 0.2, while the input values and the loss mask remain unchanged. No diffusion noise or timestep-dependent loss weight is used. At imputation time, the model is evaluated once; its predictions are clipped to $[-5, 5]$ in standardized units, and the observed values are restored exactly.

We select its hyperparameters independently using the same nine candidate configurations, validation masks, 30 training epochs, and tuning seed 42 as CG-DIFF. Each candidate is evaluated by a single forward pass. This comparison evaluates the specified direct reconstruction model against CG-DIFF.

MLP input projection with comparable parameter count. To compare KAN with an alternative nonlinear projection, we replace its base and spline responses with a two-layer MLP using a SiLU activation. We retain the input LayerNorm and clipping to $[-2.99, 2.99]$. The MLP hidden width is chosen to minimize the difference in projection parameter count. For embedding widths $d_m = 32$, 64, and 128, the hidden widths are 25, 26, and 26, respectively; the resulting total network parameter count differs from the corresponding KAN model by less than 0.2%. The graph support, attention layers, time-embedding network, training objective, and reverse sampling procedure remain unchanged. Each dataset uses the hyperparameters selected for the corresponding Markov-blanket-based CG-DIFF model.