RHYTHM: SYNCHRONISED CORRUPTIONS FOR CONTINUOUS TABULAR DIFFUSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Mainstream tabular diffusion methods use mixed-noise formulations that separately handle discrete and continuous features. While effective at capturing overall distribution trends, these methods have limited control over constrained data generation. We propose RHYTHM, a new tabular diffusion paradigm built entirely on a continuous-noise formulation, enabling theory-grounded conditional generation via training-free guidance. RHYTHM incorporates a key property of mixed-noise models: when noise is added, different features deviate from their ground-truths at synchronised rates, making all features equally informative. We theoretically characterise this property as *corruption alignment* and incorporate it into RHYTHM by computing feature-wise noise schedules, via an offline, training-free algorithm. Results on diverse datasets show that RHYTHM matches existing models on unconditional generation and significantly outperforms mixed-noise methods under conditional generation, highlighting the benefits of continuous diffusion for high-quality and controllable generation. Code: `https://anonymous.4open.science/r/BestDiffusion-3B12/README.md`.

1 INTRODUCTION

Tabular data consists of heterogeneous continuous and categorical features, which makes it challenging for generative modelling since different columns have different statistical properties, scales, and meanings (Zhao et al., 2021). Among existing generative methods, *diffusion* models have emerged as state-of-the-art (Kotelnikov et al., 2023; Zhang et al., 2024; Shi et al., 2025; Shankar et al., 2026), outperforming Generative Adversarial Networks (Goodfellow et al., 2014), Variational Autoencoders (Kingma & Welling, 2014), and large language models. They generate data by reversing a stochastic corruption process defined by a *noise schedule*. Although originally developed for continuous data modalities (e.g., images) with purely continuous corruption mechanisms, mainstream tabular diffusion methods adopt *mixed-noise* schedules that separate continuous and discrete corruption processes to model different features (Kotelnikov et al., 2023; Shi et al., 2025).

Such mixed-noise methods can capture overall distribution trends well, but have limited use for *conditional* (a.k.a constrained) generation. Real-world applications need synthetic tables adhering to user-defined constraints at inference, like conditionally imputing missing table cells (Zhang et al., 2025) or enforcing feature-wise inequality rules (Shankar et al., 2026). Continuous-noise diffusion has theory-grounded ways to steer generation via *training-free guidance* (Shankar et al., 2026; Chung et al., 2022; Bansal et al., 2023). But mixed-noise models lack this capability, since their discrete noising is non-differentiable, blocking guidance (Gruver et al., 2023; Schiff et al., 2025).

We seek to combine the benefits of continuous diffusion for conditional generation and the strong empirical performance of mixed-noise diffusion. We posit that mixed diffusion performs well because its noising mechanism causes the ground-truth values of different features to degrade at synchronised rates. This keeps features equally informative at every step, simplifying the learning of joint feature distributions. We theoretically characterise this property as *corruption alignment*, and show that we can replicate the property in a continuous-noise model by simply *synchronising* the feature-wise noise schedules to "corrupt" the ground-truth values at equal rates.

We propose RHYTHM, a new tabular diffusion model built entirely on a continuous noise formulation. Its core is driven by the objective of *corruption alignment*, deriving feature-wise noise schedules by synchronising information loss across features. This captures overall distributional trends

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.

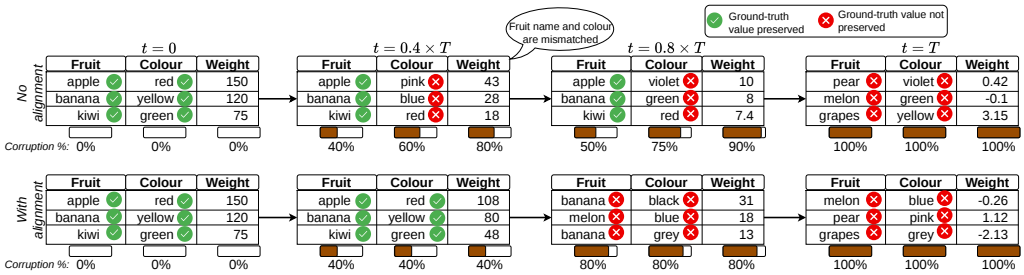


Figure 1: Unaligned continuous noise diffusion (top), degrades the ground-truth values of features at different rates, making joint distributions hard to learn. Corruption alignment (bottom) yields equally informative (or equally noisy) features at each step, making joint distributions easy to learn.

while enabling theory-grounded approaches for constrained generation via guidance. Moreover, the corruption-aligned schedules are *analytically* computable from metadata like feature cardinalities and distribution statistics, enabling our *bisection-based* algorithm to compute the schedules in a *training-free* and *offline* manner. In summary, we make the following contributions:

Model: We propose RHYTHM, a new, fully continuous tabular diffusion model to generate high-quality tabular data while enabling theoretically grounded guidance for controllable generation.

Theory: We formalise *corruption alignment*, a principle to retrieve feature-wise noise schedules analytically by aligning the information degradation rates across features.

Algorithm: We provide a practical realisation of our theoretical formulation through a *fast*, *training-free bisection* algorithm to compute feature-wise, corruption-aligned noise schedules.

We show that our continuous diffusion model with corruption-aligned schedules closes the gap to both mixed-noise methods and methods that learn feature-wise noise schedules during training. At the same time, its continuous formulation remains adaptive to inference-time constraints, significantly outperforming mixed-noise methods on conditional generation.

2 BACKGROUND AND RELATED WORK

Tabular Diffusion Models. These models generate data by reversing a gradual noising process. Given a clean tabular sample (a row) $x_0 \sim p(x_0)$, the *forward* process progressively adds noise via a Markov chain $p(x_{1:T}|x_0) = \prod_{t=1}^{T} p(x_t|x_{t-1})$, following a predefined noise schedule over T steps. Each sample consists of d_{cont} continuous and d_{cat} categorical features, denoted by $x_0 = [x_0^{\text{cont}(j\in[1..d_{\text{cont}}])}, x_0^{(i\in[1..d_{\text{cat}}])}]$. Existing models differ mainly in how the forward process is defined, giving rise to three main paradigms: *Gaussian*, *multinomial*, and *masked* diffusion (all mathematical notations are summarised in Appendix A).

(i) Gaussian Diffusion (Ho et al., 2020) methods add continuous Gaussian noise to both continuous and one-hot encoded categorical features. The forward process is given by:

$$p(x_t|x_0) = \mathcal{N}\left(\sqrt{\bar{\alpha}_t}\, x_0, (1-\bar{\alpha}_t)I\right); \quad x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon, \quad \epsilon \sim \mathcal{N}(0, I), \tag{1}$$

where $\{\bar{\alpha}_t\}_{t=0}^{T} \subset [0,1]$ is a monotonically decreasing *noise schedule* from 1 to 0, making x_T converge to isotropic Gaussian noise $\mathcal{N}(0, I)$. Tabular adaptations inject noise across all features directly over fixed or learned embeddings (Zhang et al., 2024; Mueller et al., 2025).

(ii) Multinomial Diffusion (Hoogeboom et al., 2021; Kotelnikov et al., 2023) is a mixed-noise paradigm that combines Gaussian noise for continuous features x_0^{cont} with multinomial noise for categorical ones. For a categorical feature $x_0^{(i)}$ with K_i categories, the forward process is

$$p(x_t^{(i)}|x_0^{(i)}) = \text{Cat}\left(\bar{\alpha}_t x_0^{(i)} + (1-\bar{\alpha}_t)\frac{\mathbf{1}}{K_i}\right), \tag{2}$$

where each $x_0^{(i\in[1..d_{\text{cat}}])}$, is gradually corrupted towards a uniform distribution over K_i categories.

(iii) Masked Diffusion (Shi et al., 2025; Austin et al., 2021) is also a mixed-noise paradigm that replaces multinomial transitions in (2) with an additional *mask* category. Continuous features follow the Gaussian noising process in (1), while categorical features evolve as

$$p(x_t^{(i)}|x_0^{(i)}) = \text{Cat}\Big(\bar{\alpha}_t x_0^{(i)} + (1-\bar{\alpha}_t)m\Big), \tag{3}$$

where $x_0^{(i)} \in \{e_1, \ldots, e_{K_i}\} \subseteq \{0,1\}^{K_i+1}$ is a one-hot vector with $m = e_{K_i+1}$ being the mask.

All three paradigms generate data by learning a parametrised *reverse* process $p_\theta(x_{t-1}|x_t)$ using a trained neural network, parametrised by θ. We provide the details on the reverse dynamics and training objectives for the aforementioned noise formulations in Appendix C.

Learned Noise Schedules. The noise schedule $\{\bar{\alpha}_t\}_{t=1}^T$ determines how rapidly information is removed. The standard approach is to apply a predefined schedule uniformly across all features (Ho et al., 2020; Nichol & Dhariwal, 2021). This allocates model capacity inefficiently across features, motivating feature-wise *learnable* schedules that are jointly optimised with the diffusion model (see Appendix D for details). *Time-warping* (Mueller et al., 2025; Dieleman et al., 2022) fits a function that redistributes training effort toward diffusion steps with a higher reconstruction loss. Parametric approaches (Shi et al., 2025) learn separate schedules for continuous and categorical features by restricting the function family. But both approaches learn schedules *indirectly*—they adapt the noising process to optimise proxy objectives, rather than explicitly characterising how information should degrade across features. We instead study schedule design through the lens of *corruption alignment*, and analytically derive feature-wise schedules by making features equally informative across diffusion steps, without training or loss-based optimisation.

3 RHYTHM

Figure 1 shows that applying a shared noise schedule across all features causes high-cardinality and continuous features (e.g. colour ($K = 20$)) to change from the ground-truth value faster than low-cardinality ones (e.g. fruit ($K = 6$)). Hence, some features retain their ground-truth values while others collapse into randomness, making it difficult to learn joint distributions.

RHYTHM overcomes this challenge by adapting feature-wise noise schedules to achieve *corruption alignment*, i.e., make all features equally informative at each diffusion step. Section 3.1 develops the theory behind this principle, first for categorical features and then generalises the notion to continuous features through their *effective cardinality*. Section 3.2 then describes a practical way to align noise schedules in an *offline* manner through a *fast*, *training-free* bisection algorithm (Sec. 3.2.1).

RHYTHM is applied as follows. We first align noise schedules across features as a precursor to model training. Using the given cardinalities (or estimates) for all features, we pre-compute a lookup table of feature-wise schedules. Each feature is then noised using its synchronised schedule for training and inference. If constraints are imposed at inference, we apply guidance mechanisms to our pre-trained, fully continuous diffusion backbone, as detailed in Appendix C.4.

3.1 THEORY OF CORRUPTION ALIGNMENT: FROM DISCRETE TO CONTINUOUS FEATURES

We start by introducing a general notion of identity degradation for *categorical* features. Consider a one-hot categorical feature $x_0^{(i)}$. Let $x_t^{(i)}$ be its noisy representation at step t. Pynadath et al. (2026) defines the *discrete identity corruption* (DIC) as the probability that a noised categorical feature is no longer the ground-truth value. It is defined as follows.

Definition 1 (Discrete Identity Corruption (DIC))**.** *The discrete identity corruption of the ith categorical feature $x_t^{(i)}$ at diffusion step t is given by*

$$\rho_i(t) = \mathbb{P}\left(\text{argmax}(x_t^{(i)}) \neq x_0^{(i)}\right),$$

Proposition 1 analytically derives the DICs for different noise mechanisms, proved in Appendix B.1.

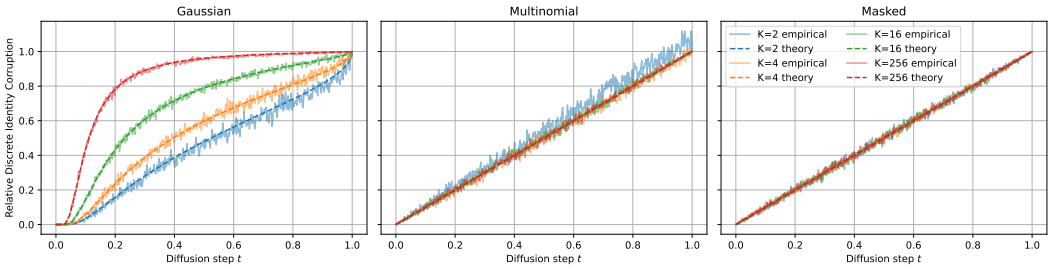


Figure 2: Theoretical (Prop. 1) and empirical RICs (averaged across 1000 samples) for different diffusion noise families and feature cardinalities. Diffusion step t is normalised over $T = 500$ steps.

Proposition 1. *For a categorical feature (indexed by i) with $K_i \geq 2$ categories and noise schedule $\{\bar{\alpha}_t\}_{t=0}^{T}$, the DIC monotonically increases with t and is given by*

$$\rho_i(t) = \begin{cases} \displaystyle\int_{-\infty}^{\infty} \left(1 - \left(\Phi\left(s + \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\right)\right)^{K_i - 1}\right) \mathcal{N}(s; 0, 1)\, ds, & \textit{Gaussian diffusion}, \quad (4a) \\ \left(1 - \dfrac{1}{K_i}\right)(1 - \bar{\alpha}_t), & \textit{Multinomial diffusion}, \quad (4b) \\ 1 - \bar{\alpha}_t, & \textit{Masked diffusion}. \quad (4c) \end{cases}$$

where $\Phi(\cdot)$ denotes the standard Gaussian cumulative distribution function (CDF).

Comparing raw DICs across tabular features having different cardinalities is meaningless. For example, at step T, a fully noisy *gender* feature (K=2) and *country* feature (K=100) should both correspond to random guessing, despite having different expected correct-guess rates of 50% and 1%, respectively. We normalise them by their respective maximums, making the resulting values have the same interpretation across features. We call this the *relative identity corruption* (RIC).

Definition 2 (Relative Identity Corruption (RIC))**.** *For a categorical feature (indexed by i) and a given noise schedule $\{\bar{\alpha}_t\}_{t=0}^{T}$, the relative identity corruption at step t is defined as*

$$\bar{\rho}_i(t) = \frac{\rho_i(t)}{\rho_i(T)}, \quad (5)$$

We then define two categorical features as being *corruption-aligned* if their RICs are the same.

Definition 3 (Corruption Alignment)**.** *Any two categorical features indexed by u and v are corruption-aligned under a diffusion process if*

$$\bar{\rho}_u(t) = \bar{\rho}_v(t), \qquad \forall t \in [0..T]. \quad (6)$$

Intuitively, corruption alignment means that each feature has the same proportion of its maximum achievable entropy at every diffusion step, regardless of its cardinality. Notably, Thm. 1 below shows that discrete-noise mechanisms induce RIC trajectories that are *cardinality-invariant*.

Theorem 1 (Inherent Corruption Alignment in Discrete Noise Paradigms)**.** *Let u and v be any two categorical features with distinct cardinalities $K_u, K_v \geq 2$. Under a shared noise schedule $\{\bar{\alpha}_t\}_{t=0}^{T}$ across features, Masked and Multinomial diffusion inherently satisfy corruption alignment:*

$$\bar{\rho}_u(t) = \bar{\rho}_v(t) = 1 - \bar{\alpha}_t, \qquad \forall t \in [0..T]. \quad (7)$$

Conversely, Gaussian diffusion is misaligned, i.e., $\exists\, t \in [0..T]$ such that $\bar{\rho}_u(t) \neq \bar{\rho}_v(t)$.

We provide the proof in Appendix B.2. We hypothesise that corruption alignment is a key reason *why* mixed diffusion performs well on tabular data. Figure 2 visualises the alignment established by Thm. 1. It shows the feature-wise RICs under a shared noise schedule, where $\bar{\alpha}_t$ linearly decreases

from 1 to 0 over the diffusion steps. Empirical trajectories closely match the theoretical results using (4). Discrete noise methods have RICs that are aligned across cardinalities while Gaussian diffusion has cardinality-dependent behaviour, with higher-cardinality features changing earlier.

Generalisation to Continuous Features. Our analysis so far assumes that each feature has a cardinality, which naturally exists for categorical features. But continuous features have no finite cardinality, so extending the analysis requires assuming (or estimating) an *effective cardinality* K_{eff} that approximates the number of distinguishable values in a feature. While one could hardcode a uniformly low (e.g., $K_{\text{eff}} = 2$) or high (e.g., $K_{\text{eff}} = 300$) cardinality for all continuous features, we try a more principled, data-dependent estimate by discretising each feature into bins using the Freedman–Diaconis rule (Freedman & Diaconis, 1981). This rule provides an outlier-robust bin width based on the empirical distribution. For the jth continuous feature column $x^{\text{cont}(j)}$, we estimate

$$K_{\text{eff}}^{(j)} = \left\lceil \frac{\max(x^{\text{cont}(j)}) - \min(x^{\text{cont}(j)})}{h_j} \right\rceil; \quad \text{where } h_j = 2 \times \frac{\text{IQR}(x^{\text{cont}(j)})}{m^{1/3}}, \tag{8}$$

where, IQR denotes the inter-quartile range of the feature using all m training samples. We then apply the same RIC analysis across both categorical and continuous features. This allows us to quantify information loss across both discrete and continuous features on an intuitive, probability-based scale. We discretise continuous features via effective cardinalities strictly as a theoretical convenience for the analysis and noise-schedule derivation. We do not discretise continuous features in practice; they remain continuous during both training and inference, preserving the inference-time guidance capabilities of continuous-noise models (Shankar et al., 2026; Scassola et al., 2025).

3.2 Deriving Schedules Achieving Corruption Alignment

Theorem 1 showed that discrete-noise paradigms are implicitly corruption-aligned while Gaussian diffusion is not. Could we synchronise the feature-wise Gaussian noise schedules so that their RIC trajectories match? We could then inherit the desirable properties mixed diffusion while retaining a continuous formulation. However, corruption alignment only requires feature-wise RICs to be equal; it does not prescribe what the shared trajectory should be. To make the criterion actionable, we choose a shared *reference* trajectory $\{\bar{\rho}_{\text{ref}}(t)\}_{t=0}^{T}$ for all features. We then define *corruption-aligned schedules* as feature-specific noise schedules whose RICs match this reference.

Definition 4 (Corruption-aligned Schedules)**.** *Consider a reference noise schedule $\{\bar{\alpha}_t^{(i)}\}_{t=0}^{T}$. A set of feature-specific noise schedules $\{\{\bar{\alpha}_t^{\prime(i)}\}_{t=0}^{T}\}_{i=0}^{d_{\text{cont}}+d_{\text{cat}}}$ satisfies corruption alignment if*

$$\bar{\rho}'_u(t) = \bar{\rho}'_v(t) = \bar{\rho}_{\text{ref}}(t), \quad \forall u, v, \ \forall t \in [0..T] \tag{9}$$

where $\rho'_d(t)$ is shorthand to denote the RIC induced by $\bar{\alpha}_t^{\prime(d)}$.

How to choose a good reference? In theory, it can be any monotonic schedule with the boundary conditions $\bar{\rho}_{\text{ref}}(\bar{\alpha}_0) = 0$ and $\bar{\rho}_{\text{ref}}(\bar{\alpha}_T) = 1$. But we adopt a *linearly increasing* schedule from 0 to 1 since it empirically yields a lower training loss across diffusion steps (see Appendix H.2).

3.2.1 Algorithm for Training-Free, Offline Corruption Alignment

To make RHYTHM's continuous noise schedules achieve corruption alignment according to Definition 4, we solve the non-linear inverse problem of finding the Gaussian noise level for each feature and diffusion step that matches a given RIC reference. Unlike prior adaptive schedule methods that rely on loss-based time-warping or network training (Shi et al., 2025; Mueller et al., 2025; Dieleman et al., 2022), ours is *training-free*, requiring no neural network updates, and can be executed *offline*.

All diffusion noise schedules share the same boundary conditions: $\bar{\alpha}_0 = 1$ (no noise) and $\bar{\alpha}_T = 0$ (pure noise). Following Sec. 3.2, we adopt a linearly increasing reference RIC trajectory $\{\bar{\rho}_{\text{ref}}(t)\}_{t=0}^{T}$ that grows steadily from 0 to 1. Using (9), we establish the absolute target DIC for the ith feature at step t by scaling the reference RIC with the feature's maximum corruption limit:

$$\rho_i^*(t) = \bar{\rho}_{\text{ref}}(t) \times \rho_i(T), \quad \forall t \in [1..T-1]. \tag{10}$$

Bisection-based corruption alignment. Since the Gaussian DIC equation (4a) does not admit a closed-form inverse, we exploit its *monotonicity* (see Prop. 1) to solve for $\bar{\alpha}_t'^{(i)}$ numerically. Since $\bar{\alpha}_t$ is monotonically decreasing, decreasing $\bar{\alpha}_t'$ increases the induced corruption $\rho_i'(t)$. Therefore, for any valid target level $\rho_i^*(t)$, there exists a unique signal level $\bar{\alpha}_t'^{(i)} \in [0,1]$ that satisfies (10). Algorithm 1 leverages this strict monotonicity to isolate $\bar{\alpha}_t'^{(i)}$ for each feature and timestep independently using bisection search. We initialise the search bounds to $[\bar{\alpha}'_{\text{low}}, \bar{\alpha}'_{\text{high}}] = [0,1]$. At each iteration, we evaluate the DIC at the midpoint (line 7). If the evaluated corruption exceeds the target $\rho_i^*(t)$, the midpoint becomes the new upper bound $\bar{\alpha}_{\text{high}}$. Otherwise, it becomes the lower bound $\bar{\alpha}_{\text{low}}$. We use N=20 iterations which yields an accurate estimate of $\bar{\alpha}_t'^{(i)}$ in practice.

DIC Computation. Algorithm 2 computes the Gaussian DIC required for schedule alignment. We approximate the integral in (4a) over a fixed grid of Q points $\{c_q\}_{q=1}^{Q}$ from [-8, 8] with standard Gaussian weights. Line 4 approximates this as the weighted trapezoidal sum across these Q grid points, and takes its complement.

Computational efficiency. The quadrature weight $w_q \Delta c$ in line 3 of Alg.2 is constant across all feature–timestep pairs and can therefore be precomputed and reused across all DIC evaluations. We further vectorise the computations across timesteps and quadrature points, reducing the runtime. Consequently, Alg. 1 scales approximately linearly with the number of diffusion steps and features for a fixed quadrature resolution and optimisation iterations.

Algorithm 1 Corruption alignment

1: **for** each feature i in $[1..(d_{\text{cont}} + d_{\text{cat}})]$ **do**
2: **for** each step t **do**
3: Set $[\bar{\alpha}'_{\text{low}}, \bar{\alpha}'_{\text{high}}] = [0,1]$
4: Set $\rho_i^*(t) = \bar{\rho}_{\text{ref}}(t) \times \rho_i(T)$ (10)
5: **for** iteration $n \in [1..N]$ **do**
6: $\bar{\alpha}'_t = (\bar{\alpha}'_{\text{low}} + \bar{\alpha}'_{\text{high}})/2$
7: Get $\rho_i'(t)$ using Alg. 2
8: **if** $\rho_i'(t) > \rho_i^*(t)$ **then**
9: $\bar{\alpha}'_{\text{high}} \leftarrow \bar{\alpha}'_t$
10: **else**
11: $\bar{\alpha}'_{\text{low}} \leftarrow \bar{\alpha}'_t$
12: Set schedule: $\bar{\alpha}'_t := \bar{\alpha}'_{\text{mid}}$
13: **return** $[\bar{\alpha}_t'^{(i)}]_{t,i} \in \mathbb{R}^{(T+1)\times(d_{\text{cont}}+d_{\text{cat}})}$

4 Experiments

We evaluate our claims from five angles: *(i) Generation quality:* whether a purely continuous diffusion model with corruption-aligned schedules can match or outperform existing approaches. *(ii) Adaptability:* whether the continuous backbone provides adaptability for inference-time constrained generation. *(iii) Efficiency:* whether the computational cost of the schedule alignment procedure is practical. *(iv) Ablations:* the effect of reference schedule choices and the cardinality estimation strategy . *(v) RIC Dynamics:* how analytical, corruption aligned RICs compare to empirical schedule-optimisation heuristics.

Algorithm 2 Gaussian DIC computation

Require: $\bar{\alpha}'_t$, cardinality K_i, quadrature points Q

1: Set grid points $\{c_q\}_{q=1}^{Q}$
2: Compute Gaussian weights:
3: $\underbrace{w_q = \mathcal{N}(c_q; 0, 1) \times \Delta_c, \quad \forall q \in [Q]}_{\text{independent of } t,i}$
4: Compute

$$\rho_i'(t) = 1 - \sum_{q=1}^{Q} \underbrace{\left(\Phi\left(c_q + \frac{\sqrt{\bar{\alpha}'_t}}{\sqrt{1-\bar{\alpha}'_t}}\right)\right)^{K_i - 1}}_{\text{depends on } t,\, i} w_q$$

5: **return** $\rho_i'(t)$

4.1 Settings

Datasets. We evaluate our approach on nine tabular datasets spanning a diverse range of domains and feature types: Adult, Bean, Churn, Default, Gesture, Intrusion, Letter, Magic, and Shoppers. Most datasets are obtained from the UCI repository (Dua et al., 2017), while Churn and Intrusion are obtained from Kaggle. We use 70-30 train-test splits. Categorical features are represented using one-hot encoding, while continuous features are Gaussian-standardised. Further details on the datasets and preprocessing are provided in Appendix E.

Baselines. We compare against representative tabular diffusion models spanning mixed and purely continuous diffusion paradigms: TabDiff (Shi et al., 2025) (Masked + Gaussian + learned schedules),

Table 1: Unconstrained sample quality on various metrics ($\text{Mean}_{\text{std.}}$), averaged across nine datasets.

Method	Accuracy (↑)	Column Pair Trends (↑)	Column Shapes (↑)	Coverage (↑)	F1 (↑)
CDCD	$0.792_{0.155}$	$0.774_{0.242}$	$0.869_{0.071}$	$\underline{0.940_{0.038}}$	$\underline{0.739_{0.175}}$
CDTD	$0.792_{0.152}$	$0.747_{0.276}$	$0.865_{0.073}$	$0.920_{0.081}$	$0.717_{0.212}$
Vanilla	$0.731_{0.184}$	$0.708_{0.284}$	$0.852_{0.070}$	$0.832_{0.203}$	$0.654_{0.245}$
TabDDPM	$0.738_{0.163}$	$\underline{0.809_{0.227}}$	$\underline{0.879_{0.051}}$	$0.889_{0.068}$	$0.679_{0.190}$
TabDiff	$\underline{0.792_{0.151}}$	$0.776_{0.224}$	$0.872_{0.069}$	$\mathbf{0.943_{0.038}}$	$0.736_{0.174}$
TabSyn	$0.757_{0.152}$	$0.786_{0.222}$	$0.861_{0.055}$	$0.865_{0.117}$	$0.697_{0.173}$
RHYTHM	$\mathbf{0.806_{0.129}}$	$\mathbf{0.830_{0.231}}$	$\mathbf{0.895_{0.055}}$	$0.871_{0.144}$	$\mathbf{0.753_{0.157}}$

TabDDPM (Kotelnikov et al., 2023) (Multinomial + Gaussian), TabSyn (Zhang et al., 2024) (Latent-space Gaussian), CDCD (Dieleman et al., 2022) and CDTD (Mueller et al., 2025) (Gaussian + learned schedules), and a Vanilla baseline (Gaussian, shared schedule across features). For a fair comparison, all methods use the same MLP backbone architecture using T=500 steps. By default we use a noise schedule, where the noise-to-signal ratio $\sqrt{\frac{1-\bar{\alpha}_t}{\bar{\alpha}_t}}$ increases linearly from 0 to an upper bound of 100 across all features. Additional details on model configurations and training hyper-parameters are provided in Appendix F.

Evaluation Protocol. We evaluate generation quality using complementary metrics that assess distributional resemblance (KSComplement, TVComplement, Trend, and correlation similarity), diversity (Coverage), and downstream utility (F1 score/accuracy). Constraint satisfaction is evaluated separately for conditional generation. All metrics are normalised to $[0, 1]$, with higher values indicating better performance, and are computed on a hold-out test set to ensure that synthetic samples are not simply memorised from the training data. Unless specified, all experiments report results over five trials. More details on the metrics and downstream task configurations are in Appendix G.

4.2 Generation Quality

We evaluate whether corruption alignment improves sample quality under unconditional generation. For compactness, Table 1 reports the aggregate scores across datasets, similar to Mueller et al. (2025). Dataset-wise results are included in Appendix H.4.1.

RHYTHM tops most metrics, competitively performing against mixed diffusion (TabDDPM, TabDiff) and learned-schedule approaches (CDCD, CDTD). While the sample coverage is not the highest, the absolute score is high, indicating that the diversity is still reasonably good. Latent-diffusion (TabSyn), while generally good, falls slightly short of mixed-diffusion and learned-schedule methods likely due to reconstruction errors between the latent and data spaces via the autoencoder.

Importantly, Vanilla scores lowest across all metrics, showing that a , continuous-noise model is insufficient for tabular data as is. Its shared, unmodified noise schedule fails to account for the heterogeneous corruption dynamics across features, motivating the need for feature-wise schedules.

4.3 Inference-Time Constrained Generation

To evaluate whether the different diffusion formulations under inference-time constraints, we follow the protocol of Shankar et al. (2026) which considers three types of constraints: *range* constraints on continuous features, *categorical* constraints on discrete ones, and their *conjunction* (see Appendix G.3). We apply training-free guidance (Chung et al., 2022; Bansal et al., 2023; Shankar et al., 2026) to steer towards the constraints. Since discrete-noise diffusion does not theoretically support such guidance mechanisms, we build a workaround that steers the pre-sampling output *logits* instead, to skew the class probabilities towards the constraint (see Appendices C.4 and F.4).

Range constraints are generally handled well by all methods (see Table 2). RHYTHM generally achieves the best or second-best quality scores with a constraint match rate above 95%. Notably, mixed-noise models, especially TabDiff, also adapt well, probably because range constraints primarily affect the continuous features of the noisy samples, for which steering gradients are permissible.

Categorical and conjunctive constraints expose a fundamental limitation of discrete-noise methods, since guidance can only steer the pre-sampling logits. This substantially limits constraint satisfac-

Table 2: Sample quality and constraint match rate, averaged across nine datasets.

Constraint	Method	Accuracy (↑)	Column Pair Trends (↑)	Column Shapes (↑)	Coverage (↑)	F1 (↑)	Match % (↑)
Range	CDCD	$0.776_{0.158}$	$0.749_{0.254}$	$0.840_{0.061}$	$0.940_{0.031}$	$0.699_{0.158}$	$0.970_{0.040}$
	CDTD	$0.785_{0.149}$	$0.722_{0.277}$	$0.835_{0.063}$	$0.910_{0.096}$	$0.678_{0.199}$	$0.939_{0.061}$
	Vanilla	$0.723_{0.182}$	$0.692_{0.297}$	$0.835_{0.072}$	$0.824_{0.165}$	$0.618_{0.223}$	$0.602_{0.166}$
	TabDDPM	$0.734_{0.153}$	$0.784_{0.247}$	$\mathbf{0.868_{0.058}}$	$0.849_{0.101}$	$0.645_{0.160}$	$0.685_{0.181}$
	TabDiff	$0.784_{0.146}$	$0.741_{0.246}$	$0.833_{0.068}$	$\mathbf{0.944_{0.053}}$	$0.700_{0.162}$	$\mathbf{0.992_{0.009}}$
	TabSyn	$0.758_{0.142}$	$0.766_{0.243}$	$0.836_{0.059}$	$0.858_{0.073}$	$0.666_{0.144}$	$0.909_{0.049}$
	RHYTHM	$\mathbf{0.800_{0.131}}$	$\mathbf{0.803_{0.243}}$	$0.864_{0.057}$	$0.888_{0.107}$	$\mathbf{0.725_{0.151}}$	$0.958_{0.047}$
Category	CDCD	$0.843_{0.146}$	$0.768_{0.313}$	$0.835_{0.084}$	$\mathbf{0.915_{0.057}}$	$0.788_{0.220}$	$0.963_{0.065}$
	CDTD	$0.823_{0.159}$	$0.748_{0.334}$	$0.822_{0.090}$	$0.871_{0.135}$	$0.760_{0.248}$	$0.809_{0.179}$
	Vanilla	$0.736_{0.309}$	$0.664_{0.413}$	$0.742_{0.289}$	$0.737_{0.293}$	$0.708_{0.314}$	$0.261_{0.166}$
	TabDDPM	$0.797_{0.161}$	$0.790_{0.304}$	$\mathbf{0.850_{0.060}}$	$0.816_{0.090}$	$0.753_{0.202}$	$0.320_{0.151}$
	TabDiff	$0.831_{0.139}$	$0.738_{0.299}$	$0.794_{0.062}$	$0.912_{0.054}$	$0.785_{0.185}$	$0.460_{0.154}$
	TabSyn	$0.823_{0.133}$	$0.765_{0.293}$	$0.816_{0.053}$	$0.829_{0.135}$	$0.778_{0.175}$	$0.915_{0.059}$
	RHYTHM	$\mathbf{0.845_{0.137}}$	$\mathbf{0.795_{0.280}}$	$0.841_{0.055}$	$0.895_{0.099}$	$\mathbf{0.816_{0.165}}$	$\mathbf{0.967_{0.062}}$
Range & Category	CDCD	$\mathbf{0.836_{0.149}}$	$0.739_{0.320}$	$0.792_{0.080}$	$\mathbf{0.921_{0.029}}$	$0.770_{0.221}$	$0.896_{0.104}$
	CDTD	$0.823_{0.160}$	$0.723_{0.339}$	$0.793_{0.082}$	$0.889_{0.087}$	$0.745_{0.245}$	$0.735_{0.206}$
	Vanilla	$0.724_{0.321}$	$0.646_{0.408}$	$0.720_{0.279}$	$0.693_{0.274}$	$0.691_{0.319}$	$0.133_{0.078}$
	TabDDPM	$0.783_{0.159}$	$0.750_{0.320}$	$\mathbf{0.828_{0.068}}$	$0.759_{0.069}$	$0.722_{0.195}$	$0.206_{0.141}$
	TabDiff	$0.819_{0.154}$	$0.712_{0.311}$	$0.751_{0.094}$	$0.912_{0.030}$	$0.765_{0.184}$	$0.422_{0.137}$
	TabSyn	$0.824_{0.143}$	$0.751_{0.310}$	$0.802_{0.061}$	$0.839_{0.090}$	$0.770_{0.179}$	$0.756_{0.161}$
	RHYTHM	$0.828_{0.143}$	$\mathbf{0.760_{0.293}}$	$0.791_{0.110}$	$0.906_{0.052}$	$\mathbf{0.786_{0.166}}$	$\mathbf{0.909_{0.121}}$

Table 3: Ablation results with different alignment configurations.

Lin. RIC	Num. card.	Accuracy	Column Pair Trends	Column Shapes	Coverage	F1
✓	Adaptive	$\mathbf{0.806_{0.129}}$	$0.830_{0.231}$	$\mathbf{0.895_{0.055}}$	$0.871_{0.144}$	$0.753_{0.157}$
✓	$k = 300$	$0.806_{0.131}$	$\mathbf{0.834_{0.227}}$	$0.893_{0.054}$	$0.858_{0.117}$	$\mathbf{0.755_{0.158}}$
✓	$k = 2$	$0.805_{0.132}$	$0.832_{0.232}$	$0.894_{0.057}$	$\mathbf{0.932_{0.053}}$	$0.754_{0.158}$
✗	Adaptive	$0.793_{0.133}$	$0.773_{0.290}$	$0.878_{0.062}$	$0.929_{0.047}$	$0.732_{0.164}$
✗	$k = 300$	$0.795_{0.133}$	$0.788_{0.285}$	$0.882_{0.057}$	$0.931_{0.041}$	$0.733_{0.168}$
✗	$k = 2$	$0.747_{0.180}$	$0.709_{0.283}$	$0.853_{0.069}$	$0.851_{0.190}$	$0.657_{0.243}$

tion to less than 50%. In contrast, continuous-noise methods (CDCD, CDTD, RHYTHM, TabSyn) achieve higher match rates, indicating the benefit of differentiable sampling for guidance.

Vanilla generally has the lowest quality and match rates across constraints, highlighting that steering is insufficient if the underlying generative model poorly captures the data distribution.

4.4 COMPUTATIONAL COST

Figure 3 reports the runtime cost of Alg.1 as we vary the number of diffusion steps and features being aligned. We use a 12-core AMD Ryzen 9 5900 processor with an NVIDIA RTX 3090 GPU, CUDA 12.1, Ubuntu 20.04.6, and Pytorch 2.2.1 with Python 3.12. We observe that the runtime scales approximately linearly with both quantities. Increasing the number of features from 2 to 16 increases the runtime by roughly 8x, while increasing the number of steps from 10 to 1000 increases the runtime by roughly 100x. Importantly, even for the largest configuration considered, with 1000 diffusion steps and 256 features, alignment takes only 30 seconds on average. In practice, the schedule can thus be computed once before training and reused throughout the entire generative modelling process, making the cost of corruption alignment negligible relative to model training and inference.

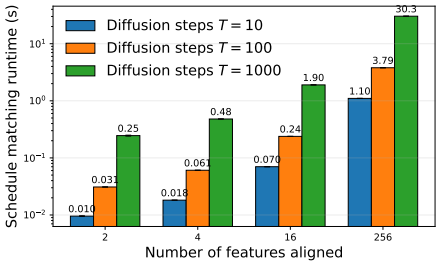


Figure 3: Runtime of the corruption alignment procedure for different numbers of diffusion steps and aligned features.

4.5 ABLATION: ALIGNMENT CONFIGURATIONS

Table 3 tests two key design choices behind our corruption-aligned schedules—the choice of the reference RIC trajectory and the cardinality estimation strategy for continuous features. For the reference, we compare our proposed *linear* RIC schedule (Sec. 3.2) against the default (Sec. 4.1). For cardinality estimation, we compare the adaptive Freedman–Diaconis rule against fixed cardinality assignments, under low-cardinality (K=2) and high-cardinality (K=300) settings.

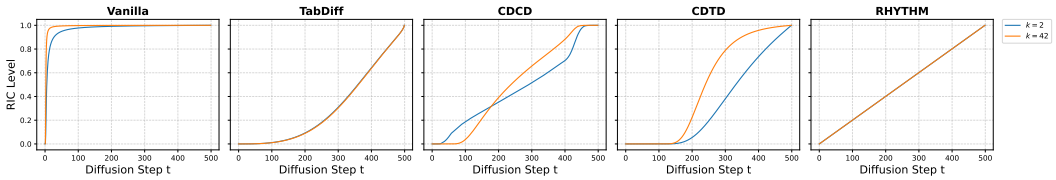


Figure 4: RICs of two categorical features ($K \in \{2, 42\}$) in Adult across methods.

Across all settings, schedules aligned to a linear RIC reference (✓) consistently outperform their un-linearised counterparts (✗). We hypothesise that this may stem from enforcing a smooth, linear information decay that prevents extreme training-loss imbalances across diffusion steps.

For cardinality estimation of continuous features, performance is similar across all strategies when using a linearised RIC reference. But when the reference is nonlinear, approximating continuous features with low cardinality (K=2) worsens performance. These results suggest that the choice of the reference trajectory has a stronger effect on performance than the precise cardinality estimation strategy, with the linearised reference providing greater robustness to cardinality choices.

4.6 Visualisation of RIC Trajectories

Figure 4 compares the RICs learned by schedule-optimisation methods (TabDiff, CDCD, CDTD) against RHYTHM for two categorical features of the Adult dataset (more in Appendix H.1). The uncorrected Vanilla RIC is included as a baseline reference.

Existing training-time methods exhibit trajectories that loosely resemble a linearly growing RIC curve. This suggests that their proxy loss-balancing objectives *implicitly* drive the RIC toward linearity. This phenomenon that aligns with recent insights in discrete diffusion literature (Yang et al., 2026), where balancing denoising difficulty across steps induces approximately linear information dynamics. However, we observe that these learned heuristics remain noisy approximations, that can result in imperfect alignment (e.g. CDCD, CDTD). In contrast, RHYTHM achieves perfectly linear, and cardinality-invariant RIC trajectories in closed form via analytical corruption alignment, eliminating the need for empirical schedule optimisation during training.

5 Discussion and Conclusion

We present RHYTHM, a fully continuous tabular diffusion model combining the empirical strengths of mixed-noise diffusion with the flexibility of continuous-space guidance. We propose the RIC, a cardinality-invariant metric to analyse schedule dynamics. Through it, we prove that established discrete-noise models are inherently *corruption-aligned*, a property we believe to be central to their empirical success. We replicate this synchronising behaviour within RHYTHM through an analytical procedure, bypassing the need for handcrafted functions or training-based optimisation. Since RHYTHM remains entirely continuous, constraints can be flexibly imposed directly during inference, going beyond unconditional generation. For example, a user can require a generated tabular record to satisfy a conjunction of conditions, restrict a numerical feature to a given range, or fix a categorical attribute, without retraining the diffusion model or introducing an auxiliary classifier.

Evaluations demonstrate that RHYTHM bridges the performance gap with state-of-the-art tabular diffusion methods, matching the generation quality of strong mixed-noise and schedule-learning baselines. We also demonstrate RHYTHM's flexibility at inference-time control for constrained generation, without requiring task-specific retraining or introducing additional model complexity. More importantly, our theoretical framework provides an explanation for *why* existing methods succeed.

Future work could explore how corruption alignment can be extended to other high-cardinality data, including graphs and language, where different components may likewise undergo substantially different corruption dynamics. Our findings challenge a core assumption in tabular diffusion—that heterogeneous tabular features require fundamentally distinct noise processes. We show that this is not the case; a fully continuous diffusion with synchronised schedules is sufficient for high-fidelity generation and inference-time control.

AI use Statement. We used generative AI tools to support research ideation and execution, including refining research hypotheses, discussing methodological and experimental choices, and reasoning about algorithmic and mathematical formulations. We also used generative AI tools to aid and polish writing, draft portions of the manuscript, and create or modify software code for data analysis and visualisation. We did not use generative AI tools to generate synthetic datasets, prove mathematical claims, or retrieve or discover related work.

All AI-assisted work was reviewed and verified by the authors. The authors checked mathematical formulations, algorithmic descriptions, code, and interpretations for correctness, and all final scientific and methodological decisions were made by the authors. We take full responsibility for the final content of this work, including all text, claims, analyses, and artifacts produced.

Ethics Statement. This work does not involve human subjects, new data collection, or experiments requiring interaction with individuals. We evaluate our method using existing public benchmark datasets and follow the applicable terms of use for these datasets. No personally identifiable information is collected or released as part of this work.

Reproducibility Statement. We provide detailed descriptions of the proposed corruption-alignment method, including the alignment procedure and Gaussian DIC computation, in the main paper and appendix. The experimental setup, datasets, preprocessing procedures, baseline configurations, evaluation metrics, and inference-time guidance procedures are documented in the main paper and appendix. The source code, experimental configurations, and scripts required to reproduce our results will be made available at `https://anonymous.4open.science/r/BestDiffusion-3B12/README.md`.

REFERENCES

Dry Bean. UCI Machine Learning Repository, 2020. DOI: https://doi.org/10.24432/C50S4B.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 17981–17993. Curran Associates, Inc., 2021. URL `https://proceedings.neurips.cc/paper_files/paper/2021/file/958c530554f78bcd8e97125b70e6973d-Paper.pdf`.

Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 843–852, 2023.

Barry Becker and Ronny Kohavi. Adult. UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.

R. Bock. MAGIC Gamma Telescope. UCI Machine Learning Repository, 2004. DOI: https://doi.org/10.24432/C52C8B.

Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye. Improving diffusion models for inverse problems using manifold constraints. *Advances in Neural Information Processing Systems*, 35:25683–25696, 2022.

Synthetic Data Metrics. DataCebo, Inc., 01 2026. URL `https://docs.sdv.dev/sdmetrics/`. Version 0.27.0.

Sander Dieleman, Laurent Sartran, Arman Roshannai, Nikolay Savinov, Yaroslav Ganin, Pierre H Richemond, Arnaud Doucet, Robin Strudel, Chris Dyer, Conor Durkan, et al. Continuous diffusion for categorical data. *arXiv preprint arXiv:2211.15089*, 2022.

Dheeru Dua, Casey Graff, et al. UCI machine learning repository. 2017. URL `http://archive.ics.uci.edu/ml`.

David Freedman and Persi Diaconis. On the histogram as a density estimator: L 2 theory. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 57(4):453–476, 1981.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Nate Gruver, Samuel Stanton, Nathan C. Frey, Tim G. J. Rudner, Isidro Hötzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew Gordon Wilson. Protein design with guided discrete diffusion. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *NeurIPS*, 2023.

Yutong He, Naoki Murata, Chieh-Hsin Lai, Yuhta Takida, Toshimitsu Uesaka, Dongjun Kim, Wei-Hsiang Liao, Yuki Mitsufuji, J Zico Kolter, Ruslan Salakhutdinov, and Stefano Ermon. Manifold preserving guided diffusion. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=o3BxOLoxm1.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in neural information processing systems*, 34:12454–12465, 2021.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems*, 35:26565–26577, 2022.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun (eds.), *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6114.

Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. Tabddpm: Modelling tabular data with diffusion models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 17564–17579. PMLR, 2023. URL https://proceedings.mlr.press/v202/kotelnikov23a.html.

Renata Madeo, Priscilla Wagner, and Sarajane Peres. Gesture Phase Segmentation. UCI Machine Learning Repository, 2013. DOI: https://doi.org/10.24432/C5Z32C.

Markus Mueller, Kathrin Gruber, and Dennis Fok. Continuous diffusion for mixed-type tabular data. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=QPtoBPn4lZ.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pp. 8162–8171. PmLR, 2021.

Patrick Pynadath, Jiaxin Shi, and Ruqi Zhang. Candi: Hybrid discrete-continuous diffusion models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=03KQGUvw1e.

Subham Sekhar Sahoo, Justin Deschenaux, Aaron Gokaslan, Guanghan Wang, Justin T Chiu, and Volodymyr Kuleshov. The diffusion duality. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=9P9Y8FOSOk.

C. Sakar and Yomi Kastro. Online Shoppers Purchasing Intention Dataset. UCI Machine Learning Repository, 2018. DOI: https://doi.org/10.24432/C5F88Q.

Davide Scassola, Sebastiano Saccani, Ginevra Carbone, and Luca Bortolussi. Zero-shot conditioning of score-based diffusion models by neuro-symbolic constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 20302–20309, 2025.

Yair Schiff, Subham Sekhar Sahoo, Hao Phung, Guanghan Wang, Sam Boshar, Hugo Dalla-torre, Bernardo P. de Almeida, Alexander M. Rush, Thomas Pierrot, and Volodymyr Kuleshov. Simple guidance mechanisms for discrete diffusion models. In *ICLR*, 2025.

Aditya Shankar, Yuandou Wang, Rihan Hai, and Lydia Y. Chen. Harpoon: Generalised manifold guidance for conditional tabular diffusion. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=G5g6tDg1ZE.

Juntong Shi, Minkai Xu, Harper Hua, Hengrui Zhang, Stefano Ermon, and Jure Leskovec. Tabdiff: a mixed-type diffusion model for tabular data generation. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=swvURjrt8z.

David Slate. Letter Recognition. UCI Machine Learning Repository, 1991. DOI: https://doi.org/10.24432/C5ZP40.

Zhihan Yang, Wei Guo, Shuibai Zhang, Subham Sekhar Sahoo, Yongxin Chen, Arash Vahdat, Morteza Mardani, and John Thickstun. Continuous diffusion scales competitively with discrete diffusion for language. *arXiv preprint arXiv:2605.18530*, 2026.

I-Cheng Yeh. Default of Credit Card Clients. UCI Machine Learning Repository, 2009. DOI: https://doi.org/10.24432/C55S3H.

Hengrui Zhang, Jiani Zhang, Zhengyuan Shen, Balasubramaniam Srinivasan, Xiao Qin, Christos Faloutsos, Huzefa Rangwala, and George Karypis. Mixed-type tabular data synthesis with score-based diffusion in latent space. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=4Ay23yeuz0.

Hengrui Zhang, Liancheng Fang, Qitian Wu, and Philip S. Yu. Diffputer: An EM-driven diffusion model for missing data imputation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=3fl1SENSYO.

Zilong Zhao, Aditya Kunar, Robert Birke, and Lydia Y Chen. CTAB-GAN: Effective table data synthesizing. In *Asian Conference on Machine Learning*, pp. 97–112. PMLR, 2021.

A NOTATIONS

Table 4: Summary of key notations used in the main text.

Symbol	Explanation
(A) Data Setup & Feature Dimensions	
x_0	Clean tabular sample (row) drawn from the data distribution
$d_{\text{cont}}, d_{\text{cat}}$	Number of continuous and categorical features, respectively
$x_0^{\text{cont}(j)}$	The jth continuous feature column ($j \in [1..d_{\text{cont}}]$)
$x_0^{(i)}$	The ith categorical feature column ($i \in [1..d_{\text{cat}}]$)
K_i	Number of categories (cardinality) of the ith categorical feature
$K_{\text{eff}}^{(j)}$	Effective cardinality of the jth continuous feature
h_j	Freedman–Diaconis bin width for continuous feature j
$\text{IQR}(\cdot)$	Inter-quartile range of a continuous feature
m	Total number of training samples
e_k	Standard k^{th} one-hot basis vector
$m = e_{K_i+1}$	Special mask category one-hot vector in Masked Diffusion
(B) Forward & Reverse Diffusion Processes	
T	Total number of diffusion steps
$t \in [0..T]$	Diffusion timestep index
x_t	Noised sample vector at diffusion step t
$\epsilon \sim \mathcal{N}(0, I)$	Standard isotropic Gaussian noise vector
$\bar{\alpha}_t$	Cumulative noise schedule at timestep t ($\bar{\alpha}_t \in [0, 1]$)
$\bar{\alpha}_t^{\prime(i)}$	Feature-specific noise schedule value for feature i at step t
$\text{Cat}(\cdot)$	Categorical (multinomial) probability distribution
$\Phi(\cdot)$	Standard Gaussian Cumulative Distribution Function (CDF)
$p_\theta(x_{t-1}\|x_t)$	Parameterised reverse diffusion process with neural network parameters θ
(C) Corruption Alignment Theory	
$\rho_i(t)$	Discrete Identity Corruption (DIC) for the ith feature at step t
$\rho_i(T)$	Maximum achievable DIC limit for feature i at step T
$\bar{\rho}_i(t)$	Relative Identity Corruption (RIC) for feature i at step t
$\bar{\rho}_{\text{ref}}(t)$	Shared reference RIC trajectory
$\rho_i^*(t)$	Target DIC value for feature i at timestep t
(D) Alignment Algorithm & Numerical Quadrature	
$[\bar{\alpha}'_{\text{low}}, \bar{\alpha}'_{\text{high}}]$	Search bounds for bisection optimization
N	Number of bisection search iterations
Q	Number of numerical quadrature grid points
c_q, w_q	Quadrature grid point and corresponding integration weight

B PROOFS

B.1 PROOF OF PROPOSITION 1

Proposition 1 (Discrete Identity Corruption under Different Diffusion Mechanisms). *For a categorical feature (indexed by i) with $K_i \geq 2$ categories and noise schedule $\{\bar{\alpha}_t\}_{t=0}^T$, the DIC monotonically increases with t and is given by*

$$\rho_i(t) = \begin{cases} \displaystyle\int_{-\infty}^{\infty} \left(1 - \left(\Phi\left(s + \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\right)\right)^{K_i - 1}\right) \mathcal{N}(s; 0, 1)\, ds, & \text{Gaussian diffusion,} \\ \left(1 - \dfrac{1}{K_i}\right)(1 - \bar{\alpha}_t), & \text{Multinomial diffusion,} \\ 1 - \bar{\alpha}_t, & \text{Masked diffusion.} \end{cases}$$

where $\Phi(\cdot)$ denotes the standard Gaussian cumulative distribution function (CDF).

Proof. We consider a categorical feature i with K_i categories, represented as a one-hot vector $\mathbf{x}_0^{(i)} \in \{0,1\}^{K_i}$. Without loss of generality, we assume $x_0^{(i)}[1] = 1$ and $x_0^{(i)}[j] = 0, \forall j \neq 1$, with discrete identity determined via $\mathrm{argmax}(\mathbf{x}_t^{(i)})$. Following Definition 1, identity is lost when at least one incorrect coordinate exceeds the ground-truth:

$$\rho_i(t) = \mathbb{P}\left(\exists j \neq 1 : x_t^{(i)}[j] > x_t^{(i)}[1]\right) = 1 - \mathbb{P}\left(x_t^{(i)}[j] < x_t^{(i)}[1],\, \forall j \neq 1\right). \tag{12}$$

Gaussian Diffusion. While the Gaussian case is already analysed by Sahoo et al. (2025); Pynadath et al. (2026), we provide an extended version, which critically also proves the monotonicity of the curve. Under forward Gaussian corruption (1), each component evolves as $x_t^{(i)}[u] = \sqrt{\bar{\alpha}_t} x_0^{(i)}[u] + \sqrt{1-\bar{\alpha}_t}\epsilon^{(i)}[u]$ with $\epsilon^{(i)} \sim \mathcal{N}(\mathbf{0}, I_{K_i})$.

Conditioning on the noisy ground-truth coordinate $x_t^{(i)}[1] = \sqrt{\bar{\alpha}_t} + \sqrt{1-\bar{\alpha}_t}s$, where $s \sim \mathcal{N}(0,1)$, Equation 12 expands to:

$$\rho_i(t) = 1 - \int_{-\infty}^{\infty} \mathbb{P}\left(x_t^{(i)}[j] < \sqrt{\bar{\alpha}_t} + \sqrt{1-\bar{\alpha}_t}s,\, \forall j \neq 1 \,\middle|\, s\right) \mathcal{N}(s; 0, 1)\, ds. \tag{13}$$

For all incorrect categories $j \neq 1$, $x_0^{(i)}[j] = 0$, so $x_t^{(i)}[j] \sim \mathcal{N}(0, 1-\bar{\alpha}_t)$. Accounting for scaling by the standard deviation yields the standard Gaussian CDF $\Phi(\cdot)$:

$$\mathbb{P}\left(x_t^{(i)}[j] < \sqrt{\bar{\alpha}_t} + \sqrt{1-\bar{\alpha}_t}s\right) = \Phi\left(s + \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\right).$$

Since noise is added independently across coordinates, the joint probability over all $K_i - 1$ incorrect classes is given by $\left(\Phi\left(s + \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\right)\right)^{K_i-1}$. Substituting this into (13) and utilising $\int_{-\infty}^{\infty} \mathcal{N}(s; 0, 1)\, ds = 1$ yields the desired result:

$$\rho_i(t) = \int_{-\infty}^{\infty} \left(1 - \left(\Phi\left(s + \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\right)\right)^{K_i-1}\right) \mathcal{N}(s; 0, 1)\, ds. \tag{14}$$

Multinomial Diffusion. For multinomial diffusion Kotelnikov et al. (2023); Hoogeboom et al. (2021), the corrupted categorical state remains a discrete random variable. The forward process is

$$p(x_t^{(i)} | x_0^{(i)}) = \mathrm{Cat}\left(\bar{\alpha}_t x_0^{(i)} + (1 - \bar{\alpha}_t)\frac{\mathbf{1}}{K_i}\right).$$

Identity corruption occurs when the sampled category differs from the original category. The probability of retaining the original category is

$$\mathbb{P}(\text{argmax}(x_t^{(i)}) = x_0^{(i)}) = \bar{\alpha}_t + (1 - \bar{\alpha}_t)\frac{1}{K_i}.$$

Consequently, the identity corruption probability is

$$\rho_i(t) = 1 - \left(\bar{\alpha}_t + \frac{1 - \bar{\alpha}_t}{K_i}\right) = \left(1 - \frac{1}{K_i}\right)(1 - \bar{\alpha}_t).$$

Masked Diffusion. For masked diffusion, the categorical variable is progressively replaced by the masked state m:

$$p(x_t^{(i)}|x_0^{(i)}) = \text{Cat}\left(\bar{\alpha}_t x_0^{(i)} + (1 - \bar{\alpha}_t)m\right).$$

The original category is preserved only when the sample remains unmasked, which occurs with probability $\bar{\alpha}_t$. Therefore,

$$\rho_i(t) = 1 - \bar{\alpha}_t.$$

These derivations show that discrete diffusion mechanisms provide direct control over identity corruption through their transition probabilities, whereas Gaussian diffusion induces a feature-cardinality-dependent corruption curve due to the competition between noisy category representations.

Monotonicity Proofs. To prove that discrete identity corruption $\rho_i(t)$ strictly increases with time step t, consider a positive time increment $h > 0$. The noise schedule is defined as a cumulative product of per-step signal retentions $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ (Ho et al., 2020) where each $\alpha_s \in (0, 1)$. Hence signal retention strictly decays over time, giving $\bar{\alpha}_{t+h} < \bar{\alpha}_t$ for any $h > 0$. Evaluating the DIC at step $t + h$ for each mechanism yields the following:

(a) Gaussian Diffusion: At step $t + h$, the corruption level is:

$$\rho_i(t + h) = \int_{-\infty}^{\infty} \left(1 - \Phi\left(s + \frac{\sqrt{\bar{\alpha}_{t+h}}}{\sqrt{1 - \bar{\alpha}_{t+h}}}\right)^{K_i - 1}\right) \mathcal{N}(s; 0, 1)\, ds.$$

We have $\bar{\alpha}_{t+h} < \bar{\alpha}_t \implies \frac{\sqrt{\bar{\alpha}_{t+h}}}{\sqrt{1-\bar{\alpha}_{t+h}}} < \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}$. Since the standard Gaussian CDF $\Phi(\cdot)$ is strictly increasing, $\Phi\left(s + \frac{\sqrt{\bar{\alpha}_{t+h}}}{\sqrt{1-\bar{\alpha}_{t+h}}}\right) < \Phi\left(s + \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\right)$ for all $s \in \mathbb{R}$. Raising to $K_i - 1 \geq 1$ and subtracting from 1 reverses the inequality:

$$1 - \left(\Phi\left(s + \frac{\sqrt{\bar{\alpha}_{t+h}}}{\sqrt{1 - \bar{\alpha}_{t+h}}}\right)\right)^{K_i - 1} > 1 - \left(\Phi\left(s + \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1 - \bar{\alpha}_t}}\right)\right)^{K_i - 1}, \qquad \forall s \in \mathbb{R}.$$

Integrating this strictly larger integrand over the positive density $\mathcal{N}(s; 0, 1) > 0$ establishes:

$$\rho_i(t + h) > \rho_i(t).$$

Multinomial Diffusion: At step $t + h$, the corruption level is:

$$\rho_i(t + h) = \left(1 - \frac{1}{K_i}\right)(1 - \bar{\alpha}_{t+h}).$$

Since $K_i \geq 2$, the cardinality scaling factor satisfies $\left(1 - \frac{1}{K_i}\right) > 0$. Combining this with $\bar{\alpha}_{t+h} < \bar{\alpha}_t$ gives:

$$\left(1 - \frac{1}{K_i}\right)(1 - \bar{\alpha}_{t+h}) > \left(1 - \frac{1}{K_i}\right)(1 - \bar{\alpha}_t) \implies \rho_i(t + h) > \rho_i(t).$$

Masked Diffusion: At step $t + h$, the corruption level is:

$$\rho_i(t + h) = 1 - \bar{\alpha}_{t+h}.$$

Since $\bar{\alpha}_{t+h} < \bar{\alpha}_t$, it immediately follows that $1 - \bar{\alpha}_{t+h} > 1 - \bar{\alpha}_t$, and thus:

$$\rho_i(t + h) > \rho_i(t).$$

Taking the positive limit as $h \to 0^+$, we obtain $\lim_{h\to 0^+} \frac{\rho_i(t+h)-\rho_i(t)}{h} > 0$ across all three mechanisms, proving that DIC is strictly monotonically increasing in t. □

B.2 Proof of Theorem 1

Theorem 1 (Inherent Corruption Alignment in Discrete Noise Paradigms). *Let u and v be any two categorical features with distinct cardinalities $K_u, K_v \geq 2$. Under a shared noise schedule $\{\bar{\alpha}_t\}_{t=0}^T$ across features, Masked and Multinomial diffusion inherently satisfy corruption alignment:*

$$\bar{\rho}_u(t) = \bar{\rho}_v(t) = 1 - \bar{\alpha}_t, \qquad \forall t \in [0..T].$$

Conversely, Gaussian diffusion is misaligned, i.e., $\exists\, t \in [0..T]$ such that $\bar{\rho}_u(t) \neq \bar{\rho}_v(t)$.

Proof. **1. Alignment in Masked and Multinomial Diffusion.** For masked diffusion, $\rho_u(t) = 1 - \bar{\alpha}_t$ and terminal corruption $\rho_u(T) = 1$, yielding $\bar{\rho}_u(t) = \frac{1-\bar{\alpha}_t}{1} = 1 - \bar{\alpha}_t$. For multinomial diffusion, $\rho_u(t) = \left(1 - \frac{1}{K_u}\right)(1 - \bar{\alpha}_t)$ and $\rho_u(T) = 1 - \frac{1}{K_u}$. Normalising by $\rho_u(T)$ cancels the cardinality-dependent factor $\left(1 - \frac{1}{K_u}\right)$, giving $\bar{\rho}_u(t) = 1 - \bar{\alpha}_t$. Thus, both paradigms yield $\bar{\rho}_u(t) = \bar{\rho}_v(t) = 1 - \bar{\alpha}_t$ invariant of feature cardinalities K_u and K_v.

2. Misalignment in Gaussian Diffusion. For Gaussian diffusion, the relative identity corruption $\bar{\rho}_u(t)$ depends on cardinality K_u through the non-linear exponent $(\Phi(\cdot))^{K_u-1}$ inside the integral:

$$\bar{\rho}_u(t) = \frac{\int_{-\infty}^{\infty} \left(1 - \left(\Phi\left(s + \sqrt{\frac{\bar{\alpha}_t}{1-\bar{\alpha}_t}}\right)\right)^{K_u-1}\right) \mathcal{N}(s; 0, 1)\, ds}{\int_{-\infty}^{\infty} \left(1 - (\Phi(s))^{K_u-1}\right) \mathcal{N}(s; 0, 1)\, ds}. \tag{15}$$

For the denominator, we can substitute $a = \Phi(s)$, which makes the denominator integral $\int_0^1 (1 - a^{K_u-1}) da = 1 - \frac{1}{K_u}$. Substituting this into (15) gives us the following:

$$\bar{\rho}_u(t) = \frac{\int_{-\infty}^{\infty} \left(1 - \left(\Phi\left(s + \sqrt{\frac{\bar{\alpha}_t}{1-\bar{\alpha}_t}}\right)\right)^{K_u-1}\right) \mathcal{N}(s; 0, 1)\, ds}{1 - \frac{1}{K_u}}.$$

To achieve corruption alignment between two features u and v with distinct cardinalities under a shared schedule $\bar{\alpha}_t$, we require $\bar{\rho}_u(t) = \bar{\rho}_v(t)$ for all $t \in [0, T]$, implying the following condition:

$$\frac{\int_{-\infty}^{\infty} \left(1 - \left(\Phi\left(s + \sqrt{\frac{\bar{\alpha}_t}{1-\bar{\alpha}_t}}\right)\right)^{K_u-1}\right) \phi(s)\, ds}{\int_{-\infty}^{\infty} \left(1 - \left(\Phi\left(s + \sqrt{\frac{\bar{\alpha}_t}{1-\bar{\alpha}_t}}\right)\right)^{K_v-1}\right) \phi(s)\, ds} = \frac{1 - \frac{1}{K_u}}{1 - \frac{1}{K_v}}. \tag{16}$$

For Equation 16 to hold continuously across t, the left-hand side ratio of integrals, which depends non-linearly on t through the signal-to-noise ratio $\frac{\bar{\alpha}_t}{1-\bar{\alpha}_t}$, would have to remain constant and equal to the time-invariant right-hand side ratio of cardinalities.

For any intermediate timestep $t \in (0, T)$, the integral on the left scales non-linearly with K_u due to the exponent on $\Phi(\cdot)$, causing the ratio to strictly vary with t. Thus, for $K_u \neq K_v$, Equation 16 cannot hold across $t \in (0, T)$, proving inherent corruption misalignment under a shared Gaussian noise schedule. □

C Additional background on diffusion models

For all three diffusion noise paradigms (Gaussian, Multinomial and Masked) mentioned in Sec.2, the reverse process $p_\theta(x_{t-1}|x_t)$ is trained by minimising a variational bound on the negative log-likelihood. The specific form this bound takes, and how p_θ is parametrised, differs across paradigms.

C.1 Gaussian Diffusion

Training objective. Gaussian diffusion models train a network (parametrised by θ) to predict the noise ϵ injected at a randomly sampled step t via a squared-error between the true and predicted noise (Ho et al., 2020).

$$\mathcal{L}_{\text{Gauss}}(\theta) = \mathbb{E}_{x_0,t,\epsilon}\left[\|\epsilon - \epsilon_\theta(x_t,t)\|^2\right], \qquad \epsilon \sim \mathcal{N}(0,I), \tag{17}$$

Reverse step. Given ϵ_θ, the reverse transition is parametrized as a Gaussian with mean computed by inverting the forward corruption:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}\big(x_{t-1};\, \mu_\theta(x_t,t),\, \sigma_t^2 I\big), \qquad \mu_\theta(x_t,t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar\alpha_t}}\,\epsilon_\theta(x_t,t)\right), \tag{18}$$

with σ_t^2 denoting the posterior variance. Ho et al. (2020) use $\sigma_t^2 = \frac{1-\bar\alpha_{t-1}}{1-\bar\alpha_t}(1-\alpha_t)$. While the above ϵ-prediction objective is helpful for understanding the reverse sampling process, our implementation adopts the EDM style for training (Karras et al., 2022). Specifically, we train the network to directly predict x_0 using variance-scaled loss weighting and continuous noise sampling, which provides improved training stability and sample quality over standard ϵ-matching.

C.2 Multinomial Diffusion

Training objective. For each categorical feature $x_0^{(i)}$, represented as a one-hot vector, training maximises a variational lower bound (ELBO) on the data log-likelihood, obtained by decomposing the bound (Kotelnikov et al., 2023; Hoogeboom et al., 2021). In practice, this simplifies to minimising the following KL divergence at each step t:

$$\mathcal{L}_{\text{mult}}^{(i)}(\theta) = \mathbb{E}_{x_0,t,x_t}\Big[\mathrm{KL}\big(p(x_{t-1}^{(i)}|x_t^{(i)},x_0^{(i)}) \,\|\, p_\theta(x_{t-1}^{(i)}|x_t)\big)\Big]. \tag{19}$$

The target $p(x_{t-1}^{(i)}|x_t^{(i)},x_0^{(i)})$ is the true forward posterior, obtained from the forward process in (2):

$$p\big(x_{t-1}^{(i)}|x_t^{(i)},x_0^{(i)}\big) = \frac{\big[\alpha_t x_t^{(i)} + (1-\alpha_t)\frac{\mathbf{1}}{K_i}\big] \odot \big[\bar\alpha_{t-1} x_0^{(i)} + (1-\bar\alpha_{t-1})\frac{\mathbf{1}}{K_i}\big]}{\pi}, \tag{20}$$

where $\odot$ denotes the elementwise product and π is a normalising constant. Since $x_0^{(i)}$ is unavailable at inference time, the reverse model $p_\theta(x_{t-1}^{(i)}|x_t^{(i)})$ is parametrised by substituting the network's prediction $\hat x_0^{(i)} = p_\theta(x_0^{(i)}|x_t,t)$ into the same posterior form, so that $\mathcal{L}_{\text{mult}}^{(i)}$ effectively becomes a cross-entropy-style loss between $\hat x_0^{(i)}$ and the true $x_0^{(i)}$.

Reverse step. At sampling time, the reverse transition plugs $\hat x_0^{(i)}$ into (20) in place of $x_0^{(i)}$:

$$p_\theta\big(x_{t-1}^{(i)}|x_t\big) = p\big(x_{t-1}^{(i)}|x_t^{(i)},\hat x_0^{(i)}\big), \qquad \hat x_0^{(i)} = p_\theta\big(x_0^{(i)}|x_t,t\big), \tag{21}$$

i.e., (20) evaluated with $x_0^{(i)}$ replaced by the network's predicted distribution over the classes.

C.3 Masked Diffusion

Training objective. The training loss for masked diffusion (Shi et al., 2025) is a reweighted cross-entropy evaluated only at positions currently masked, since once a token is unmasked it cannot revert back to the masked state m:

$$\mathcal{L}_{\text{mask}}^{(i)}(\theta) = -\,\mathbb{E}_{x_0,t,x_t}\left[\mathbf{1}\Big[x_t^{(i)} = m\Big]\; w_t\, \log p_\theta\big(x_0^{(i)}|x_t\big)\right], \qquad w_t = \frac{\bar\alpha_{t-1}-\bar\alpha_t}{1-\bar\alpha_t}. \tag{22}$$

Reverse step. Since unmasked tokens are frozen under the forward process, the true posterior retains $x_t^{(i)}$ whenever $x_t^{(i)} \neq m$, and only redistributes mass over classes and the mask token when $x_t^{(i)} = m$:

$$p_\theta\big(x_{t-1}^{(i)} | x_t\big) = \begin{cases} \mathrm{Cat}(x_t^{(i)}), & x_t^{(i)} \neq m, \\ \mathrm{Cat}\left(x_{t-1}^{(i)};\ \dfrac{(1-\bar{\alpha}_{t-1})\, m + (\bar{\alpha}_{t-1} - \bar{\alpha}_t)\, \hat{x}_0^{(i)}}{1-\bar{\alpha}_t}\right), & x_t^{(i)} = m, \end{cases} \tag{23}$$

where $\hat{x}_0^{(i)} = p_\theta(x_0^{(i)} | x_t, t)$ is again the model's predicted class distribution.

C.4 Training-Free Guidance for Conditional Generation

In inference-time conditional generation, the goal is to sample from the conditional posterior distribution $p(x_{t-1} \mid x_t, c)$, where c represents an arbitrary differentiable constraint or loss function $\mathcal{L}_{\text{guide}}(\hat{x}_0(x_t), c)$ defined on the predicted clean sample $\hat{x}_0$. Training-free inference-time guidance applies decompose the transition into an unconditional step and a correcting guidance term:

$$x'_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)\right) + \sqrt{\frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}}\epsilon, \quad \epsilon \sim \mathcal{N}(0, I) \tag{24}$$

$$x_{t-1} = x'_{t-1} - \eta \cdot \nabla_{x_t} \mathcal{L}_{\text{guide}}\big(\hat{x}_0(x_t), c\big) \tag{25}$$

where $\eta > 0$ denotes the guidance strength. We use differentiable guidance losses $\mathcal{L}_{\text{guide}}$ according to the target constraint type (see Appendix G.3) following Shankar et al. (2026):

Range Constraints: For the jth continuous feature bounded by $[\min, \max]$, the guidance penalises values falling outside the valid interval using a ReLU-based boundary distance loss:

$$\mathcal{L}_{\text{guide}}(\hat{x}_0) = \mathrm{ReLU}(\min - \hat{x}_0^{\text{cont}(j)}) + \mathrm{ReLU}(\hat{x}_0^{\text{cont}(j)} - \max). \tag{26}$$

Categorical Constraints: For the ith categorical feature, we minimise the negative log-likelihood of the probability mass assigned to the valid classes $\mathcal{C}_{\text{allowed}}$:

$$\mathcal{L}_{\text{guide}}(\hat{x}_0) = -\log\left(\sum_{k \in \mathcal{C}_{\text{allowed}}} \mathrm{softmax}\big(\mathrm{logits}(\hat{x}_0^{(i)})\big)_k + \epsilon\right). \tag{27}$$

Composite Logic Constraints: Multiple constraint losses can be combined logically during inference. For conjunctions, we sum the individual range and categorical constraint losses, enforcing joint constraint satisfaction. For disjunctions, we multiply the individual constraint losses, allowing the model to satisfy either constraint.

Since RHYTHM operates entirely within a continuous score-based formulation, the predicted clean sample $\hat{x}_0(x_t)$ is a continuous, differentiable function of the noisy input x_t. Consequently, gradients from arbitrary inference-time constraint losses $\mathcal{L}_{\text{guide}}$ flow directly back to x_t via backpropagation through the neural network backbone.

In contrast, discrete diffusion paradigms (Multinomial and Masked) parametrise their reverse steps over categorical indicator vectors or discrete probability simplexes via discrete state transitions (21) and (23). Since discrete sampling steps ($x_t \sim \mathrm{Cat}(\cdot)$) and argmax operations are non-differentiable, exact gradient-based guidance $\nabla_{x_t}\mathcal{L}_{\text{guide}}$ cannot be computed. Discrete models are thus forced to rely on ad-hoc heuristics such as discrete logit-steering or rejection sampling, which lack rigorous theoretical guarantees (Schiff et al., 2025; Gruver et al., 2023). We provide details of the specific guidance heuristics for the baselines in Appendix F.

D ADAPTIVE NOISE SCHEDULE PARAMETRISATIONS

We provide background on two prominent training-based paradigms for optimising feature-wise noise schedules: *timewarping* and *parametric* approaches.

D.1 TIMEWARPED SCHEDULES

Timewarping (Mueller et al., 2025; Dieleman et al., 2022) adapts the noise schedule so that the sampling probability distribution tracks the feature-specific loss curve. The higher the denoising loss at a given noise level $\bar{\alpha}_t$, the higher the probability $p(\bar{\alpha}_t)$ of drawing that noise level during training, i.e., $p(\bar{\alpha}_t) \propto \mathcal{L}(\bar{\alpha}_t)$. This implicitly balances the training loss across timesteps, directing network capacity toward difficult feature-noise regimes, boosting performance.

To draw noise levels according to $p(\bar{\alpha}_t)$ using normalised uniformly drawn timesteps $t \sim \text{Unif}(0, 1)$, t passed through the inverse of a Cumulative Distribution Function (CDF), $\bar{\alpha}_t = F^{-1}(t)$. In practice, an auxiliary network $F_\phi(\bar{\alpha}_t)$ is trained alongside the main diffusion network as a parametric surrogate for the loss profile.

To train $F_\phi(\bar{\alpha}_t)$ time-warping optimises an auxiliary *importance-weighted* loss:

$$\mathcal{L}_{\text{warp}}(\phi) = \mathbb{E}_{\bar{\alpha}_t \sim p_\phi}\left[\frac{\left(F_\phi(\bar{\alpha}_t) - \mathcal{L}(\bar{\alpha}_t)\right)^2}{p_\phi(\bar{\alpha}_t)}\right]$$

Dividing by the probability density $p_\phi(\bar{\alpha}_t)$ cancels out the non-uniform sampling weight inside the expectation integral, preventing $F_\phi(\bar{\alpha}_t)$ from collapsing or neglecting rare, low-density noise regions.

Practical implementations use $F_\phi(\bar{\alpha}_t)$ as a bounded monotonic proxy ($0 \leq F_\phi \leq 1$). Regressing F_ϕ against the observed loss forces it to climb rapidly where loss is high. Because the sampling density is $p_\phi(\bar{\alpha}_t) = \frac{dF_\phi}{d\bar{\alpha}_t}$, a steep region in F_ϕ automatically creates a peak in the sampling density p_ϕ centred at the highest-loss regimes.

Existing frameworks primarily differ in how they parameterise this proxy function F_ϕ. Dieleman et al. (2022) discretise the noise domain into discrete bins with predicted positive weights, forming $F_\phi(\bar{\alpha}_t)$ as a continuous cumulative sum. This non-parametric approach offers maximum flexibility to fit arbitrary loss curves. In contrast, Mueller et al. (2025) parametrises $F_\phi(\bar{\alpha}_t)$ using a scaled S-shaped logistic function. Its derivative $p_\phi(\bar{\alpha}_t)$ forms a unimodal bell curve that peaks explicitly at the loss curve's inflexion point, trading multi-peak flexibility for fast, closed-form inversion.

D.2 PARAMETRIC SCHEDULES

In contrast to timewarping models that train auxiliary networks, Shi et al. (2025) uses a lighter, feature-wise formulation that parameterises the noise growth with closed-form functions mapping a normalised time variable $t \in [0, 1]$ to feature-specific noise levels.

For continuous features, the total noise level for feature i is set as $\gamma_i(t) = \frac{\sqrt{1-\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_t}}$, which is then modelled as a power-mean interpolation between pre-defined boundaries $[\gamma_{\min}, \gamma_{\max}]$:

$$\gamma_i(t) = \left(\gamma_{\min}^{1/\lambda_i} + t \cdot \left(\gamma_{\max}^{1/\lambda_i} - \gamma_{\min}^{1/\lambda_i}\right)\right)^{\lambda_i} \tag{28}$$

where the parameter λ_i controls the curvature of noise accumulation over time.

Categorical schedules are set using a log-linear parametrisation:

$$\gamma_i(t) = -\log\left(1 - \left[(1 - \epsilon_{\max} - \epsilon_{\min})t^{\lambda_i} + \epsilon_{\min}\right]\right)$$

where $\epsilon_{\min}, \epsilon_{\max} > 0$ serve as numerical stability bounds ensuring transitions remain well-behaved near $t = 0$ and $t = 1$. The parameter λ_i governs the rate at which categorical tokens transition into noise. In both cases, the parameter λ_i is updated jointly with the diffusion model, unlike timewarping, which optimises an auxiliary network separately.

Table 5: Datasets used in our experiments, along with the maximum and minimum cardinalities of the categorical features. #Cat. includes the target column used in downstream tasks.

Dataset	Task	Target	#Cat. features	#Num. features	Max. Cardinality	Min. Cardinality
Adult	Binary	`salary`	9	6	42	2
Bean	Multiclass	`Class`	1	16	7	7
Churn	Binary	`Exited`	6	7	2932	2
Default	Binary	`default payment`	4	20	7	2
Gesture	Multiclass	`phase`	1	32	5	5
Intrusion	Binary	`class`	4	38	66	2
Letter	Multiclass	`lettr`	1	16	26	26
Magic	Binary	`class`	1	10	2	2
Shoppers	Binary	`Revenue`	4	14	10	2

E Dataset and Preprocessing Details

Datasets. Table 5 summarises the nine datasets used in our experiments, their prediction task, target column, and source. Seven datasets (Adult (Becker & Kohavi, 1996), Bean (bea, 2020), Default (Yeh, 2009), Gesture (Madeo et al., 2013), Letter (Slate, 1991), Magic (Bock, 2004), Shoppers (Sakar & Kastro, 2018)) are obtained from the UCI Machine Learning Repository (Dua et al., 2017); Churn and Intrusion are obtained from Kaggle. Datasets span binary classification, multiclass classification, and regression tasks, and vary in the proportion of categorical to continuous features, allowing us to assess robustness across heterogeneous feature compositions.

Preprocessing. All rows with missing values are dropped. For datasets with dataset-specific structure, additional cleaning is applied before this step: for Default, the `ID` column is dropped; for Gesture, the individual trials are concatenated into a single table. Each dataset is then split into training and test sets with a $70/30$ ratio using a fixed random seed ($\text{seed} = 42$) for reproducibility, and column types (categorical vs. continuous) are recorded per dataset based on a predefined feature-type mapping.

Categorical features, including the target column when applicable, are encoded as one-hot vectors. Continuous features are standardised to zero mean and unit variance (Gaussian standardisation) using statistics computed on the training split only, and the same transformation is applied to the test split. This preprocessing pipeline is shared across all baselines and our method to ensure a fair comparison.

F BASELINE MODEL CONFIGURATIONS

We first describe the shared backbone architecture for baselines, general hyperparameter configurations, training objectives, and sampling details. Specifics regarding how guidance was adapted for mixed-noise (TabDiff and TabDDPM) and latent diffusion models (TabSyn) are in Appendix F.4.

F.1 SHARED BACKBONE ARCHITECTURE

To ensure a fair comparison across diffusion paradigms, all baselines (TabDiff, TabDDPM, TabSyn, CDCD, CDTD, and Vanilla) use an identical MLP denoiser backbone, differing only in their respective noise schedules and forward/reverse diffusion processes. The backbone consists of a timestep embedding module followed by a residual MLP trunk.

Timestep embedding. Diffusion timesteps are embedded via a sinusoidal positional encoding of dimension $d_{\text{emb}} = 256$, following the standard transformer-style construction: for embedding dimension d and timestep t, we compute $\lceil d/2 \rceil$ frequencies $f_i = \exp\big(-\frac{\log(T_{\max})}{\lceil d/2 \rceil} i\big)$ with $T_{\max} = 10{,}000$, and form $\big[\cos(t \times f_i), \sin(t \times f_i)\big]$. This embedding is passed through a lightweight feed-forward network (two linear layers with a SiLU nonlinearity) before being added to the projected input features.

Backbone. Input features (concatenated continuous and one-hot-encoded categorical variables) are first linearly projected to the embedding dimension $d_{\text{emb}} = 256$ and summed with the timestep embedding. The result is passed through an MLP trunk of 5 layers with hidden width 796 and ReLU activations, followed by a final linear layer projecting back to the input dimensionality. The model therefore predicts a full reconstruction of the (noised) input feature vector at each timestep.

F.2 TRAINING HYPERPARAMETERS

All models are trained with $T = 500$ diffusion timesteps. Table 6 summarizes the training configuration shared across baselines; paradigm-specific parameters (e.g., masking schedules for TabDiff, transition matrices for TabDDPM) are set to their respective default/recommended values from the original works.

Table 6: Training hyperparameters used for all baselines.

Hyperparameter	**Value**
Diffusion timesteps T	500
Embedding dimension d_{emb}	256
MLP layers	5
MLP hidden width	796
Activation	ReLU
Batch size	1024
Epochs	1000
Optimizer	Adam
Learning rate	1×10^{-3}
Weight decay	0
LR warmup epochs	150
LR schedule	Linear warmup, then linear decay
Max variance $\sigma_{\max}$	100

Learning rate schedule. The learning rate follows a linear warmup for the first 150 epochs, scaling linearly from 0 to the base learning rate as $\eta(e) = \eta_0 \cdot \frac{e}{150}$ for $e < 150$. After warmup, the learning rate decays linearly to 0 over the remaining epochs, $\eta(e) = \eta_0 \cdot \max\big(0,\ \frac{E-e}{E-150}\big)$, where E is the total number of training epochs.

F.3 TRAINING OBJECTIVE

For consistency across paradigms, all baselines are trained using the EDM loss formulation Karras et al. (2022), adapted per baseline as follows:

Base loss. By default, all models, including RHYTHM and the vanilla baseline, minimise an EDM-weighted denoising loss over continuous features and cross entropy loss for the categorical features, with the network trained to predict the denoised sample under the EDM preconditioning and loss-weighting scheme.

Mixed-noise models. For models combining discrete and continuous diffusion processes (TabDiff, TabDDPM), we additionally include a separate discrete loss term for the categorical features: a masked-token cross-entropy loss for TabDiff and a multinomial KL loss for TabDDPM, which is then added to the continuous EDM loss.

Learnable-schedule models. For models with learned noise schedules (TabDiff, CDCD, CDTD), we further add a time-warping loss that jointly trains the noise schedule parameterization alongside the denoiser, following the respective formulations in the original works.

TabSyn (autoencoder) TabSyn's autoencoder maps a tabular input $x_0 \in \mathbb{R}^D$ into a continuous latent space $z_0 \in \mathbb{R}^{d_{\text{lat}}}$, where $d_{\text{lat}} = d_{\text{num}} + \frac{1}{2}\sum_i K_i$, where d_{num} is the dimension of the numerical features and K_i is the cardinality of the ith categorical feature. Both encoder and decoder comprise 2-layer MLPs with a hidden dimension of 64, BatchNorm1d, and ReLU activations. The autoencoder is trained using a composite loss combining MSE over continuous features and Cross-Entropy over categorical chunks:

$$\mathcal{L}_{\text{AE}} = \left\|\hat{x}_0^{\text{cont}} - x_0^{\text{cont}}\right\|_2^2 + \sum_i \text{CE}\left(\hat{x}_0^{(i)}, x_0^{(i)}\right). \tag{29}$$

TabSyn (latent diffusion model) Once trained, the autoencoder weights are frozen, and a 5-layer MLP diffusion backbone is trained strictly on the extracted latent representations $z_0 = \text{Encoder}(x_0)$ using the continuous EDM objective.

F.4 Inference-Time Guidance Across Baseline Paradigms

While continuous methods (RHYTHM, vanilla, CDCD, and CDTD) support standard guidance (25), models operating in latent or discrete spaces require adaptations.

Latent-Space Guidance (TabSyn). Recall that TabSyn uses a continuous latent space $z_0 \in \mathbb{R}^{d_{\text{lat}}}$. Since constraint losses $\mathcal{L}_{\text{guide}}$ are defined in the data-space values and not latent vectors, guidance cannot be computed directly on the noisy latent z_t. Instead, at each step t, the predicted clean latent state $\hat{z}_0(z_t)$ is mapped back to the feature space via the pre-trained decoder, i.e., $\hat{x}_0 = \text{Decoder}(\hat{z}_0(z_t))$ following (He et al., 2024):

$$\hat{x}_0 = \text{Decoder}\left(\frac{z_t - \sqrt{(1-\bar{\alpha}_t)}\epsilon_\theta(z_t, t)}{\sqrt{\bar{\alpha}_t}}\right) \tag{30}$$

$$g_t = \nabla_{z_t}\mathcal{L}_{\text{guide}}(\hat{x}_0, c) \quad \text{(see Appendix C.4)} \tag{31}$$

$$z_t \leftarrow z_t - \eta \cdot g_t \tag{32}$$

where η is the guidance scale.

Discrete Logit-Steering (TabDDPM and TabDiff). Since the discrete sampling operations ($x_t \sim \text{Cat}(\cdot)$) are non-differentiable, exact gradient flow $\nabla_{x_t}\mathcal{L}_{\text{guide}}$ breaks down for categorical features. To enable guidance, we implement discrete logit-steering. For continuous features, we can apply the standard guidance (25): $x_{t-1}^{\text{cont}} \leftarrow x_{t-1}'^{\text{cont}} - \eta \cdot \nabla_{x_t^{\text{cont}}}\mathcal{L}_{\text{guide}}(\hat{x}_0, c)$. For the ith categorical feature, we apply guidance on the discrete category logits (before softmax) $\hat{l}_0^{(i)}$ are perturbed along the gradient of the guidance loss evaluated over the predicted probability distribution:

$$\hat{l}_0^{(i)} \leftarrow \hat{l}_0^{(i)} - \eta \cdot \nabla_{\hat{l}_0^{(i)}}\mathcal{L}_{\text{guide}}\Big(\text{softmax}(\hat{l}_0^{(i)}), c\Big) \tag{33}$$

The perturbed logits are then plugged into the reverse posterior transitions (21) or (23) to sample the discrete states. While logit-steering provides heuristic control, it lacks the formal convergence guarantees of standard training-free guidance.

Table 7: Inference-time guidance configurations and empirically selected guidance scales (η).

Model Class / Baseline	**Guidance Mechanism**	**Selected Scale (η)**
RHYTHM (Aligned; Ours)	Continuous (data-space)	0.01
Vanilla (Unaligned)	Continuous (data-space)	0.01
CDCD	Continuous (data-space)	0.01
CDTD	Continuous (data-space)	0.01
TabDiff	Continuous (data-space) + Logit-steering	0.01
TabDDPM	Continuous (data-space) + Logit-steering	0.01
TabSyn	Continuous (latent-space)	0.10

Guidance Hyperparameters Across Paradigms. To ensure a fair comparison, we perform a grid search over guidance scale values $\eta \in \{0.01, 0.1, 1.0, 2.0\}$ for each baseline and constraint type. We select the optimal scale for each method that maximizes overall sample quality (resemblance, coverage, and utility) among samples satisfying the given constraints. Table 7 summarises the guidance mechanisms, gradient target tensors, and final selected guidance scales (η) determined via grid search.

Table 8: Summary of evaluation metrics.

Axis	Metric	Description
Resemblance	KSComplement / TVComplement	Marginal (per-column) similarity
	Column Pair Trends	Pairwise correlation similarity
Diversity	Coverage	Real-data support covered by synthetic data
Downstream Utility	Accuracy	TSTR classification accuracy
	F1 (macro)	TSTR macro-averaged F1 score

G Evaluation Metrics and Downstream Task Configurations

G.1 Distributional Resemblance and Diversity

We use the SDMetrics library (Dat, 2026) to compute distributional resemblance and diversity metrics via its QualityReport module, augmented with the Coverage property. All metrics are computed between the synthetic data and a held-out real test set to guard against trivial memorisation.

Column Shapes. Marginal distributional similarity is measured per-column and averaged: KSComplement (one minus the Kolmogorov–Smirnov statistic) for continuous columns, and TVComplement (one minus total variation distance) for categorical columns.

Column Pair Trends. Bivariate dependency structure is assessed by comparing pairwise column correlations (Pearson correlation for continuous–continuous pairs, contingency-based association for pairs involving categorical columns) between real and synthetic data, aggregated into a single trend-similarity score.

Coverage. We additionally report the Coverage property, which quantifies the fraction of the real data's support (value ranges for continuous columns, categories for categorical columns) that is represented in the synthetic data, serving as a proxy for sample diversity and mode coverage.

All three scores are normalized to $[0, 1]$, with higher values indicating better fidelity. In the degenerate case where a model produces no valid synthetic samples, all resemblance and diversity metrics are assigned a score of 0.

G.2 Downstream Utility (Train-on-Synthetic, Test-on-Real)

We evaluate downstream utility using a train-on-synthetic, test-on-real (TSTR) protocol: a classifier is trained exclusively on the synthetic data and evaluated on the real, held-out test set. This directly measures whether the synthetic data preserves the predictive signal needed for practical downstream use. All datasets used in our evaluation involve classification targets (binary or multiclass); we did not include regression benchmarks in this study.

Binary classification. A Decision Tree classifier is fit on the synthetic data and evaluated on the real test set, reporting Accuracy and macro-averaged F1.

Multiclass classification. An MLP classifier (single hidden layer, 128 units, trained for up to 300 iterations) is fit on the synthetic data and evaluated on the real test set, reporting Accuracy and macro-averaged F1.

Failure case. If a model fails to produce any valid synthetic samples, downstream utility is assigned the worst-case score (Accuracy $= 0$, F1 $= 0$).

Table 8 summarizes the metrics used for each evaluation axis.

Table 9: Per-dataset constraints used for conditional generation evaluation. We also report the percentage of training samples matching the constraint.

Dataset	Range Constraint	Categorical Constraint	% Satisfying Range	% Satisfying Category	% Satisfying Range AND Category
Adult	$25 \leq$ age ≤ 45	education $\in$ {Bachelors}	53.27	16.39	10.60
Bean	$30{,}000 \leq$ Area $\leq 60{,}000$	Class $\in$ {SEKER, CALI}	64.61	27.45	15.56
Churn	$30 \leq$ Age ≤ 50	Geography $\in$ {Germany}	71.25	25.40	18.28
Default	$50{,}000 \leq$ LIMIT_BAL $\leq 200{,}000$	EDUCATION $\in \{1, 3\}$	53.58	51.66	26.43
Gesture	$-0.01 \leq$ vel. left hand (x) ≤ 0.01	phase $\in$ {D, H, R}	89.01	48.86	45.59
Intrusion	$0 \leq$ dst_host_count ≤ 150	protocol_type $\in$ {udp, icmp}	32.48	18.43	5.40
Letter	$3 \leq$ width ≤ 5	lettr $\in$ {A, B, C, D}	50.32	15.45	8.47
Magic	$0 \leq$ alpha ≤ 30	class $\in$ {h}	62.88	35.03	11.99
Shoppers	$0 \leq$ Administrative_Duration ≤ 500	VisitorType $\in$ {New_Visitor}	97.31	13.55	13.32

G.3 Constraint Satisfaction

For conditional generation, we additionally evaluate constraint satisfaction: the rate at which generated samples adhere to the conditioning signal (e.g., the specified target class or feature constraints), reported separately from the unconditional resemblance, diversity, and utility metrics above.

Table 9 lists the per-dataset constraints used to evaluate conditional generation. Each constraint combines a *range* restriction on a continuous feature with a *categorical* restriction on a discrete feature; a generated (or real) sample satisfies the constraint only if it meets both conditions jointly.

For each dataset, we generate samples conditioned on the corresponding constraint and report the fraction of generated samples that satisfy it, compared against the base rate of constraint satisfaction in the real training data (Table 9) as a reference for how far the constraint departs from the learned distribution. The remaining quality and utility scores are computed only using the samples that satisfy the constraint.

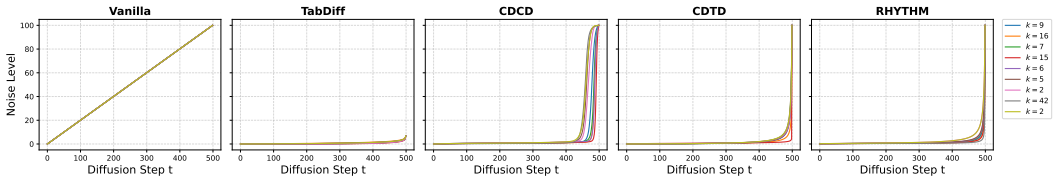


Figure 5: Noise-to-signal ratios of learned schedules: Adult.

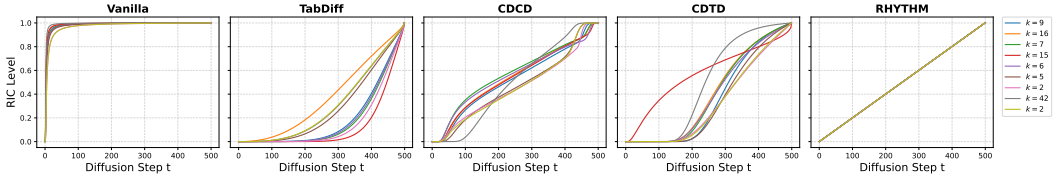


Figure 6: RICs of learned schedules: Adult

H Additional Experiments and Results

We provide extended empirical analyses of RHYTHM, including noise schedule and RIC trajectory plots against learned baselines, training loss dynamics across reference schedules and cardinality estimation strategies, qualitative synthetic data visualisations, and dataset-wise performance results.

H.1 Learned Noise Schedules vs RHYTHM

To further investigate how analytical corruption alignment compares to learned schedule optimisation, we compare the noise-to-signal ratio $\sqrt{(1-\bar{\alpha}_t)/\bar{\alpha}_t}$ and corresponding RIC trajectories of RHYTHM against both the uncorrected Vanilla model and schedule-learning approaches (TabDiff, CDCD, CDTD). Figures 5 and 6 present these trajectories on the Adult dataset, while Figures 7 and 8 showcase results on the Churn dataset.

We observe that methods utilising continuous-noise formulations (e.g., CDTD in fig.5 and both CDCD and CDTD in fig.7) produce noise schedules that exhibit a striking visual resemblance to the RHYTHM's analytically derived schedules. This strongly suggests that empirical schedule-learning methods implicitly attempt to discover the same corruption-aligned dynamics that RHYTHM derives explicitly; without requiring any neural network modules or extra loss terms during training.

Analyzing the corresponding RIC curves reveals the limitations of empirical schedule optimization. On moderate-cardinality datasets such as Adult ($\max K_u = 42$), learned schedule methods manage to approximate linear RIC growth reasonably well. However, when evaluated on datasets with extreme cardinalities, these learned approaches result in diverging RICs across features (e.g., CDCD and CDTD in fig.8). In contrast, RHYTHM's analytical corruption alignment enforces exact, perfectly linear, and cardinality-invariant RIC trajectories across all features regardless of cardinality.

H.2 Training loss curves across RHYTHM settings

To validate our architectural design choices, fig. 9 illustrates the average training loss across diffusion steps $t \in [0, T]$ under varying reference RIC schedules and continuous feature cardinality estimates K_{eff}.

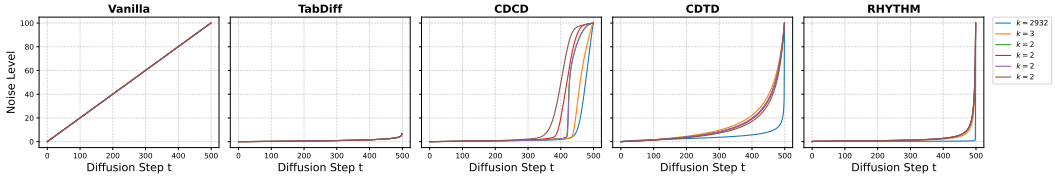


Figure 7: Noise-to-signal ratios of learned schedules: Churn.

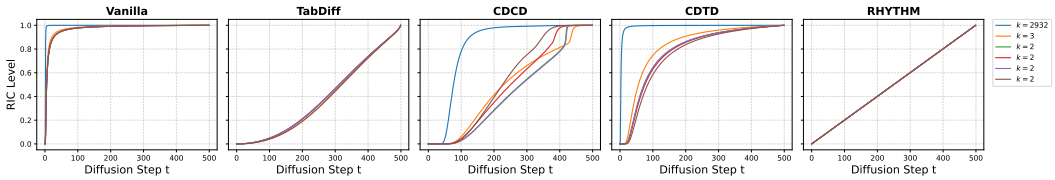
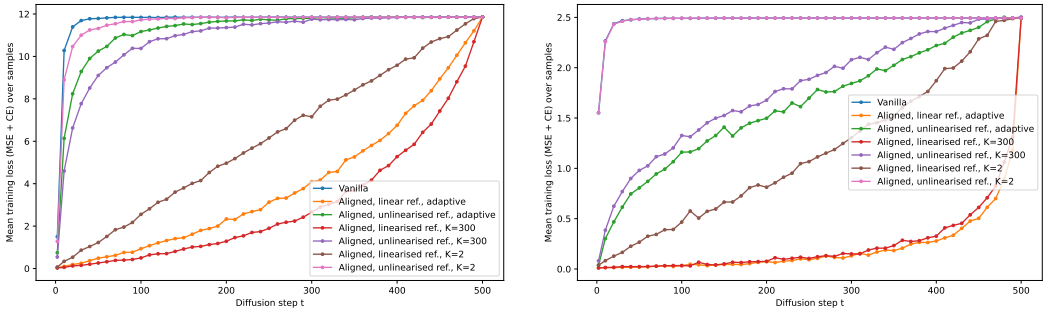


Figure 8: RICs of learned schedules: Churn



(a) Adult. (b) Gesture.

Figure 9: Average training loss across diffusion steps for different alignment configurations.

On both the Adult and Gesture, enforcing a perfectly linear reference RIC trajectory ($\bar{\rho}_{\text{ref}}(t) \propto t$) consistently yields a lower training loss across virtually all diffusion steps compared to the default. Alignment with this linear reference minimises the overall integrated loss over the entire diffusion process. While showcased on two representative datasets for conciseness, we find this result to hold in general across datasets.

Regarding the cardinality assignment for continuous variables, we observe that assigning a higher effective cardinality estimate ($K_{\text{eff}} = 300$) achieves strictly lower training loss than treating continuous features as low-cardinality entities ($K_{\text{eff}} = 2$). This observation is intuitive, since accurately representing a continuous real-valued feature requires a fine-grained discretisation space (i.e., a large number of effective "buckets").

H.3 Qualitative Results

H.3.1 Feature-wise correlations

To evaluate how well RHYTHM captures inter-feature dependencies, we analyse the absolute difference in feature correlation matrices between generated synthetic samples and the real test set distribution ($\Delta_{\text{Corr}} = |\mathbf{R}_{\text{real}} - \mathbf{R}_{\text{syn}}|$). For brevity, we focus our qualitative analysis on the four most challenging benchmarks: those with high categorical cardinalities (Intrusion, Churn) and those with high dimensional feature spaces (Gesture, Intrusion) (see Table 5 in Appendix E). The Adult dataset serves as a representative baseline for simpler benchmark datasets.

For unconditional generation, all baselines perform comparably well on simpler benchmarks such as Adult (see fig. 10). However, as dataset complexity increases, baseline performance begins to degrade. On high-cardinality and high-dimensional datasets (Gesture, Churn, Intrusion), schedule-learning continuous models (CDCD, CDTD) and strong mixed-noise baselines (TabDiff) exhibit noticeable correlation distortion, struggling to capture feature dependencies (see figs. 11,12,13). In contrast, RHYTHM maintains consistent cross-dataset robustness, maintaining low absolute correlation errors across scenarios.

We also stress-test models by imposing range and categorical constraints at inference time. We filter the generated outputs to include only those satisfying all criteria before computing correlation error against the filtered real test set. As expected, imposing joint constraints increases task difficulty, resulting in a general increase in correlation error across all frameworks. However, baseline degradation is more pronounced. On Gesture, Vanilla fails to produce a single valid sample satisfying the

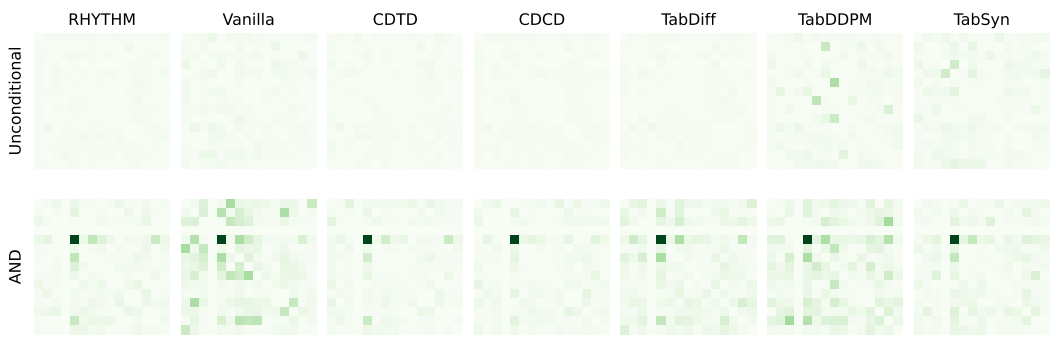


Figure 10: Adult: absolute feature correlation differences between real and synthetic data (lighter is better).

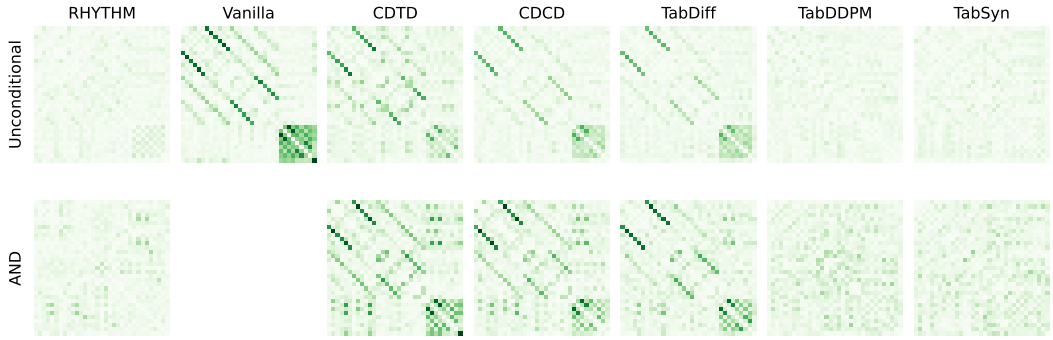


Figure 11: Gesture: absolute feature correlation differences between real and synthetic data (lighter is better). Vanilla fails to produce any valid samples under conjunctive (AND) constraints.

joint constraints, illustrating that its unaligned base model fails to learn a viable underlying distribution of real samples. The schedule-learning approaches (CDCD, CDTD, TabDiff) sometimes show larger correlation errors (fig. 11) compared to RHYTHM. By combining continuous-space guidance with exact analytical schedule alignment, RHYTHM is able to learn the underlying (unconditional) distributions well, which in turn results in good performance under constrained generation.

H.3.2 MARGINAL DISTRIBUTIONS

Figures 14–21 show the marginal distributions for a representative continuous and categorical feature across Adult, Gesture, Churn, and Intrusion under unconditional generation and conjunctive constraints.

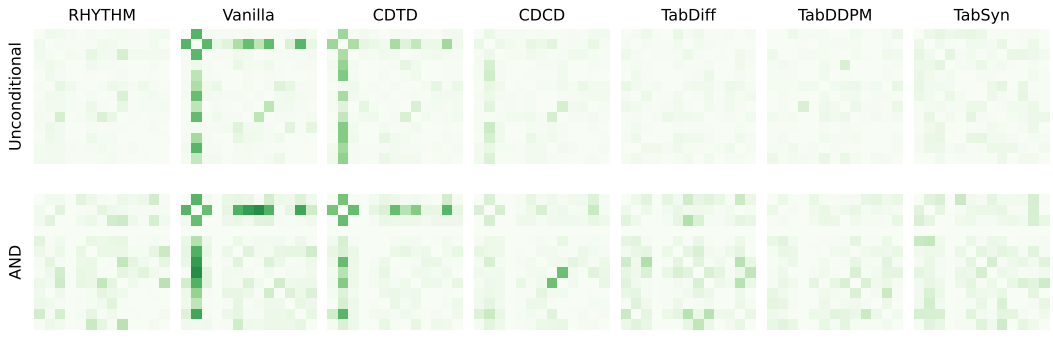


Figure 12: Churn: absolute feature correlation differences between real and synthetic data (lighter is better).

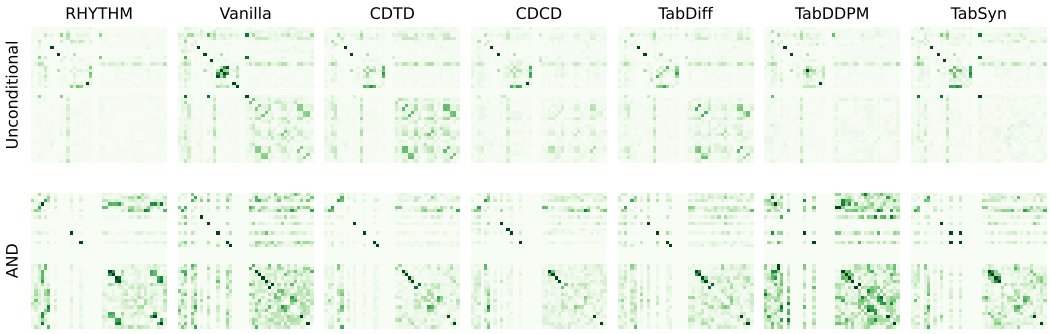


Figure 13: Intrusion: absolute feature correlation differences between real and synthetic data (lighter is better).

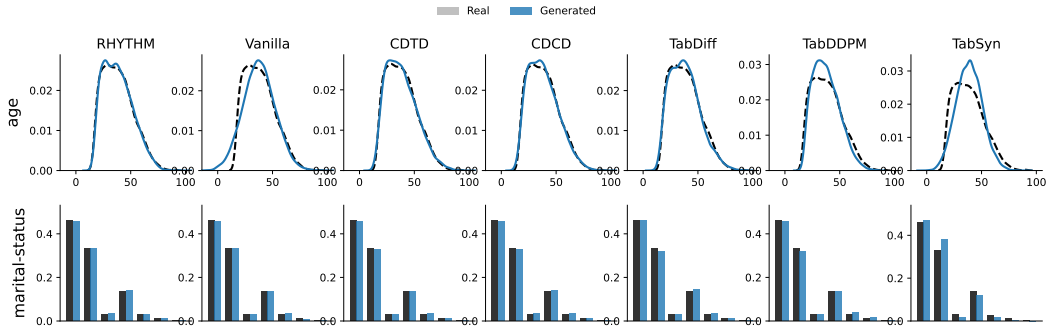


Figure 14: Adult: Feature distributions (unconditional).

On simpler datasets such as Adult (figs. 14 and 15), all evaluated baselines, capture the marginal probability densities and class frequencies reasonably well. However, baseline fidelity degrades under harder datasets. On high-cardinality and high-dimensional datasets (Gesture, Churn, Intrusion; figs. 16, 18, and 20), Vanilla fails to match feature distributions or has severe categorical class imbalances. Even strong schedule-learning baselines (CDTD, CDCD) and mixed-noise models (TabDiff) exhibit noticeable density distortion and category frequency mismatches on Gesture and Intrusion.

Under conjunctive constraints at inference time (figs. 17, 19, and 21), the performance gap widens significantly. CDCD and CDTD show severe class imbalances on Gesture and Intrusion.

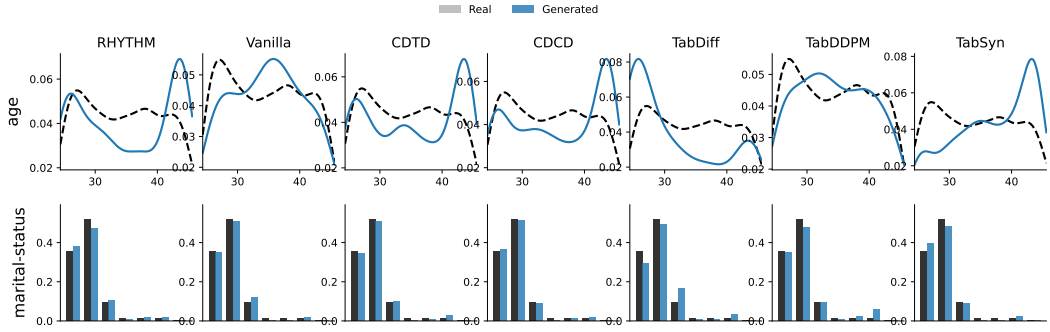


Figure 15: Adult: Feature distributions (conjunction).

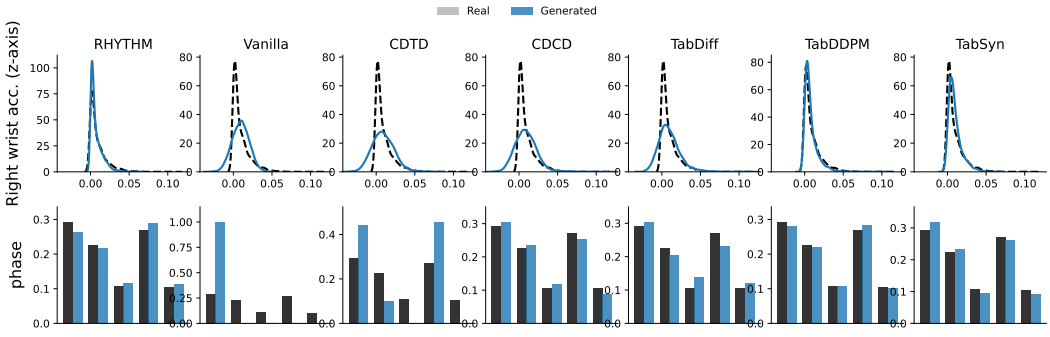


Figure 16: Gesture: Feature distributions (unconditional).

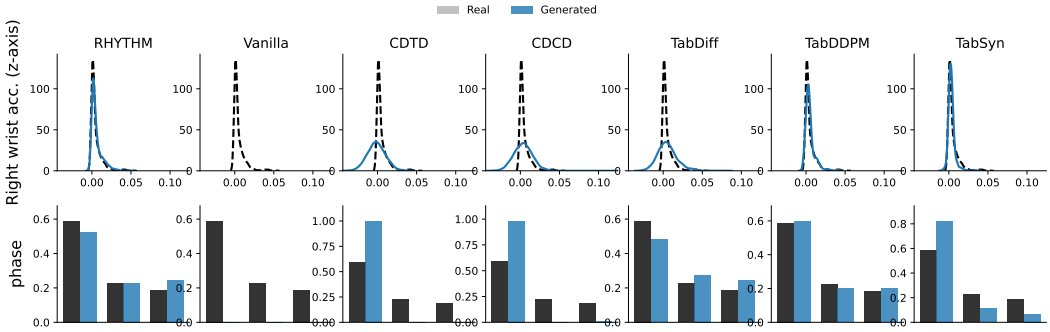


Figure 17: Gesture: Feature distributions (conjunctions).

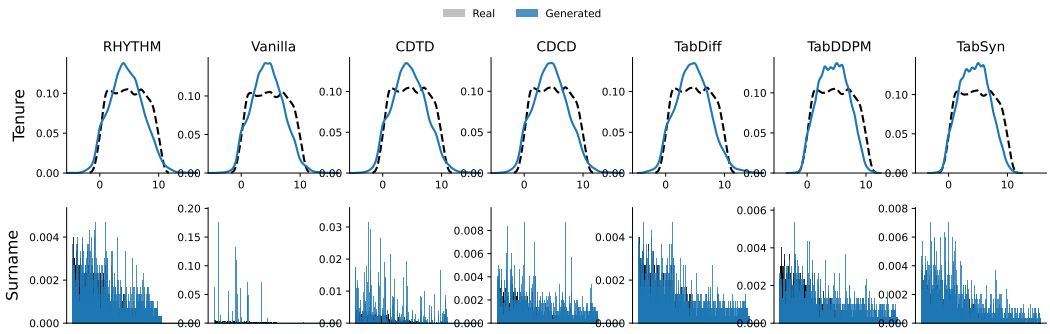


Figure 18: Churn: Feature distributions (unconditional).

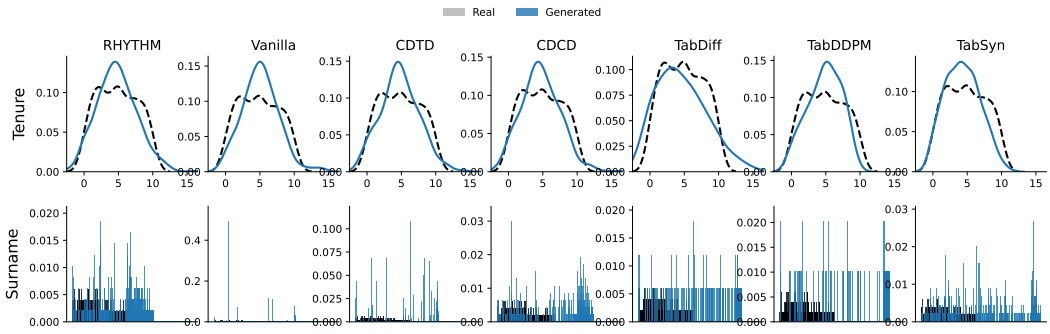


Figure 19: Churn: Feature distributions (conjunctions).

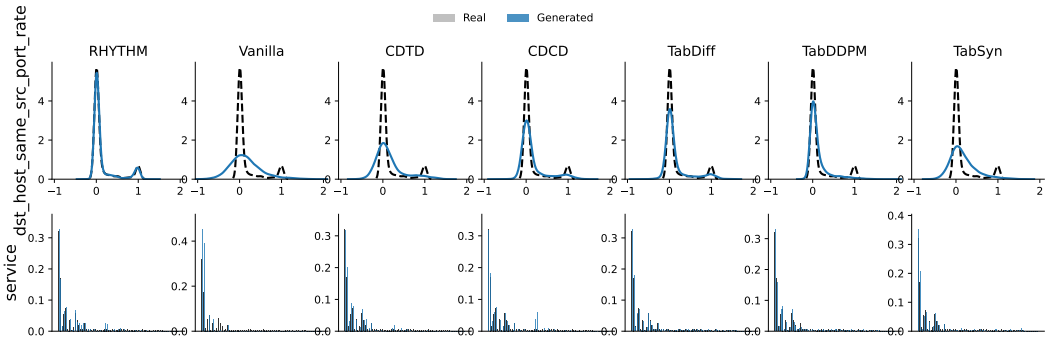


Figure 20: Intrusion: Feature distributions (unconditional).

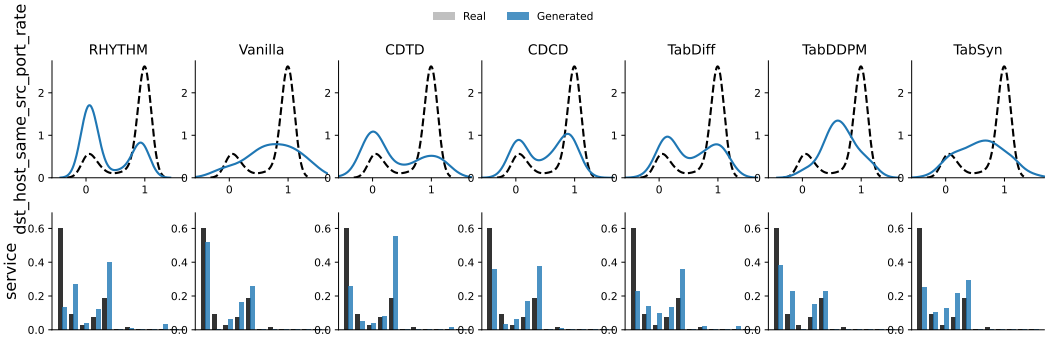


Figure 21: Intrusion: Feature distributions (conjunctions).

H.4 Per-dataset results

H.4.1 Unconditional Generation

Table 10 details the per-dataset performance across all tabular benchmarks under unconditional generation. Analyzing performance across specific metric categories highlights RHYTHM's strength in capturing complex multi-column structures while preserving high sample utility.

Since RHYTHM is explicitly designed to synchronise noise schedules across features, its primary objective is to capture joint feature interactions. Empirically, RHYTHM achieves the best or second-best Column Pair Trends score on 7 out of the 9 evaluated datasets. Even on datasets where it does not occupy the top two spots (e.g., Default, Gesture), its absolute score remains very high (≥ 0.97), indicating that performance differences among top models in this regime are negligible. In contrast, on high-cardinality datasets like Churn, where joint feature modelling is difficult, the uncorrected Vanilla baseline collapses completely (Column Pair Trends drops to 0.076), whereas RHYTHM reliably maintains strong pairwise correlation fidelity.

In downstream machine learning utility, RHYTHM ranks first or second in Accuracy on 5 of the 9 datasets and in F1 score on 6 of the 9 datasets. Its superiority is most pronounced on complex, high-dimensional datasets such as Gesture (achieving an F1 of 0.491 versus 0.091 for Vanilla) and Intrusion (0.981 versus 0.821 for Vanilla). These results confirm that analytical corruption alignment is essential for continuous diffusion backbones to generate downstream-usable synthetic data.

Regarding univariate marginal distributions, RHYTHM achieves the best or second best performance on 6 out of 9 datasets. Similar to the pairwise trend results, even when RHYTHM falls outside the top two positions, its absolute Column Shapes score remains high across all datasets (≥ 0.877).

The primary area where RHYTHM has lower relative performance is sample diversity (coverage), particularly on Default and Gesture. Nevertheless, its absolute coverage scores remain robust across the majority of benchmarks.

Overall, the results demonstrate that RHYTHM successfully matches existing baselines across nearly all metrics. Importantly, it demonstrates the critical necessity of corruption alignment when compared to the unaligned Vanilla baseline.

H.4.2 CONDITIONAL GENERATION

Tables 11, 12, and 13 report the per-dataset performance under range, categorical, and conjunctive (range AND category) constraints, respectively.

When evaluating quality metrics exclusively on samples that satisfy the target constraints (Accuracy, F1, Column Pair Trends, Column Shapes, and Coverage), performance across methods is similar. However, evaluating sample quality in isolation masks a crucial practical bottleneck: the constraint match percentage (Match %). Methods such as TabDDPM, TabDiff, and the unaligned Vanilla continuous baseline exhibit severely lower match rates across most datasets. In practice, these methods require rejecting a vast majority of generated samples to compile a valid target dataset, creating massive computational inefficiency. In contrast, RHYTHM achieves consistently high match rates (frequently exceeding 90%–99%) while maintaining competitive performance scores against the best baselines across the other metrics.

On continuous range constraints (Table 11), TabDDPM and Vanilla frequently drop below 70% match fidelity (e.g., 54.5% on Adult and 40.6% on Default for Vanilla). In contrast, continuous diffusion architectures with learned or aligned noise schedules, including CDCD, CDTD, and RHYTHM, demonstrate better constraint adaptability, consistently exceeding 90% match rates while achieving competitive scores on the other metrics.

Under categorical (Table 12) and conjunctive constraints (Table 13), the performance gap widens drastically. The match rates for Vanilla, TabDDPM, and TabDiff collapse further, regularly dropping below 20%–35% on datasets like Adult, Letter, Gesture, Intrusion, and Shoppers. Although the Vanilla baseline shares a continuous backbone with RHYTHM, its failure under conditioning demonstrates that unaligned noise schedules only capture superficial feature correlations. When stress-tested by pushing generation toward constrained sub-spaces, its underlying joint distribution breaks down, leading to significantly lower constraint matches (e.g., 0% match rate on Gesture).

By enforcing analytical corruption alignment across mixed feature spaces, RHYTHM dynamically balances noise schedules between continuous and categorical channels. This allows it to maintain performance quality while also frequently achieving a constraint match rate ¿90%. In summary, only corruption-aligned models like RHYTHM or continuous-noise models that are effective at learning the underlying unconditional distribution well (CDTD, CDCD, TabSyn) combine high conditional generation accuracy with high constraint satisfaction rates.

Table 10: Results on unconditional generation (all datasets)

Dataset	Method	Accuracy	Column Pair Trends	Column Shapes	Coverage	F1
	CDCD	$0.801_{0.004}$	$0.884_{0.001}$	$0.895_{0.001}$	$0.975_{0.016}$	$0.727_{0.005}$
	CDTD	$0.792_{0.013}$	$0.881_{0.002}$	$0.892_{0.001}$	$0.976_{0.014}$	$0.717_{0.020}$
	TabDDPM	$0.762_{0.014}$	$0.873_{0.017}$	$0.898_{0.000}$	$0.945_{0.007}$	$0.687_{0.023}$
adult	TabDiff	$0.799_{0.004}$	$0.891_{0.012}$	$\mathbf{0.900_{0.001}}$	$\mathbf{0.981_{0.014}}$	$\mathbf{0.733_{0.004}}$
	TabSyn	$0.741_{0.025}$	$0.807_{0.009}$	$0.844_{0.001}$	$0.880_{0.004}$	$0.670_{0.013}$
	Vanilla	$\mathbf{0.802_{0.007}}$	$0.856_{0.012}$	$0.865_{0.001}$	$0.913_{0.002}$	$0.727_{0.012}$
	RHYTHM	$0.798_{0.008}$	$\mathbf{0.930_{0.002}}$	$0.895_{0.001}$	$0.976_{0.012}$	$0.731_{0.008}$
	CDCD	$0.922_{0.002}$	$0.948_{0.009}$	$\mathbf{0.975_{0.002}}$	$0.928_{0.016}$	$0.935_{0.002}$
	CDTD	$0.921_{0.002}$	$0.954_{0.007}$	$0.972_{0.003}$	$0.922_{0.019}$	$0.935_{0.002}$
	TabDDPM	$0.913_{0.003}$	$0.946_{0.005}$	$0.953_{0.003}$	$0.915_{0.024}$	$0.925_{0.004}$
bean	TabDiff	$\mathbf{0.926_{0.002}}$	$0.953_{0.005}$	$0.974_{0.002}$	$\mathbf{0.937_{0.019}}$	$0.938_{0.002}$
	TabSyn	$0.916_{0.002}$	$0.937_{0.011}$	$0.956_{0.004}$	$0.899_{0.011}$	$0.929_{0.002}$
	Vanilla	$0.917_{0.004}$	$0.941_{0.005}$	$0.967_{0.002}$	$0.920_{0.014}$	$0.930_{0.003}$
	RHYTHM	$0.926_{0.003}$	$\mathbf{0.966_{0.004}}$	$0.972_{0.002}$	$0.918_{0.019}$	$\mathbf{0.938_{0.002}}$
	CDCD	$0.788_{0.004}$	$0.230_{0.005}$	$0.890_{0.001}$	$0.945_{0.014}$	$0.663_{0.007}$
	CDTD	$0.786_{0.009}$	$0.126_{0.001}$	$0.868_{0.001}$	$0.901_{0.005}$	$0.623_{0.013}$
	TabDDPM	$0.695_{0.071}$	$\mathbf{0.287_{0.010}}$	$0.876_{0.001}$	$0.938_{0.020}$	$0.558_{0.047}$
churn	TabDiff	$\mathbf{0.804_{0.011}}$	$0.280_{0.006}$	$0.896_{0.001}$	$\mathbf{0.959_{0.001}}$	$\mathbf{0.667_{0.014}}$
	TabSyn	$0.769_{0.008}$	$0.283_{0.009}$	$0.864_{0.002}$	$0.943_{0.014}$	$0.644_{0.012}$
	Vanilla	$0.637_{0.019}$	$0.076_{0.001}$	$0.829_{0.001}$	$0.889_{0.005}$	$0.499_{0.007}$
	RHYTHM	$0.785_{0.007}$	$0.284_{0.013}$	$\mathbf{0.899_{0.002}}$	$0.956_{0.003}$	$0.666_{0.012}$
	CDCD	$0.725_{0.016}$	$\mathbf{0.988_{0.001}}$	$0.884_{0.001}$	$\mathbf{0.880_{0.022}}$	$0.612_{0.004}$
	CDTD	$0.726_{0.008}$	$0.986_{0.001}$	$0.880_{0.001}$	$0.866_{0.024}$	$0.612_{0.009}$
	TabDDPM	$0.658_{0.009}$	$0.981_{0.002}$	$0.885_{0.002}$	$0.827_{0.026}$	$0.568_{0.008}$
default	TabDiff	$\mathbf{0.731_{0.012}}$	$0.982_{0.002}$	$0.874_{0.001}$	$0.866_{0.017}$	$\mathbf{0.617_{0.015}}$
	TabSyn	$0.712_{0.007}$	$0.985_{0.001}$	$0.863_{0.001}$	$0.840_{0.014}$	$0.614_{0.005}$
	Vanilla	$0.710_{0.018}$	$0.961_{0.002}$	$0.877_{0.001}$	$0.858_{0.040}$	$0.602_{0.010}$
	RHYTHM	$0.710_{0.008}$	$0.976_{0.002}$	$\mathbf{0.902_{0.002}}$	$0.726_{0.020}$	$0.598_{0.004}$
	CDCD	$0.427_{0.038}$	$0.806_{0.005}$	$0.773_{0.000}$	$0.889_{0.016}$	$0.389_{0.013}$
	CDTD	$0.438_{0.002}$	$0.784_{0.009}$	$0.764_{0.001}$	$0.740_{0.047}$	$0.253_{0.002}$
	TabDDPM	$0.389_{0.012}$	$\mathbf{0.978_{0.004}}$	$0.892_{0.002}$	$0.755_{0.038}$	$0.313_{0.009}$
gesture	TabDiff	$0.442_{0.037}$	$0.806_{0.013}$	$0.779_{0.001}$	$\mathbf{0.906_{0.041}}$	$0.391_{0.017}$
	TabSyn	$0.433_{0.019}$	$0.973_{0.001}$	$0.867_{0.003}$	$0.576_{0.021}$	$0.359_{0.021}$
	Vanilla	$0.293_{0.000}$	$0.647_{0.003}$	$0.779_{0.001}$	$0.384_{0.004}$	$0.091_{0.000}$
	RHYTHM	$\mathbf{0.539_{0.010}}$	$0.971_{0.004}$	$\mathbf{0.935_{0.002}}$	$0.545_{0.016}$	$\mathbf{0.491_{0.011}}$
	CDCD	$0.966_{0.003}$	$0.538_{0.003}$	$0.766_{0.001}$	$0.919_{0.014}$	$0.966_{0.003}$
	CDTD	$0.962_{0.002}$	$0.496_{0.005}$	$0.761_{0.000}$	$0.915_{0.024}$	$0.962_{0.002}$
	TabDDPM	$0.935_{0.005}$	$0.598_{0.030}$	$0.794_{0.001}$	$0.837_{0.034}$	$0.935_{0.005}$
intrusion	TabDiff	$0.976_{0.003}$	$0.544_{0.016}$	$0.776_{0.000}$	$\mathbf{0.930_{0.013}}$	$0.976_{0.003}$
	TabSyn	$0.943_{0.008}$	$0.594_{0.006}$	$0.763_{0.000}$	$0.827_{0.012}$	$0.942_{0.008}$
	Vanilla	$0.830_{0.088}$	$0.456_{0.001}$	$0.751_{0.001}$	$0.606_{0.005}$	$0.821_{0.099}$
	RHYTHM	$\mathbf{0.981_{0.004}}$	$\mathbf{0.628_{0.011}}$	$\mathbf{0.806_{0.001}}$	$0.884_{0.007}$	$\mathbf{0.981_{0.004}}$
	CDCD	$0.837_{0.011}$	$0.879_{0.010}$	$0.878_{0.003}$	$0.972_{0.007}$	$0.837_{0.010}$
	CDTD	$0.837_{0.005}$	$0.890_{0.005}$	$0.880_{0.003}$	$0.971_{0.013}$	$0.837_{0.005}$
	TabDDPM	$0.713_{0.016}$	$0.844_{0.017}$	$0.881_{0.003}$	$0.891_{0.008}$	$0.712_{0.016}$
letter	TabDiff	$0.798_{0.013}$	$0.844_{0.015}$	$\mathbf{0.883_{0.002}}$	$0.978_{0.007}$	$0.800_{0.013}$
	TabSyn	$0.678_{0.011}$	$0.831_{0.007}$	$0.881_{0.002}$	$0.915_{0.005}$	$0.676_{0.010}$
	Vanilla	$0.797_{0.005}$	$0.860_{0.005}$	$0.877_{0.003}$	$\mathbf{0.979_{0.006}}$	$0.799_{0.006}$
	RHYTHM	$\mathbf{0.858_{0.007}}$	$\mathbf{0.908_{0.006}}$	$0.877_{0.003}$	$0.952_{0.009}$	$\mathbf{0.858_{0.007}}$
	CDCD	$\mathbf{0.794_{0.005}}$	$0.895_{0.017}$	$0.945_{0.002}$	$0.978_{0.014}$	$\mathbf{0.774_{0.006}}$
	CDTD	$0.792_{0.008}$	$0.884_{0.015}$	$0.945_{0.003}$	$\mathbf{1.000_{0.001}}$	$0.771_{0.009}$
	TabDDPM	$0.735_{0.006}$	$0.886_{0.015}$	$0.925_{0.005}$	$0.943_{0.030}$	$0.713_{0.007}$
magic	TabDiff	$0.789_{0.007}$	$0.862_{0.011}$	$0.948_{0.002}$	$0.964_{0.014}$	$0.766_{0.007}$
	TabSyn	$0.759_{0.009}$	$0.814_{0.015}$	$0.905_{0.002}$	$0.943_{0.011}$	$0.718_{0.008}$
	Vanilla	$0.744_{0.007}$	$0.828_{0.009}$	$0.927_{0.002}$	$0.983_{0.009}$	$0.714_{0.007}$
	RHYTHM	$0.787_{0.004}$	$\mathbf{0.923_{0.013}}$	$\mathbf{0.952_{0.003}}$	$0.955_{0.010}$	$0.770_{0.005}$
	CDCD	$0.869_{0.007}$	$0.802_{0.054}$	$0.817_{0.001}$	$0.974_{0.009}$	$\mathbf{0.746_{0.013}}$
	CDTD	$\mathbf{0.872_{0.010}}$	$0.721_{0.022}$	$0.820_{0.001}$	$\mathbf{0.987_{0.008}}$	$0.746_{0.014}$
	TabDDPM	$0.841_{0.009}$	$0.884_{0.016}$	$0.807_{0.001}$	$0.945_{0.008}$	$0.700_{0.020}$
shoppers	TabDiff	$0.865_{0.004}$	$0.820_{0.044}$	$0.821_{0.003}$	$0.970_{0.017}$	$0.734_{0.012}$
	TabSyn	$0.863_{0.013}$	$0.852_{0.043}$	$0.803_{0.001}$	$0.958_{0.032}$	$0.721_{0.034}$
	Vanilla	$0.853_{0.009}$	$0.744_{0.017}$	$0.795_{0.002}$	$0.958_{0.014}$	$0.701_{0.013}$
	RHYTHM	$0.869_{0.005}$	$\mathbf{0.884_{0.003}}$	$\mathbf{0.821_{0.002}}$	$0.929_{0.037}$	$0.743_{0.014}$

Table 11: Results on Range constraints, per dataset.

Dataset	Method	Accuracy	Column Pair Trends	Column Shapes	Coverage	F1	Match %
	CDCD	$0.780_{0.006}$	$0.880_{0.030}$	$0.864_{0.001}$	$0.955_{0.016}$	$0.713_{0.009}$	$0.998_{0.001}$
	CDTD	$0.780_{0.005}$	$0.866_{0.027}$	$0.862_{0.001}$	$0.961_{0.008}$	$0.702_{0.014}$	$\mathbf{0.999_{0.000}}$
	TabDDPM	$0.745_{0.016}$	$0.854_{0.022}$	$\mathbf{0.894_{0.002}}$	$0.928_{0.007}$	$0.681_{0.014}$	$0.622_{0.007}$
adult	TabDiff	$0.782_{0.005}$	$0.846_{0.014}$	$0.868_{0.001}$	$\mathbf{0.966_{0.019}}$	$0.714_{0.011}$	$0.998_{0.000}$
	TabSyn	$0.750_{0.009}$	$0.813_{0.016}$	$0.824_{0.001}$	$0.868_{0.008}$	$0.670_{0.014}$	$0.912_{0.003}$
	Vanilla	$0.775_{0.026}$	$0.857_{0.014}$	$0.854_{0.003}$	$0.889_{0.009}$	$0.693_{0.023}$	$0.545_{0.008}$
	RHYTHM	$\mathbf{0.787_{0.009}}$	$\mathbf{0.904_{0.010}}$	$0.863_{0.001}$	$0.958_{0.010}$	$\mathbf{0.720_{0.012}}$	$0.986_{0.002}$
	CDCD	$0.894_{0.005}$	$0.931_{0.002}$	$0.904_{0.011}$	$0.907_{0.014}$	$0.835_{0.021}$	$0.884_{0.006}$
	CDTD	$0.896_{0.003}$	$0.925_{0.005}$	$0.901_{0.009}$	$0.907_{0.009}$	$0.835_{0.015}$	$0.889_{0.005}$
	TabDDPM	$0.888_{0.002}$	$\mathbf{0.943_{0.005}}$	$0.945_{0.005}$	$0.849_{0.013}$	$0.815_{0.025}$	$0.723_{0.004}$
bean	TabDiff	$0.898_{0.003}$	$0.906_{0.004}$	$0.870_{0.004}$	$\mathbf{0.941_{0.008}}$	$0.838_{0.006}$	$\mathbf{0.969_{0.005}}$
	TabSyn	$0.890_{0.003}$	$0.930_{0.008}$	$0.933_{0.003}$	$0.864_{0.008}$	$0.830_{0.015}$	$0.863_{0.005}$
	Vanilla	$0.891_{0.005}$	$0.935_{0.002}$	$\mathbf{0.951_{0.002}}$	$0.892_{0.012}$	$0.838_{0.011}$	$0.687_{0.012}$
	RHYTHM	$\mathbf{0.898_{0.003}}$	$0.933_{0.004}$	$0.882_{0.008}$	$0.897_{0.004}$	$\mathbf{0.844_{0.019}}$	$0.936_{0.003}$
	CDCD	$0.777_{0.015}$	$0.200_{0.007}$	$0.877_{0.001}$	$0.927_{0.013}$	$0.598_{0.011}$	$\mathbf{0.999_{0.001}}$
	CDTD	$0.790_{0.006}$	$0.117_{0.002}$	$0.858_{0.002}$	$0.907_{0.000}$	$0.546_{0.035}$	$0.877_{0.004}$
	TabDDPM	$0.729_{0.032}$	$0.228_{0.007}$	$0.865_{0.002}$	$0.925_{0.013}$	$0.541_{0.025}$	$0.801_{0.013}$
churn	TabDiff	$\mathbf{0.796_{0.012}}$	$0.239_{0.004}$	$0.879_{0.002}$	$\mathbf{0.955_{0.002}}$	$0.628_{0.018}$	$0.997_{0.002}$
	TabSyn	$0.764_{0.016}$	$\mathbf{0.254_{0.006}}$	$0.856_{0.001}$	$0.929_{0.011}$	$0.601_{0.016}$	$0.950_{0.005}$
	Vanilla	$0.700_{0.059}$	$0.071_{0.002}$	$0.822_{0.002}$	$0.899_{0.001}$	$0.509_{0.012}$	$0.704_{0.004}$
	RHYTHM	$0.780_{0.007}$	$0.235_{0.006}$	$\mathbf{0.883_{0.003}}$	$0.952_{0.003}$	$\mathbf{0.633_{0.013}}$	$0.995_{0.002}$
	CDCD	$0.698_{0.022}$	$\mathbf{0.988_{0.001}}$	$0.881_{0.001}$	$0.929_{0.013}$	$\mathbf{0.603_{0.020}}$	$0.995_{0.001}$
	CDTD	$0.694_{0.012}$	$0.982_{0.002}$	$0.875_{0.002}$	$0.920_{0.011}$	$0.592_{0.016}$	$\mathbf{0.999_{0.001}}$
	TabDDPM	$0.618_{0.025}$	$0.976_{0.003}$	$0.888_{0.002}$	$0.719_{0.021}$	$0.534_{0.015}$	$0.607_{0.005}$
default	TabDiff	$\mathbf{0.711_{0.028}}$	$0.986_{0.002}$	$0.872_{0.001}$	$\mathbf{0.942_{0.014}}$	$0.597_{0.020}$	$0.995_{0.002}$
	TabSyn	$0.694_{0.016}$	$0.985_{0.004}$	$0.860_{0.000}$	$0.852_{0.019}$	$0.603_{0.012}$	$0.909_{0.006}$
	Vanilla	$0.672_{0.013}$	$0.964_{0.003}$	$0.874_{0.002}$	$0.811_{0.018}$	$0.576_{0.010}$	$0.406_{0.006}$
	RHYTHM	$0.673_{0.010}$	$0.973_{0.003}$	$\mathbf{0.913_{0.002}}$	$0.813_{0.021}$	$0.576_{0.010}$	$0.957_{0.002}$
	CDCD	$0.397_{0.039}$	$0.791_{0.007}$	$0.757_{0.001}$	$0.919_{0.014}$	$0.379_{0.023}$	$0.986_{0.002}$
	CDTD	$0.438_{0.001}$	$0.734_{0.005}$	$0.748_{0.001}$	$0.674_{0.013}$	$0.252_{0.001}$	$0.885_{0.006}$
	TabDDPM	$0.401_{0.008}$	$0.973_{0.004}$	$0.879_{0.002}$	$0.790_{0.023}$	$0.314_{0.008}$	$0.943_{0.003}$
gesture	TabDiff	$0.440_{0.033}$	$0.801_{0.009}$	$0.762_{0.001}$	$\mathbf{0.943_{0.013}}$	$0.371_{0.008}$	$0.995_{0.002}$
	TabSyn	$0.438_{0.021}$	$\mathbf{0.973_{0.003}}$	$0.846_{0.004}$	$0.685_{0.012}$	$0.362_{0.020}$	$0.986_{0.003}$
	Vanilla	$0.274_{0.000}$	$0.648_{0.002}$	$0.758_{0.001}$	$0.491_{0.005}$	$0.086_{0.000}$	$0.870_{0.002}$
	RHYTHM	$\mathbf{0.535_{0.008}}$	$0.972_{0.003}$	$\mathbf{0.938_{0.001}}$	$0.655_{0.024}$	$\mathbf{0.479_{0.009}}$	$\mathbf{0.997_{0.001}}$
	CDCD	$0.931_{0.011}$	$0.466_{0.013}$	$0.730_{0.001}$	$\mathbf{0.902_{0.029}}$	$0.891_{0.013}$	$0.927_{0.004}$
	CDTD	$0.939_{0.006}$	$0.439_{0.018}$	$0.722_{0.001}$	$0.894_{0.020}$	$0.901_{0.006}$	$\mathbf{0.994_{0.001}}$
	TabDDPM	$0.893_{0.014}$	$0.555_{0.027}$	$\mathbf{0.758_{0.002}}$	$0.680_{0.026}$	$0.828_{0.025}$	$0.326_{0.010}$
intrusion	TabDiff	$0.954_{0.009}$	$0.420_{0.019}$	$0.686_{0.002}$	$0.812_{0.063}$	$0.927_{0.016}$	$0.992_{0.002}$
	TabSyn	$0.907_{0.018}$	$0.495_{0.006}$	$0.726_{0.001}$	$0.817_{0.021}$	$0.853_{0.024}$	$0.897_{0.006}$
	Vanilla	$0.764_{0.022}$	$0.358_{0.004}$	$0.720_{0.001}$	$0.609_{0.006}$	$0.713_{0.018}$	$0.423_{0.012}$
	RHYTHM	$\mathbf{0.970_{0.003}}$	$\mathbf{0.588_{0.035}}$	$0.745_{0.002}$	$0.819_{0.072}$	$\mathbf{0.954_{0.005}}$	$0.896_{0.005}$
	CDCD	$0.843_{0.009}$	$0.786_{0.013}$	$0.855_{0.003}$	$0.977_{0.005}$	$0.845_{0.009}$	$0.982_{0.002}$
	CDTD	$0.858_{0.006}$	$0.815_{0.020}$	$0.860_{0.003}$	$\mathbf{0.991_{0.005}}$	$0.858_{0.006}$	$0.990_{0.002}$
	TabDDPM	$0.722_{0.020}$	$0.751_{0.012}$	$\mathbf{0.886_{0.002}}$	$0.925_{0.012}$	$0.724_{0.020}$	$0.608_{0.011}$
letter	TabDiff	$0.824_{0.009}$	$0.764_{0.019}$	$0.860_{0.004}$	$0.987_{0.007}$	$0.825_{0.008}$	$\mathbf{0.997_{0.001}}$
	TabSyn	$0.714_{0.010}$	$0.730_{0.015}$	$0.806_{0.003}$	$0.904_{0.008}$	$0.705_{0.011}$	$0.878_{0.009}$
	Vanilla	$0.794_{0.007}$	$0.758_{0.012}$	$0.878_{0.004}$	$0.975_{0.015}$	$0.794_{0.008}$	$0.639_{0.009}$
	RHYTHM	$\mathbf{0.878_{0.006}}$	$\mathbf{0.824_{0.012}}$	$0.864_{0.003}$	$0.980_{0.007}$	$\mathbf{0.878_{0.006}}$	$0.997_{0.001}$
	CDCD	$\mathbf{0.813_{0.007}}$	$0.888_{0.023}$	$0.884_{0.004}$	$0.958_{0.017}$	$0.701_{0.010}$	$\mathbf{0.994_{0.001}}$
	CDTD	$0.807_{0.003}$	$0.897_{0.023}$	$0.884_{0.003}$	$0.951_{0.024}$	$0.686_{0.006}$	$0.965_{0.002}$
	TabDDPM	$0.768_{0.017}$	$0.886_{0.036}$	$\mathbf{0.906_{0.006}}$	$0.851_{0.023}$	$0.661_{0.018}$	$0.663_{0.002}$
magic	TabDiff	$0.808_{0.010}$	$0.880_{0.007}$	$0.889_{0.005}$	$\mathbf{0.961_{0.016}}$	$0.697_{0.016}$	$0.993_{0.001}$
	TabSyn	$0.808_{0.006}$	$0.849_{0.013}$	$0.882_{0.002}$	$0.905_{0.019}$	$0.657_{0.018}$	$0.832_{0.006}$
	Vanilla	$0.777_{0.015}$	$0.888_{0.011}$	$0.878_{0.003}$	$0.912_{0.005}$	$0.648_{0.012}$	$0.407_{0.012}$
	RHYTHM	$0.804_{0.014}$	$\mathbf{0.911_{0.016}}$	$0.885_{0.003}$	$0.958_{0.007}$	$\mathbf{0.705_{0.017}}$	$0.988_{0.002}$
	CDCD	$0.852_{0.030}$	$0.816_{0.028}$	$0.808_{0.002}$	$0.988_{0.007}$	$0.721_{0.029}$	$0.963_{0.004}$
	CDTD	$0.866_{0.011}$	$0.722_{0.020}$	$0.809_{0.001}$	$0.988_{0.012}$	$0.730_{0.034}$	$0.853_{0.006}$
	TabDDPM	$0.844_{0.015}$	$\mathbf{0.889_{0.009}}$	$0.794_{0.002}$	$0.970_{0.008}$	$0.704_{0.022}$	$0.874_{0.007}$
shoppers	TabDiff	$0.843_{0.037}$	$0.827_{0.052}$	$\mathbf{0.810_{0.002}}$	$\mathbf{0.990_{0.010}}$	$0.705_{0.031}$	$\mathbf{0.993_{0.001}}$
	TabSyn	$0.856_{0.008}$	$0.867_{0.032}$	$0.788_{0.002}$	$0.898_{0.016}$	$0.714_{0.027}$	$0.957_{0.005}$
	Vanilla	$0.862_{0.003}$	$0.745_{0.018}$	$0.776_{0.003}$	$0.943_{0.025}$	$0.701_{0.016}$	$0.738_{0.005}$
	RHYTHM	$\mathbf{0.871_{0.011}}$	$0.885_{0.002}$	$0.808_{0.003}$	$0.959_{0.023}$	$\mathbf{0.736_{0.023}}$	$0.871_{0.005}$

Table 12: Results on Category constraints, per dataset. Score of 1.0 on Magic for F1 and accuracy is because the downstream task collapses to a single class classification problem.

Dataset	Method	Accuracy	Column Pair Trends	Column Shapes	Coverage	F1	Match %
	CDCD	$0.738_{0.024}$	$\mathbf{0.919_{0.005}}$	$0.844_{0.004}$	$\mathbf{0.961_{0.009}}$	$0.731_{0.024}$	$\mathbf{0.996_{0.001}}$
	CDTD	$0.705_{0.039}$	$0.914_{0.007}$	$0.821_{0.002}$	$0.948_{0.006}$	$0.685_{0.042}$	$0.961_{0.003}$
	TabDDPM	$0.702_{0.026}$	$0.875_{0.008}$	$\mathbf{0.848_{0.002}}$	$0.855_{0.010}$	$0.686_{0.033}$	$0.192_{0.010}$
adult	TabDiff	$0.722_{0.022}$	$0.873_{0.008}$	$0.831_{0.003}$	$0.945_{0.014}$	$0.713_{0.024}$	$0.389_{0.008}$
	TabSyn	$0.705_{0.023}$	$0.857_{0.011}$	$0.836_{0.002}$	$0.852_{0.020}$	$0.684_{0.028}$	$0.935_{0.004}$
	Vanilla	$\mathbf{0.750_{0.021}}$	$0.861_{0.008}$	$0.839_{0.007}$	$0.821_{0.015}$	$\mathbf{0.745_{0.023}}$	$0.188_{0.013}$
	RHYTHM	$0.714_{0.026}$	$0.890_{0.008}$	$0.843_{0.002}$	$0.950_{0.010}$	$0.711_{0.026}$	$0.934_{0.005}$
	CDCD	$0.997_{0.000}$	$0.953_{0.006}$	$0.926_{0.006}$	$0.894_{0.013}$	$0.997_{0.000}$	$0.986_{0.002}$
	CDTD	$0.996_{0.001}$	$\mathbf{0.955_{0.007}}$	$\mathbf{0.942_{0.004}}$	$0.890_{0.042}$	$0.996_{0.001}$	$0.728_{0.015}$
	TabDDPM	$0.996_{0.001}$	$0.943_{0.007}$	$0.910_{0.010}$	$0.857_{0.016}$	$0.996_{0.001}$	$0.355_{0.010}$
bean	TabDiff	$0.993_{0.002}$	$0.903_{0.006}$	$0.813_{0.003}$	$0.913_{0.009}$	$0.993_{0.002}$	$0.475_{0.020}$
	TabSyn	$\mathbf{0.997_{0.000}}$	$0.920_{0.009}$	$0.816_{0.014}$	$0.836_{0.021}$	$\mathbf{0.997_{0.000}}$	$0.948_{0.005}$
	Vanilla	$0.997_{0.001}$	$0.943_{0.006}$	$0.916_{0.014}$	$0.868_{0.013}$	$0.997_{0.001}$	$0.355_{0.016}$
	RHYTHM	$0.996_{0.001}$	$0.910_{0.004}$	$0.845_{0.005}$	$\mathbf{0.925_{0.011}}$	$0.996_{0.001}$	$\mathbf{0.990_{0.004}}$
	CDCD	$\mathbf{0.721_{0.015}}$	$0.076_{0.003}$	$0.873_{0.005}$	$0.907_{0.006}$	$\mathbf{0.664_{0.018}}$	$0.958_{0.012}$
	CDTD	$0.696_{0.026}$	$0.025_{0.001}$	$0.839_{0.006}$	$0.884_{0.004}$	$0.635_{0.033}$	$0.647_{0.013}$
	TabDDPM	$0.624_{0.026}$	$0.043_{0.001}$	$0.841_{0.006}$	$0.869_{0.015}$	$0.541_{0.044}$	$0.270_{0.012}$
churn	TabDiff	$0.650_{0.031}$	$0.058_{0.006}$	$0.865_{0.007}$	$0.907_{0.020}$	$0.551_{0.051}$	$0.361_{0.011}$
	TabSyn	$0.677_{0.013}$	$0.087_{0.004}$	$0.836_{0.005}$	$0.922_{0.016}$	$0.637_{0.012}$	$0.924_{0.008}$
	Vanilla	$0.558_{0.029}$	$0.009_{0.000}$	$0.821_{0.005}$	$0.865_{0.020}$	$0.490_{0.007}$	$0.355_{0.007}$
	RHYTHM	$0.696_{0.008}$	$\mathbf{0.089_{0.005}}$	$\mathbf{0.876_{0.004}}$	$\mathbf{0.924_{0.008}}$	$0.636_{0.012}$	$\mathbf{0.997_{0.001}}$
	CDCD	$0.745_{0.009}$	$\mathbf{0.978_{0.001}}$	$0.864_{0.001}$	$\mathbf{0.861_{0.011}}$	$\mathbf{0.613_{0.013}}$	$1.000_{0.000}$
	CDTD	$0.707_{0.021}$	$0.971_{0.001}$	$0.845_{0.001}$	$0.835_{0.020}$	$0.605_{0.014}$	$1.000_{0.000}$
	TabDDPM	$0.668_{0.015}$	$0.972_{0.004}$	$\mathbf{0.886_{0.003}}$	$0.699_{0.019}$	$0.564_{0.016}$	$0.552_{0.007}$
default	TabDiff	$0.745_{0.020}$	$0.938_{0.003}$	$0.827_{0.003}$	$0.838_{0.019}$	$0.610_{0.010}$	$0.659_{0.005}$
	TabSyn	$\mathbf{0.750_{0.010}}$	$0.977_{0.000}$	$0.859_{0.001}$	$0.826_{0.008}$	$0.603_{0.011}$	$0.983_{0.000}$
	Vanilla	$0.722_{0.019}$	$0.950_{0.004}$	$0.873_{0.001}$	$0.807_{0.006}$	$0.610_{0.019}$	$0.544_{0.009}$
	RHYTHM	$0.703_{0.009}$	$0.967_{0.002}$	$0.879_{0.001}$	$0.718_{0.017}$	$0.605_{0.005}$	$\mathbf{1.000_{0.000}}$
	CDCD	$0.597_{0.002}$	$0.634_{0.016}$	$0.691_{0.002}$	$0.816_{0.050}$	$0.347_{0.005}$	$0.970_{0.004}$
	CDTD	$0.561_{0.000}$	$0.569_{0.003}$	$0.662_{0.001}$	$0.556_{0.007}$	$0.240_{0.000}$	$0.789_{0.012}$
	TabDDPM	$0.566_{0.011}$	$\mathbf{0.882_{0.017}}$	$0.847_{0.002}$	$0.675_{0.044}$	$0.453_{0.012}$	$0.548_{0.013}$
gesture	TabDiff	$0.667_{0.012}$	$0.636_{0.004}$	$0.716_{0.002}$	$\mathbf{0.834_{0.047}}$	$0.559_{0.019}$	$0.678_{0.007}$
	TabSyn	$0.647_{0.007}$	$0.848_{0.013}$	$0.765_{0.003}$	$0.491_{0.024}$	$0.515_{0.015}$	$0.989_{0.003}$
	Vanilla	$0.000_{0.000}$	$0.000_{0.000}$	$0.000_{0.000}$	$0.000_{0.000}$	$0.000_{0.000}$	$0.000_{0.000}$
	RHYTHM	$\mathbf{0.709_{0.004}}$	$0.868_{0.017}$	$\mathbf{0.876_{0.004}}$	$0.726_{0.022}$	$\mathbf{0.656_{0.011}}$	$\mathbf{0.999_{0.001}}$
	CDCD	$\mathbf{0.942_{0.013}}$	$0.488_{0.011}$	$0.734_{0.001}$	$0.961_{0.005}$	$\mathbf{0.940_{0.013}}$	$\mathbf{0.976_{0.004}}$
	CDTD	$0.905_{0.012}$	$0.445_{0.003}$	$0.720_{0.002}$	$\mathbf{0.962_{0.003}}$	$0.903_{0.012}$	$0.806_{0.006}$
	TabDDPM	$0.871_{0.027}$	$0.594_{0.006}$	$\mathbf{0.761_{0.002}}$	$0.899_{0.022}$	$0.865_{0.027}$	$0.238_{0.013}$
intrusion	TabDiff	$0.920_{0.021}$	$0.497_{0.012}$	$0.724_{0.001}$	$0.960_{0.011}$	$0.916_{0.021}$	$0.216_{0.009}$
	TabSyn	$0.873_{0.019}$	$0.484_{0.005}$	$0.733_{0.001}$	$0.849_{0.023}$	$0.868_{0.019}$	$0.811_{0.010}$
	Vanilla	$0.819_{0.035}$	$0.405_{0.006}$	$0.702_{0.004}$	$0.818_{0.010}$	$0.811_{0.035}$	$0.169_{0.012}$
	RHYTHM	$0.938_{0.015}$	$\mathbf{0.657_{0.009}}$	$0.753_{0.002}$	$0.955_{0.010}$	$0.936_{0.015}$	$0.812_{0.004}$
	CDCD	$0.956_{0.005}$	$\mathbf{0.912_{0.004}}$	$0.879_{0.005}$	$0.968_{0.005}$	$0.957_{0.005}$	$\mathbf{0.992_{0.002}}$
	CDTD	$0.958_{0.002}$	$0.903_{0.007}$	$0.857_{0.006}$	$0.965_{0.008}$	$0.959_{0.002}$	$0.928_{0.009}$
	TabDDPM	$0.881_{0.033}$	$0.870_{0.001}$	$0.864_{0.007}$	$0.851_{0.024}$	$0.884_{0.033}$	$0.188_{0.014}$
letter	TabDiff	$0.945_{0.009}$	$0.848_{0.008}$	$0.856_{0.012}$	$\mathbf{0.975_{0.009}}$	$0.946_{0.009}$	$0.403_{0.014}$
	TabSyn	$0.878_{0.024}$	$0.848_{0.007}$	$0.864_{0.003}$	$0.930_{0.016}$	$0.878_{0.025}$	$0.884_{0.010}$
	Vanilla	$0.897_{0.017}$	$0.874_{0.016}$	$0.861_{0.013}$	$0.894_{0.030}$	$0.900_{0.016}$	$0.186_{0.011}$
	RHYTHM	$\mathbf{0.963_{0.003}}$	$0.889_{0.006}$	$\mathbf{0.884_{0.004}}$	$0.960_{0.014}$	$\mathbf{0.963_{0.003}}$	$0.986_{0.002}$
	CDCD	$\mathbf{1.000_{0.000}}$	$\mathbf{0.990_{0.001}}$	$0.929_{0.004}$	$0.981_{0.005}$	$\mathbf{1.000_{0.000}}$	$0.996_{0.001}$
	CDTD	$1.000_{0.000}$	$0.989_{0.003}$	$0.928_{0.006}$	$\mathbf{0.997_{0.004}}$	$1.000_{0.000}$	$0.968_{0.002}$
	TabDDPM	$1.000_{0.000}$	$0.985_{0.003}$	$\mathbf{0.938_{0.004}}$	$0.911_{0.018}$	$1.000_{0.000}$	$0.382_{0.007}$
magic	TabDiff	$1.000_{0.000}$	$0.936_{0.007}$	$0.803_{0.002}$	$0.964_{0.014}$	$1.000_{0.000}$	$0.592_{0.016}$
	TabSyn	$1.000_{0.000}$	$0.973_{0.001}$	$0.881_{0.002}$	$0.936_{0.009}$	$1.000_{0.000}$	$0.915_{0.005}$
	Vanilla	$1.000_{0.000}$	$0.978_{0.002}$	$0.932_{0.003}$	$0.960_{0.019}$	$1.000_{0.000}$	$0.405_{0.014}$
	RHYTHM	$1.000_{0.000}$	$0.944_{0.006}$	$0.871_{0.005}$	$0.962_{0.013}$	$1.000_{0.000}$	$\mathbf{1.000_{0.000}}$
	CDCD	$\mathbf{0.893_{0.026}}$	$\mathbf{0.966_{0.012}}$	$0.776_{0.004}$	$0.884_{0.024}$	$\mathbf{0.839_{0.035}}$	$0.795_{0.011}$
	CDTD	$0.877_{0.017}$	$0.962_{0.013}$	$\mathbf{0.781_{0.008}}$	$0.799_{0.024}$	$0.817_{0.017}$	$0.457_{0.018}$
	TabDDPM	$0.863_{0.048}$	$0.947_{0.011}$	$0.759_{0.010}$	$0.730_{0.077}$	$0.787_{0.070}$	$0.152_{0.005}$
shoppers	TabDiff	$0.839_{0.103}$	$0.952_{0.017}$	$0.707_{0.002}$	$0.870_{0.040}$	$0.776_{0.123}$	$0.369_{0.017}$
	TabSyn	$0.882_{0.028}$	$0.893_{0.021}$	$0.756_{0.005}$	$0.817_{0.045}$	$0.816_{0.037}$	$0.845_{0.014}$
	Vanilla	$0.884_{0.030}$	$0.952_{0.019}$	$0.731_{0.011}$	$0.601_{0.031}$	$0.822_{0.037}$	$0.146_{0.011}$
	RHYTHM	$0.889_{0.028}$	$0.940_{0.006}$	$0.744_{0.005}$	$\mathbf{0.932_{0.017}}$	$0.839_{0.038}$	$\mathbf{0.989_{0.006}}$

Table 13: Results on Range AND Category constraints, per dataset. Score of 1.0 on Magic for F1 and accuracy is because the downstream task collapses to a single class classification problem.

Dataset	Method	Accuracy	Column Pair Trends	Column Shapes	Coverage	F1	Match %
adult	CDCD	$\mathbf{0.743_{0.009}}$	$0.888_{0.014}$	$0.832_{0.007}$	$\mathbf{0.955_{0.006}}$	$\mathbf{0.735_{0.005}}$	$\mathbf{0.988_{0.003}}$
	CDTD	$0.719_{0.030}$	$\mathbf{0.891_{0.012}}$	$0.811_{0.004}$	$0.941_{0.010}$	$0.689_{0.051}$	$0.956_{0.007}$
	TabDDPM	$0.671_{0.048}$	$0.854_{0.013}$	$\mathbf{0.837_{0.004}}$	$0.834_{0.030}$	$0.653_{0.063}$	$0.142_{0.012}$
	TabDiff	$0.671_{0.056}$	$0.842_{0.014}$	$0.833_{0.008}$	$0.904_{0.030}$	$0.641_{0.060}$	$0.349_{0.016}$
	TabSyn	$0.697_{0.049}$	$0.883_{0.006}$	$0.824_{0.002}$	$0.852_{0.010}$	$0.652_{0.080}$	$0.807_{0.010}$
	Vanilla	$0.725_{0.020}$	$0.838_{0.006}$	$0.821_{0.008}$	$0.802_{0.015}$	$0.714_{0.022}$	$0.102_{0.012}$
	RHYTHM	$0.711_{0.054}$	$0.869_{0.014}$	$0.833_{0.003}$	$0.943_{0.007}$	$0.707_{0.055}$	$0.911_{0.009}$
bean	CDCD	$0.993_{0.001}$	$0.876_{0.005}$	$0.702_{0.011}$	$0.878_{0.017}$	$0.936_{0.010}$	$0.796_{0.019}$
	CDTD	$0.992_{0.002}$	$0.889_{0.005}$	$0.806_{0.010}$	$0.873_{0.006}$	$0.922_{0.013}$	$0.506_{0.009}$
	TabDDPM	$0.993_{0.003}$	$0.909_{0.005}$	$\mathbf{0.900_{0.009}}$	$0.777_{0.026}$	$0.932_{0.028}$	$0.223_{0.020}$
	TabDiff	$0.991_{0.003}$	$0.842_{0.006}$	$0.575_{0.014}$	$\mathbf{0.907_{0.014}}$	$0.918_{0.021}$	$0.461_{0.009}$
	TabSyn	$0.994_{0.001}$	$0.904_{0.007}$	$0.838_{0.009}$	$0.819_{0.013}$	$0.945_{0.013}$	$0.671_{0.026}$
	Vanilla	$\mathbf{0.996_{0.001}}$	$\mathbf{0.910_{0.015}}$	$0.884_{0.009}$	$0.789_{0.027}$	$\mathbf{0.956_{0.009}}$	$0.208_{0.012}$
	RHYTHM	$0.993_{0.001}$	$0.848_{0.005}$	$0.553_{0.009}$	$0.889_{0.005}$	$0.933_{0.012}$	$\mathbf{0.928_{0.008}}$
churn	CDCD	$\mathbf{0.718_{0.012}}$	$0.052_{0.003}$	$0.857_{0.006}$	$0.903_{0.005}$	$\mathbf{0.641_{0.023}}$	$0.948_{0.010}$
	CDTD	$0.665_{0.045}$	$0.023_{0.002}$	$0.825_{0.008}$	$0.899_{0.003}$	$0.581_{0.054}$	$0.543_{0.024}$
	TabDDPM	$0.645_{0.018}$	$0.021_{0.003}$	$0.823_{0.010}$	$0.867_{0.036}$	$0.522_{0.050}$	$0.220_{0.020}$
	TabDiff	$0.661_{0.029}$	$0.037_{0.004}$	$0.862_{0.009}$	$0.901_{0.018}$	$0.562_{0.019}$	$0.338_{0.022}$
	TabSyn	$0.657_{0.031}$	$0.070_{0.003}$	$0.830_{0.006}$	$\mathbf{0.920_{0.017}}$	$0.601_{0.032}$	$0.878_{0.016}$
	Vanilla	$0.457_{0.099}$	$0.006_{0.000}$	$0.806_{0.010}$	$0.879_{0.013}$	$0.411_{0.080}$	$0.256_{0.017}$
	RHYTHM	$0.665_{0.053}$	$\mathbf{0.071_{0.008}}$	$\mathbf{0.864_{0.004}}$	$0.918_{0.010}$	$0.591_{0.060}$	$\mathbf{0.957_{0.011}}$
default	CDCD	$0.680_{0.022}$	$0.986_{0.002}$	$0.866_{0.003}$	$\mathbf{0.915_{0.006}}$	$0.581_{0.013}$	$0.987_{0.001}$
	CDTD	$0.675_{0.012}$	$0.976_{0.003}$	$0.848_{0.003}$	$0.912_{0.013}$	$0.583_{0.008}$	$\mathbf{0.988_{0.002}}$
	TabDDPM	$0.648_{0.017}$	$0.973_{0.005}$	$0.879_{0.004}$	$0.665_{0.029}$	$0.558_{0.019}$	$0.308_{0.010}$
	TabDiff	$0.688_{0.040}$	$0.953_{0.007}$	$0.805_{0.002}$	$0.914_{0.014}$	$0.578_{0.028}$	$0.604_{0.002}$
	TabSyn	$\mathbf{0.727_{0.034}}$	$\mathbf{0.988_{0.002}}$	$0.854_{0.003}$	$0.867_{0.005}$	$\mathbf{0.624_{0.033}}$	$0.865_{0.010}$
	Vanilla	$0.687_{0.053}$	$0.961_{0.010}$	$0.868_{0.003}$	$0.759_{0.025}$	$0.577_{0.032}$	$0.204_{0.009}$
	RHYTHM	$0.670_{0.041}$	$0.973_{0.002}$	$\mathbf{0.892_{0.001}}$	$0.807_{0.045}$	$0.581_{0.024}$	$0.949_{0.005}$
gesture	CDCD	$0.611_{0.005}$	$0.626_{0.022}$	$0.680_{0.002}$	$0.893_{0.028}$	$0.327_{0.014}$	$0.959_{0.003}$
	CDTD	$0.589_{0.000}$	$0.563_{0.001}$	$0.649_{0.001}$	$0.695_{0.009}$	$0.247_{0.000}$	$0.721_{0.011}$
	TabDDPM	$0.566_{0.015}$	$\mathbf{0.853_{0.040}}$	$0.834_{0.002}$	$0.714_{0.024}$	$0.432_{0.011}$	$0.523_{0.017}$
	TabDiff	$0.631_{0.069}$	$0.635_{0.023}$	$0.699_{0.003}$	$\mathbf{0.903_{0.032}}$	$0.546_{0.019}$	$0.654_{0.011}$
	TabSyn	$0.648_{0.008}$	$0.824_{0.023}$	$0.756_{0.003}$	$0.614_{0.029}$	$0.500_{0.011}$	$0.986_{0.003}$
	Vanilla	$0.000_{0.000}$	$0.000_{0.000}$	$0.000_{0.000}$	$0.000_{0.000}$	$0.000_{0.000}$	$0.000_{0.000}$
	RHYTHM	$\mathbf{0.707_{0.013}}$	$0.838_{0.013}$	$\mathbf{0.858_{0.004}}$	$0.839_{0.024}$	$\mathbf{0.641_{0.011}}$	$\mathbf{0.995_{0.002}}$
intrusion	CDCD	$0.908_{0.018}$	$0.413_{0.005}$	$\mathbf{0.697_{0.003}}$	$0.925_{0.027}$	$0.891_{0.024}$	$0.710_{0.015}$
	CDTD	$0.912_{0.007}$	$0.353_{0.004}$	$0.671_{0.004}$	$\mathbf{0.940_{0.014}}$	$0.891_{0.006}$	$\mathbf{0.785_{0.016}}$
	TabDDPM	$0.796_{0.165}$	$0.427_{0.021}$	$0.687_{0.013}$	$0.677_{0.035}$	$0.753_{0.193}$	$0.043_{0.012}$
	TabDiff	$0.908_{0.012}$	$0.392_{0.017}$	$0.676_{0.005}$	$0.920_{0.025}$	$0.886_{0.019}$	$0.211_{0.015}$
	TabSyn	$\mathbf{0.914_{0.040}}$	$0.399_{0.007}$	$0.674_{0.004}$	$0.872_{0.014}$	$0.894_{0.056}$	$0.579_{0.028}$
	Vanilla	$0.881_{0.016}$	$0.363_{0.018}$	$0.679_{0.005}$	$0.764_{0.009}$	$0.845_{0.029}$	$0.077_{0.011}$
	RHYTHM	$0.914_{0.036}$	$\mathbf{0.508_{0.005}}$	$0.693_{0.004}$	$0.919_{0.018}$	$\mathbf{0.899_{0.040}}$	$0.596_{0.027}$
letter	CDCD	$\mathbf{0.971_{0.004}}$	$0.860_{0.008}$	$0.856_{0.005}$	$0.965_{0.002}$	$\mathbf{0.970_{0.004}}$	$0.943_{0.005}$
	CDTD	$0.970_{0.010}$	$\mathbf{0.867_{0.010}}$	$0.848_{0.004}$	$0.975_{0.005}$	$0.969_{0.010}$	$0.874_{0.020}$
	TabDDPM	$0.882_{0.029}$	$0.797_{0.010}$	$0.856_{0.013}$	$0.804_{0.026}$	$0.881_{0.030}$	$0.114_{0.013}$
	TabDiff	$0.960_{0.007}$	$0.822_{0.004}$	$0.840_{0.008}$	$\mathbf{0.984_{0.015}}$	$0.959_{0.007}$	$0.381_{0.012}$
	TabSyn	$0.920_{0.024}$	$0.822_{0.006}$	$0.843_{0.004}$	$0.914_{0.016}$	$0.918_{0.025}$	$0.738_{0.020}$
	Vanilla	$0.884_{0.046}$	$0.817_{0.021}$	$0.849_{0.015}$	$0.867_{0.032}$	$0.883_{0.048}$	$0.118_{0.019}$
	RHYTHM	$0.965_{0.009}$	$0.835_{0.007}$	$\mathbf{0.861_{0.002}}$	$0.967_{0.007}$	$0.964_{0.010}$	$\mathbf{0.960_{0.004}}$
magic	CDCD	$\mathbf{1.000_{0.000}}$	$0.988_{0.001}$	$0.863_{0.007}$	$0.938_{0.026}$	$\mathbf{1.000_{0.000}}$	$0.951_{0.007}$
	CDTD	$1.000_{0.000}$	$\mathbf{0.988_{0.001}}$	$\mathbf{0.898_{0.008}}$	$\mathbf{0.950_{0.028}}$	$1.000_{0.000}$	$0.830_{0.012}$
	TabDDPM	$1.000_{0.000}$	$0.977_{0.009}$	$0.886_{0.011}$	$0.734_{0.041}$	$1.000_{0.000}$	$0.141_{0.020}$
	TabDiff	$1.000_{0.000}$	$0.947_{0.009}$	$0.765_{0.007}$	$0.908_{0.020}$	$1.000_{0.000}$	$0.436_{0.020}$
	TabSyn	$1.000_{0.000}$	$0.967_{0.002}$	$0.848_{0.004}$	$0.855_{0.009}$	$1.000_{0.000}$	$0.468_{0.027}$
	Vanilla	$1.000_{0.000}$	$0.963_{0.008}$	$0.858_{0.023}$	$0.800_{0.036}$	$1.000_{0.000}$	$0.112_{0.006}$
	RHYTHM	$1.000_{0.000}$	$0.962_{0.005}$	$0.830_{0.006}$	$0.931_{0.017}$	$1.000_{0.000}$	$\mathbf{0.981_{0.002}}$
shoppers	CDCD	$\mathbf{0.901_{0.012}}$	$\mathbf{0.965_{0.009}}$	$0.771_{0.005}$	$0.911_{0.014}$	$\mathbf{0.848_{0.021}}$	$0.783_{0.010}$
	CDTD	$0.887_{0.019}$	$0.962_{0.011}$	$\mathbf{0.777_{0.007}}$	$0.819_{0.025}$	$0.827_{0.023}$	$0.410_{0.017}$
	TabDDPM	$0.845_{0.044}$	$0.937_{0.006}$	$0.753_{0.011}$	$0.760_{0.061}$	$0.765_{0.048}$	$0.142_{0.005}$
	TabDiff	$0.856_{0.094}$	$0.934_{0.020}$	$0.709_{0.005}$	$0.870_{0.043}$	$0.792_{0.107}$	$0.367_{0.016}$
	TabSyn	$0.858_{0.073}$	$0.907_{0.015}$	$0.749_{0.005}$	$0.837_{0.052}$	$0.792_{0.088}$	$0.810_{0.011}$
	Vanilla	$0.888_{0.045}$	$0.950_{0.021}$	$0.717_{0.007}$	$0.580_{0.047}$	$0.832_{0.055}$	$0.119_{0.008}$
	RHYTHM	$0.825_{0.077}$	$0.935_{0.004}$	$0.739_{0.005}$	$\mathbf{0.945_{0.018}}$	$0.762_{0.086}$	$\mathbf{0.907_{0.015}}$