A framework for conditional diffusion modelling with applications in motif scaffolding for protein design

Anonymous Author(s)
Affiliation
Address
email

Abstract

Many protein design applications, such as binder or enzyme design, require scaffolding a structural motif with high precision. Generative modelling paradigms based on denoising diffusion processes emerged as a leading candidate to address this *motif scaffolding* problem and have shown early experimental success in some cases. In the diffusion paradigm, motif scaffolding is treated as a conditional generation task, and several conditional generation protocols were proposed or imported from the Computer Vision literature. However, most of these protocols are motivated heuristically, e.g. via analogies to Langevin dynamics, and lack a unifying framework, obscuring connections between the different approaches. In this work, we unify conditional training and conditional sampling procedures under one common framework based on the mathematically well-understood *Doob's h-transform*. This new perspective allows us to draw connections between existing methods and propose a new conditional training protocol. We illustrate the effectiveness of this new protocol in both, *image outpainting* and *motif scaffolding* and find that it outperforms standard methods.

1 Introduction

Denoising diffusion models are a powerful class of generative models where noise is gradually added to data samples until they converge to pure noise. The time-reversal of this noising process then allows to transform noise into samples. This process has been widely successful in generating high-quality images [Ho et al., 2020] and has more recently shown promise in designing protein backbones that were validated in experimental protein design workflows [Watson et al., 2023].

Importantly for protein design, diffusion models allow to subject this time-reversed sampling process to a target condition. For proteins, a key condition is the inclusion of a structural motif that grants the protein a particular function, such as binding to a specific target or forming an enzyme active site. However, for these motifs to be foldable and stable, they often need to be integrated into a larger protein structure. While there have been notable successes in scaffolding some motifs experimentally, many still prove challenging to scaffold [Watson et al., 2023]. This makes the development of better conditional generation methods for diffusion models an active area of research, with several contributions from the computer vision, molecular and protein design communities in recent times.

For instance, several methods cast the conditional sampling problem as an inverse (posterior sampling) problem and propose adding a *guidance* term to the time-reversal's drift (Fig. 1c) [e.g. Ho et al., 2022, Chung et al., 2022a]. Another line of work, focusing on 'inpainting', suggests *replacing* the observed variable in the diffused state (Fig. 1b) [e.g. Song et al., 2021c, Dutordoir et al., 2023, Mathieu et al., 2023]. Yet other work performs heuristic conditional training with the target variables in place [Watson et al., 2023, Torge et al., 2023].

Submitted to the NeurIPS 2023 Workshop on New Frontiers of AI for Drug Discovery and Development (AI4D3 2023). Do not distribute.

Method	Stage	Operator Leveraged	Constraint Soft	Constraint Hard	Framework
Amortised h-transform (ours)	Training	✓	✓	✓	Amortised trained h
Classifier free [Ho and Salimans, 2022]	Training	×	×	✓	Amortised trained h
Replacement [Song et al., 2021b]	Sampling	✓	×	✓	?
w/ particles: SMCDiff [Trippe et al., 2022]	Sampling	✓	✓	✓	?
RFDiffusion [Watson et al., 2023]	Training	✓	×	✓	Marginal of h
Classifier guidance [Dhariwal and Nichol, 2021]	Finetuning	×	×	✓	Trained separate $p(\boldsymbol{y}\|\boldsymbol{H}_t)$
Reconstruction guidance [Chung et al., 2022a,b]	Sampling	✓	✓	✓	Moment matching h
w/ particles: TDS [Wu et al., 2023]	Sampling	✓	✓	✓	Moment matching h

Table 1: Taxonomy of conditional methods. **Stage** indicates when the conditional information is acquired. **Operator** indicates whether the measurement operator $\mathcal{A}$ is assumed to be known and thus leveraged by methods. **Constraint** classifies the likelihood as either *hard* or *soft*, as detailed in the main text. **Framework** specifies the mechanism by which conditioning is accomplished. The '?' means that it is unclear how this method fits into the h-transform framework.

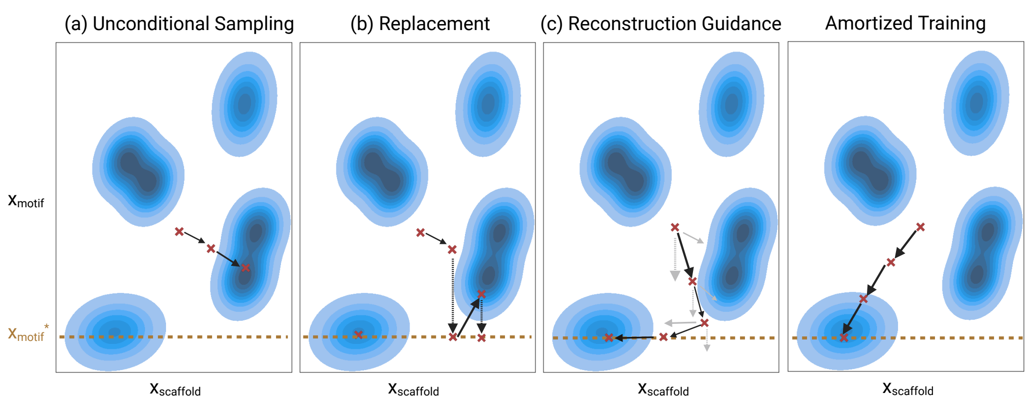

Figure 1: Schematic illustration of several common approaches to (conditionally) sample from a diffusion model. The sampling space is partitioned into motif coordinates (vertical) and scaffold coordinates (horizontal). The target motif is marked as $x^{\star}_{\text{motif}}$ and regions with plausible scaffolds are illustrated as blue blobs. A clear definition of each approach as pseudo-algorithm is given in App. B.

In this work, we reinterpret the conditioning problem leveraging Doob's h-transform. This new perspective provides theoretical backing to existing approaches and naturally leads us to propose a novel method, which we call *amortised training* (Fig. 1d, Alg. 5). We highlight the straightforward implementation and practical use of our theoretical framework by applying it to problems, first as a proof of concept in image generation. We then study the merits and shortcomings of our newly proposed *amortised training* method in more detail for the *motif scaffolding* problem in protein design. We do so by comparing an amortised training implementation of the small-scale diffusion model Genie [Lin and AlQuraishi, 2023] on the RFDiffusion benchmark as well as a newly proposed benchmark dataset based on the SCOPe classification [Chandonia et al., 2022].

Our main contributions are as follows: *i*) We derive a formal framework for conditioning diffusion processes using Doob's h-transform (Sec. 2). *ii*) We use our framework to create a taxonomy of existing methods (Table 1). *iii*) Our taxonomy elucidated the absence of a specific method within the current literature, prompting us to develop and implement this novel approach (Sec. 2.3). *iv*) We empirically assess these different approaches on image generation and protein design (Sec. 3). *v*) Finally, we present plug-and-play algorithms to implement various conditioning schemes (App. B).

2 Theory: Conditioning diffusions via the h-transform, a new perspective

We first show how Doob's h-transform enables diffusion models to satisfy *hard* equality constraints and then generalise this result to handle *soft* constraints in the context of noisy observations.

2.1 Doob's h-transform with hard constraint

Doob's transform provides a formal mechanism for conditioning a stochastic differential equation (SDE) to hit an event at a given time. Formally:

Proposition 2.1. *(Doob's h-transform Rogers and Williams [2000]) Consider the reverse SDE:*

$$\mathrm{d}\boldsymbol{X}_t = \overleftarrow{b}_t(\boldsymbol{X}_t)\,\mathrm{d}t + \sigma_t \overleftarrow{\mathrm{d}\mathbf{W}}_t, \quad \boldsymbol{X}_T \sim \mathcal{P}_T, \tag{1}$$

where time flows backwards and with transition densities $\overleftarrow{p}_{t|s}$. It then follows that the conditioned process $\boldsymbol{X}_t|\boldsymbol{X}_0 \in B$ is a solution of

$$\mathrm{d}\boldsymbol{H}_t = \left(\overleftarrow{b}_t(\boldsymbol{H}_t) - \sigma_t^2\, \nabla_{\boldsymbol{H}_t} \ln \overleftarrow{P}_{0|t}(\boldsymbol{X}_0 \in B|\boldsymbol{H}_t)\right) \mathrm{d}t + \sigma_t \overleftarrow{\mathrm{d}\mathbf{W}}_t, \quad \boldsymbol{X}_T \sim \mathcal{P}_T, \tag{2}$$

such that $\mathrm{Law}\,(\boldsymbol{H}_s|\boldsymbol{H}_t) = \overleftarrow{p}_{s|t,0}(\boldsymbol{h}_s|\boldsymbol{h}_t, \boldsymbol{x}_0 \in B)$ *and* $\mathbb{P}(\boldsymbol{H}_0 \in B) = 1$.

This says that by conditioning a diffusion process to hit a particular event $\boldsymbol{X}_0 \in B$ at a boundary time (e.g. $t = 0$), the resulting conditional process is itself an SDE with an additional drift term. Furthermore, the resulting SDE will hit the specified event within a finite time T. The function $h(t, \boldsymbol{H}_t) \triangleq \overleftarrow{P}_{0|t}(\boldsymbol{X}_0 \in B \mid \boldsymbol{H}_t)$ is referred to as the *h-transform* [Rogers and Williams, 2000, De Bortoli et al., 2021a]. The h-transform drift decomposes into two terms via Bayes rule, a conditional and a prior score:

$$\nabla_{\boldsymbol{H}_t} \ln \overleftarrow{P}_{0|t}(\boldsymbol{X}_0 \in B \mid \boldsymbol{H}_t) = \nabla_{\boldsymbol{H}_t} \ln \overrightarrow{P}_{t|0}(\boldsymbol{H}_t \mid \boldsymbol{X}_0 \in B) - \nabla_{\boldsymbol{H}_t} \ln P_t(\boldsymbol{H}_t), \tag{3}$$

whereby the conditional score ensures that the event is hit at the specified boundary time, while the prior score ensures it is time-reversal of the correct forward process [De Bortoli et al., 2021a] (see App. A.3).

Hard constraint We now consider events of the form $\boldsymbol{X}_0 \in B$ which are described by an equality constraint $\mathcal{A}(\boldsymbol{X}_0) = \boldsymbol{y}$ with $\mathcal{A}$ a known *measurement* (or *forward*) operator and $\boldsymbol{y}$ an observation. We will see concrete examples of $\mathcal{A}$ in Sec. 3.

Corollary 2.2. *Consider the reverse SDE (1), then it follows that*

$$\mathrm{d}\boldsymbol{H}_t = (\overleftarrow{b}_t(\boldsymbol{H}_t) - \sigma_t^2 \nabla_{\boldsymbol{H}_t} \ln \overleftarrow{P}_{0|t}(\mathcal{A}(\boldsymbol{X}_0) = \boldsymbol{y} \mid \boldsymbol{H}_t))\,\mathrm{d}t + \sigma_t \overleftarrow{\mathrm{d}\mathbf{W}}_t, \tag{4}$$

satisfies $\mathrm{Law}\,(\boldsymbol{H}_s|\boldsymbol{H}_t) = \mathrm{Law}\,(\boldsymbol{X}_s|\boldsymbol{X}_t, \mathcal{A}(\boldsymbol{X}_0) = \boldsymbol{y})$ *thus* $\mathrm{Law}\,(\boldsymbol{H}_0) = \mathrm{Law}\,(\boldsymbol{X}_0|\mathcal{A}(\boldsymbol{X}_0) = \boldsymbol{y})$.

Sampling (4) directly provides samples $\boldsymbol{x} \sim p_{\mathrm{data}}$ which also satisfy $\mathcal{A}(\boldsymbol{x}) = \boldsymbol{y}$. Crucially, this SDE is guaranteed to hit the conditioning in finite time, unlike prior equilibrium-motivated approaches [Chung et al., 2022a, Meng and Kabashima, 2022, Finzi et al., 2023, Song et al., 2022, Han et al., 2022, Dutordoir et al., 2023].

Reconstruction guidance To get better insight into the challenge of sampling from Doob's h-transform (4) let us re-express the h-transform as

$$\overleftarrow{P}_{0|t}(\mathcal{A}(\boldsymbol{X}_0) = \boldsymbol{y} \mid \boldsymbol{H}_t) = \int \mathbb{1}_{\mathcal{A}(\boldsymbol{x}_0)=\boldsymbol{y}}(\boldsymbol{x}_0) \overleftarrow{p}_{0|t}(\boldsymbol{x}_0|\boldsymbol{H}_t)\mathrm{d}\boldsymbol{x}_0 \tag{5}$$

where in the case of denoising diffusion models $\overleftarrow{p}_{0|t}(\boldsymbol{x}_0|\cdot)$ is the transition density of the reverse SDE (1). In practice, one does not have access to this transition density – i.e. we can sample from this distribution, but we cannot easily get its value at a certain point. This makes it difficult to approximate the integral. To alleviate this, a strand of recent works [Finzi et al., 2023, Song et al., 2022, Rozet and Louppe, 2023] have proposed Gaussian approximation of $\overleftarrow{p}_{0|t}(\boldsymbol{x}_0|\cdot) \approx \mathcal{N}(\boldsymbol{x}_0 \mid \mathbb{E}[\boldsymbol{X}_0|\boldsymbol{X}_t = \cdot], \Gamma_t)$ leveraging Tweedie's formula and the already trained score network. This line of work is referred as *reconstruction guidance*. We note that whilst proposing to approximate the quantity $\overleftarrow{P}_{0|t}(\mathcal{A}(\boldsymbol{X}_0) = \boldsymbol{y}|\cdot)$, they do not make the connection to Doob's transform and thus are unable to provide guarantees on the conditional sampling that Cor. 2.2 provides. Overall, the Gaussian-based approximations of Doob's h-transform lead to reconstruction guidance-based approaches [Finzi et al., 2023, Rozet and Louppe, 2023, Chung et al., 2022a, Han et al., 2022, Song et al., 2022] $\mathrm{d}\boldsymbol{H}_t = \left(\overleftarrow{b}_t(\boldsymbol{H}_t) + \sigma_t^2 \nabla_{\boldsymbol{H}_t} ||\boldsymbol{y} - \mathrm{A}\mathbb{E}[\boldsymbol{X}_0|\boldsymbol{X}_t = \boldsymbol{H}_t]||^2_{\Gamma_t}\right) \mathrm{d}t + \sigma_t \overleftarrow{\mathrm{d}\mathbf{W}}_t$, $\boldsymbol{X}_T \sim \mathcal{P}_T$, where Γ_t acts as a guidance scale [Simon V et al., 2023, Rozet and Louppe, 2023], and A is a matrix if $\mathcal{A}$ is linear otherwise $\mathrm{A} = \mathrm{d}\mathcal{A}(\mathbb{E}[\boldsymbol{X}_0|\boldsymbol{X}_t = \boldsymbol{H}_t])$.

2.2 Generalised h-transform for soft constraints

In the previous Sec. 2.1, we showed how the h-transform allows for conditioning on *hard* constraints, correcting the reverse process to satisfy some observation $P(\boldsymbol{y}|\boldsymbol{x}_0) \propto \mathbb{1}_{\mathcal{A}(\boldsymbol{X}_0)=\boldsymbol{y}}(\boldsymbol{x}_0)$. Yet, many scenarios deal with *soft* constraints, modelling noisy observation $\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}) + \eta$ with a density $p(\boldsymbol{y}|\boldsymbol{x}_0)$, typically with the goal of sampling from the posterior $p(\boldsymbol{x}_0|\boldsymbol{Y} = \boldsymbol{y}) = p(\boldsymbol{y}|\boldsymbol{x}_0)p_{\mathrm{data}}(\boldsymbol{x}_0)/p(\boldsymbol{y})$ as in noisy inverse problems [Song et al., 2021a, Chung et al., 2022a,b]. In this section, we present a generalisation of the h-transform applicable to denoising diffusion models that build on results in [Vargas et al., 2023]:

Proposition 2.3. *(Noisy conditioning) Given the following forward SDE:*

$$\mathrm{d}\boldsymbol{X}_t = f_t(\boldsymbol{X}_t)\,\mathrm{d}t + \sigma_t\,\overrightarrow{\boldsymbol{W}_t}, \quad \boldsymbol{X}_0 \sim \mathcal{P}_{data} \tag{6}$$

it follows that the following reverse SDE with marginals p_t

$$\begin{aligned} &\boldsymbol{H}_T \sim \mathrm{Law}\,(\boldsymbol{X}_T|\boldsymbol{X}_0) \\ &\mathrm{d}\boldsymbol{H}_t = \big(f_t(\boldsymbol{H}_t) + \sigma_t^2(\nabla_{\boldsymbol{H}_t}\ln p_t(\boldsymbol{H}_t) + \nabla_{\boldsymbol{H}_t}\ln p_{y|t}(\boldsymbol{Y}=\boldsymbol{y}|\boldsymbol{H}_t))\big)\,\mathrm{d}t + \sigma_t\,\overleftarrow{\mathrm{d}\mathbf{W}}_t, \end{aligned} \tag{7}$$

satisfies $\mathrm{Law}\,(\boldsymbol{H}_0) = p(\boldsymbol{x}_0|\boldsymbol{Y}=\boldsymbol{y})$ *where* $p_{y|t}(\boldsymbol{Y}=\boldsymbol{y}|\cdot) = \int p(\boldsymbol{Y}=\boldsymbol{y}|\boldsymbol{x}_0)\overrightarrow{p}_{0|t}(\boldsymbol{x}_0|\cdot)\mathrm{d}\boldsymbol{x}_0$.

In short, the above results give a variant of the h-transform that allows to sample from noisy posteriors. This provides theoretical backing to methodologies such as DPS [Chung et al., 2022a], in which the SDE (8) is used to solve noisy inverse problems.

Corollary 2.4. *Furthermore, for an Ornstein-Uhlenbeck (OU) forward process, i.e. with drift* $f_t(\boldsymbol{x}) = -\beta_t\boldsymbol{x}$ *and diffusion* $\sigma_t = \sqrt{2\beta_t}$*, we have that*

$$\mathrm{d}\boldsymbol{H}_t = -\beta_t\big(\boldsymbol{H}_t + 2\nabla_{\boldsymbol{H}_t}\ln p_t(\boldsymbol{H}_t) + 2\nabla_{\boldsymbol{H}_t}\ln p_{y|t}(\boldsymbol{Y}=\boldsymbol{y}|\boldsymbol{H}_t)\big)\mathrm{d}t + \sqrt{2\beta_t}\,\overleftarrow{\mathrm{d}\mathbf{W}}_t,\ \ \boldsymbol{H}_T \sim \mathcal{N}(0, I) \tag{8}$$

satisfies $\mathrm{Law}\,(\boldsymbol{H}_0) \approx p(\boldsymbol{x}_0|\boldsymbol{Y}=\boldsymbol{y})$*. As such,* $\boldsymbol{H}_T$ *inherits the rapid convergence guarantees of the OU process [De Bortoli, 2022, De Bortoli et al., 2021b], in particular* $||\mathrm{Law}\,(\boldsymbol{H}_T) - \mathcal{N}(0,I)||_{\mathrm{TV}} \leq \mathcal{O}\big(e^{-T/\bar{\beta}}\big)$ *for some* $\bar{\beta} > 0$.

2.3 Amortised training of h-transform

In this section, we propose an objective for learning Doob's h-transform at training time in an amortised fashion. Note that since $\overrightarrow{P}_{0|t}(\mathcal{A}(\boldsymbol{X}_0) = \boldsymbol{y}|\boldsymbol{X}_t = \boldsymbol{h}) = \overrightarrow{P}_{t|0}(\boldsymbol{h}|\mathcal{A}(\boldsymbol{X}_0) = \boldsymbol{y})p_0(\mathcal{A}(\boldsymbol{X}_0) = \boldsymbol{y})/p_t(\boldsymbol{X}_t = \boldsymbol{h})$ we can re-express the Doob's transformed SDE of a reversed OU process as:

$$\mathrm{d}\boldsymbol{H}_t = -\beta_t\left(\boldsymbol{H}_t + 2\nabla_{\boldsymbol{H}_t}\ln\overrightarrow{P}_{t|0}(\boldsymbol{H}_t|\mathcal{A}(\boldsymbol{X}_0) = \boldsymbol{y})\right)\mathrm{d}t + \sqrt{2\beta_t}\,\overleftarrow{\mathrm{d}\mathbf{W}}_t, \quad \boldsymbol{H}_T \sim \mathrm{Law}\,(\boldsymbol{X}_T).$$

Proposition 2.5. *The minimiser of*

$$f^* = \arg\min_f \mathbb{E}_{\boldsymbol{Y}\sim p_{|\mathcal{A},\boldsymbol{X}_0},\mathcal{A}\sim p,\boldsymbol{X}_0\sim p_{\mathrm{data}}}\left[\int_0^T ||f(t,\boldsymbol{X}_t,\boldsymbol{Y},\mathcal{A}) - \nabla_{\boldsymbol{X}_t}\ln\overrightarrow{p}_{t|0}(\boldsymbol{X}_t|\boldsymbol{X}_0)||^2\mathrm{d}t\right] \tag{9}$$

is given by the conditional score $f_t^*(\boldsymbol{h},\boldsymbol{y},\mathcal{A}) = \nabla_{\boldsymbol{h}}\ln\overrightarrow{p}_{t|0}(\boldsymbol{h}|\boldsymbol{Y}=\boldsymbol{y})$.

This is referred as *amortised* learning for conditional sampling, since practically the neural network approximating the (conditional) score is amortised over $\mathcal{A}$ and $\boldsymbol{y}$, instead of learning a separate network for each condition. This approach is reminiscent of 'classifier free guidance' [Ho and Salimans, 2022] where the score network is amortised over some auxiliary variable (e.g. as in text-to-image models [Ramesh et al., 2021]), or of RFDiffusion [Watson et al., 2023] where proteins are designed given a specific subset motif. Our framework is different to 'classifier free guidance' as $\mathcal{A}$ is assumed to be known (e.g. an inpainting mask), and to RFDiffusion since the conditioning variable $\boldsymbol{Y}$ being a subset of $\boldsymbol{X}$, is also being noised during training and denoised when sampling (see Alg. 5), also note due to its formulation classifier guidance would be unable to noise a subset of $\boldsymbol{X}$ (the motif) as we do.

2.4 Conceptual comparison with RFDiffusion

As highlighted in Alg. 4 and in contrast to our approach, RFDiffusion [Watson et al., 2023] does not noise the motif coordinates $\boldsymbol{X}_0^{[M]}$ with the forward OU-Process, instead it directly aims to sample from $p(\boldsymbol{X}_t^{[\setminus M]}|\boldsymbol{X}_0^{[M]})$ and estimate this score while keeping the motif fixed.

We can relate this approach to our amortised learning of Doob's h-transform, by noting that RF diffusion can be understood as learning the marginal conditional score:

$$p(\boldsymbol{X}_t^{[\setminus M]}|\boldsymbol{X}_0^{[M]}) = \int \overbrace{p(\boldsymbol{X}_t|\boldsymbol{X}_0^{[M]})}^{\propto h(t,\boldsymbol{X}_t)p_t(\boldsymbol{X}_t)} \, d\boldsymbol{X}_t^{[M]}. \tag{10}$$

This can be viewed as RFDiffusion estimating a marginal counterpart of our amortised h-transform approach. See Algs. 4 and 5 for more details on how these approaches differ in a pseudo-code implementation.

3 Experimental results

To compare the various conditional generation methods, we first highlight our results from initial tests in the image setting and then discuss the motif scaffolding problem in protein design in more detail.

3.1 Conditional image generation.

The task of 'image outpainting' mimics the motif scaffolding problem in protein design and amounts to conditioning the diffusion model on a central patch of an image. The measurement model $\mathcal{A} \in \{0,1\}^{n \times d}$ will select n central pixels out of an image in $\mathbb{R}^d$. We consider noise-free conditions (i.e. hard constraints). We focus on the CELEBA [Liu et al., 2015] and FLOWERS [Nilsback and Zisserman, 2008] image datasets. We empirically evaluate the AMORTISED approach where the mask is provided at training time as an extra channel, along with RECONSTRUCTION GUIDANCE (Alg. 8) and REPLACEMENT (Alg. 9) methods for which the score network is trained without access to the mask, and are then queried at sampling time. The quality of conditional samples is measured by the mean squared error (MSE) and LPIPS perceptual metric [Zhang et al., 2018]. See App. D for further details. We empirically observe from Table 3 that the AMORTISED approach slightly outperforms sampling-based methods, which are on par with each other.

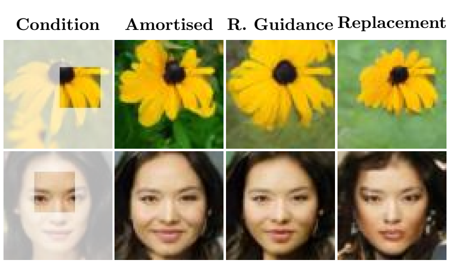

Figure 2: Some conditional samples.

METRIC	AMORTISED	R. GUIDANCE	REPLACEMENT
FLOWERS			
MSE (↓)	$0.34_{\pm 0.01}$	$0.27_{\pm 0.01}$	$0.28_{\pm 0.01}$
LPIPS (↓)	$0.25_{\pm 0.00}$	$0.29_{\pm 0.01}$	$0.33_{\pm 0.01}$
CELEBA			
MSE (↓)	$0.26_{\pm 0.01}$	$0.30_{\pm 0.01}$	$0.34_{\pm 0.00}$
LPIPS (↓)	$0.14_{\pm 0.00}$	$0.15_{\pm 0.01}$	$0.17_{\pm 0.00}$

Table 2: Quantitative assessment of conditional samples w.r.t to ground-truth.

3.2 Conditional protein design: motif scaffolding

The task of motif scaffolding in our protein setting amounts to sampling protein C alpha atom coordinates $\boldsymbol{x} \in \mathbb{R}^d$ such that it contains a given subset of C alpha coordinates $\boldsymbol{y} \in \mathbb{R}^n$, i.e. $\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}) = \mathrm{A}\boldsymbol{x}$, where $\mathcal{A} \in \{0,1\}^{n \times d}$ is a masking matrix which selects n observed C alpha coordinates. We perform two sets of motif scaffolding experiments. We firstly compare our proposed AMORTISED approach to REPLACEMENT and RECONSTRUCTION GUIDANCE as we did in the image case. Upon observing that AMORTISED performs significantly better, we then dive into a more detailed analysis of this method on the RFDiffusion benchmark, as well as a new SCOPe based benchmark that is created from a hierarchical structure and sequence-based split described below.

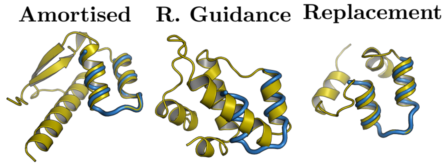

Figure 3: Conditional protein designs in yellow with target motif 3IXT in blue.

METRIC	AMORTISED	R. GUIDANCE	REPLACEMENT
% Success (↑)	20.0	1.5	0.5
% scRMSD < 2 Å(↑)	35.1	45.1	0.6
% mRMSD < 1 Å(↑)	50.0	4.2	24.3

Table 3: RFDIFF benchmark metrics. Success: pAE < 5, scRMSD < 2Å, motifRMSD < 1Å, pLDDT > 70, scTM > 0.5. Details in Sec. 3.2.

Data We evaluate on the RFDiffusion motif benchmark [Watson et al., 2023] and on a self-curated SCOPe benchmark based on a hierarchical structure-based split, jointly with a sequence-similarity based split. For the RFDiffusion benchmark, we tested all sequence-contiguous motifs, resulting in 11 different motif design tasks. Our method readily extends to the non-contiguous motif setting and future work will address this in more detail. The performance on each of these targets is depicted in Fig. 4. For the SCOPe dataset, we leverage the hierarchical structure classification scheme of the SCOPe database [Chandonia et al., 2022] to create train-test splits that allow us to investigate how well the model can scaffold motifs from unseen folds, families and superfamilies and how difficult these tasks are with respect to each other. In particular, for training, we hold out four clusters of protein structures at the fold level, four at the family level and four at the superfamily level (Fig. 5a) and evaluate the motif-scaffolding performance of the model on this structure-based hold-out set (Fig. 5b-d).

Diffusion process We use a discrete-time DDPM [Ho et al., 2020] formulation for the diffusion model with $N = 1000$ steps and cosine β-schedule [Dhariwal and Nichol, 2021].

Noise model The denoising model ε_θ is adapted from the Genie diffusion model [Lin and AlQuraishi, 2023]. In Genie, the denoiser architecture consists of an SE(3)-invariant encoder and an SE(3)-equivariant decoder. While the network uses Frenet-Serret frames as intermediate representations, the diffusion process itself is defined in Euclidean space over the C alpha coordinates. Similar to AlphaFold2, the denoiser network consists of a single representation track that is initialised via a single feature network and a pair representation track that is initialised via a pair feature network. These two representations are further transformed via a pair transform network and are used in the decoder for noise prediction via IPA Jumper et al. [2021].
To evaluate unconditional sampling-based methods, we retrained the Genie denoising network for 4000 epochs on 4 A100 GPUs ($\sim$300 A100 hours in total). We stopped training at this point, as we observed an almost comparable performance to the publicly available model weights (which were obtained after training for 50'000 epochs).
To evaluate the AMORTISED approach (Alg. 5), we perform a minor modification to the unconditional Genie model by adding an additional conditional pair feature network that takes the motif frames as input with the ground truth coordinates for the motif and 0 as values for all other coordinates that are not part of the motif. The output of this motif-conditional pair feature network is concatenated with the output of the unconditional pair feature network to form an intermediate dimension of twice the channel size compared to the unconditional model, before being linearly projected down to the channel size of the unconditional model. From then onward the output is processed by the remaining Genie components as in the unconditional model. The implementation is therefore similar to the image case, where the motif features are presented as additional input and the model learns to use these for reconstructing the motif. This minor alteration of the Genie architecture means our amortised network has 4.162M parameters while the unconditional Genie networks have 4.087M parameters ($\sim 1.8\%$ fewer).

Methods In the amortised setting we follow the pseudo-code definition given in Alg. 5. In 80% of the training steps, we pass a condition to the network. The other 20% contains an empty mask consisting of only 0's. For the reconstruction guidance method (Alg. 8), we use a time-dependent guidance term of $\gamma_t = \alpha_t(1 - \alpha_t)$.

Metrics We measure the performance of the methods across two axes: designability and success rate.

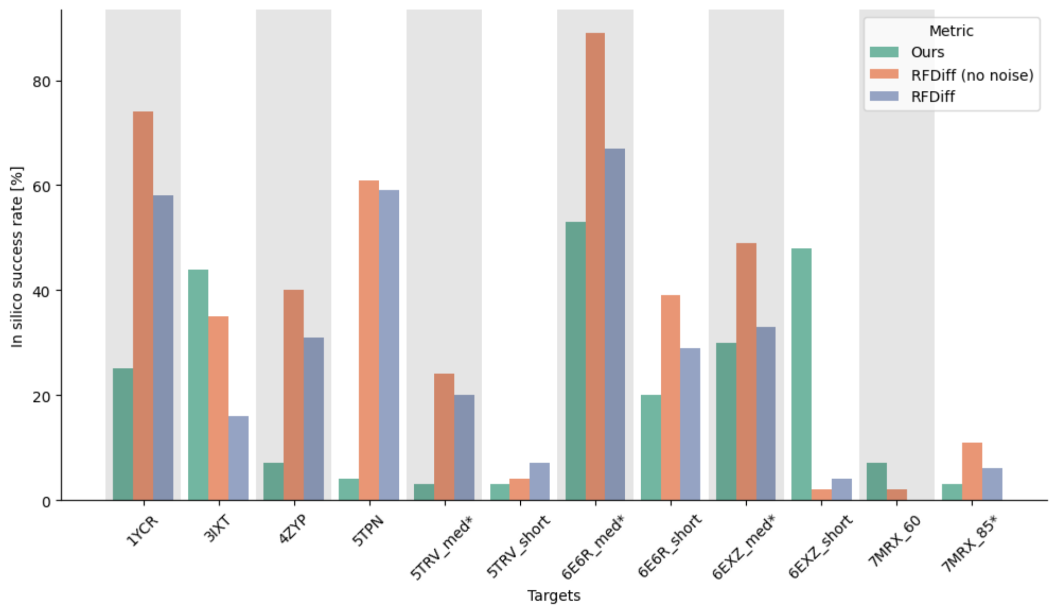

Figure 4: Comparison of our method to RFDiffusion for motif scaffolding for 12 continuous targets. Note that we trained our 4.1M parameter model for only 4000 epochs (∼300 A100 hours in total), which is significantly less both in compute and parameter size than RFDiffusion (∼26’000+ A100 hours, 59.8M parameters). For the motifs marked with *, we had to shorten the sampled scaffold ranges on both sides of the motif from 0-65 (0-63 for TMRX80) to 0-50 since we trained our version of Genie only on protein generation up to a length of 128 residues. Performance numbers from RFDiffusion are taken from the original publication Watson et al. [2023] and our designs were created with the same design specifications as described there. We note that our folding step uses ESMFold instead of AlphaFold2, but we have future plans to use AlphaFold2 for a more direct comparison.

To assess whether a particular protein scaffold is *designable*, we run the same pipeline as Lin and AlQuraishi [2023], consisting of an inverse folding generated C_α backbones with ProteinMPNN and then re-folding the designed sequences via ESMFold. The considered metrics and their corresponding thresholds are the following:

- scTM > 0.5: This refers to the TM-score between the structure that’s been designed and the predicted structure based on self-consistency as previously described. The scTM-score ranges from 0 to 1. Higher scores indicate a higher likelihood that the input structure can be designed.
- scRMSD < 2 Å : The scRMSD metric is akin to the scTM metric. However, it uses the RMSD (Root Mean Square Deviation) to measure the difference between the designed and predicted structures, instead of the TM-score. This metric is more stringent than scTM as RMSD, being a local metric, is more sensitive to minor structural variances.
- pLDDT > 70 and pAE < 5: Both scTM and scRMSD metrics depend on a structure prediction method like AlphaFold2 or ESMFold to be reliable. Hence, additional confidence metrics such as pLDDT and pAE are employed to ascertain the reliability of the self-consistency metrics.

In addition, we want to judge whether the motif scaffolding was successful or not. Therefore, similar to previous work by Watson et al. [2023], we calculate the motifRMSD between the predicted design structure and the original input motif and judge samples with < 1 Å motifRMSD as a successful motif scaffold.

Results We evaluate all three approaches on the continuous motifs from the RFDIFFusion motif benchmark [Watson et al., 2023]. For the AMORTISED approach we retrain the Genie model [Lin and AlQuraishi, 2023] in an amortised fashion (Alg. 5), while for the R. GUIDANCE and REPLACEMENT

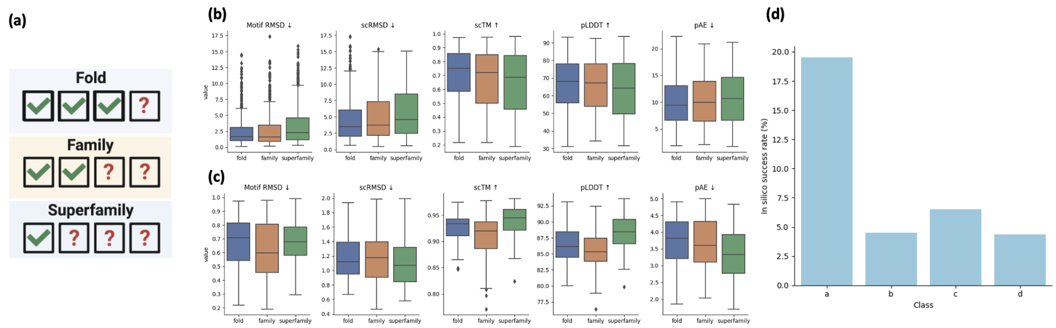

Figure 5: Data ablation study on a newly curated SCOPe benchmark dataset with our amortised training model. (a) We utilise the hierarchical structural clustering of SCOPe to create hold-out sets at three different levels of structural hierarchy: the fold, the family and the superfamily level. (b) We test the motif scaffolding performance on these splits and see decreasing scaffolding success for structurally dissimilar samples. (c) The same metrics as in (b), but only for samples that fulfill the definition of in silico success. (d) Scaffolding success by SCOPe class. Alpha helices can be scaffolded successfully, whereas other classes are more challenging.

methods we used the publicly available unconditional model. We observe that amortised training outperforms the other approaches, especially replacement sampling (Fig. 3.2).

To better understand how well the AMORTISED conditioning approach works, we break down our model performance on the different targets and compare it to the performance of RFDiffusion (Fig. 4). Despite having trained a smaller model with fewer computing resources, we obtained competitive performance on several targets.

We also ablate our model performance w.r.t. structural dissimilarity of the motif compared to the training set via our previously described SCOPe benchmark. Testing the motif-scaffolding performance of the amortised model on this data, we see that the scaffolding success decreases from fold-over family to superfamily, indicating that scaffolding a motif from a protein that is more different to the training set is harder (Fig. 5b-c). We also quantitatively observe an anecdotal phenomenon in protein design: while alpha helices are relatively easy to scaffold, domains from other classes have significantly lower success rates (Fig. 5d). We hope that this benchmark set will help to address these issues in future modelling efforts.

4 Conclusion

We presented a unified framework, based on Doob's h-transform, to better understand and classify different conditional diffusion methods. Based on the gained insights, we developed a novel AMORTISED conditional sampling scheme (Alg. 5) which differs from existing approaches in that it takes into account the measurement operator. For the motif scaffolding task this means we denoise both the scaffold and the motif. We evaluated the AMORTISED approach on image outpainting and motif scaffolding in protein design and outperform standard methods. We further investigated the performance of the AMORTISED approach by comparing to RFDiffusion on contiguous motifs. Surprisingly, our AMORTISED implementation of Genie [Lin and AlQuraishi, 2023] achieves notable *in-silico* success rates of between 3 – 50% (as per the RFDiffusion definition) across the targets. Though it lags behind RFDiffusion in 9/12 targets, it achieves this without low-temperature sampling, a mere 10% of RFDiffusion's parameter count, and after being trained for just 1.2% of the time. This positions the AMORTISED approach as a promising candidate for further improving motif scaffolding, potentially opening up new applications in protein engineering for drug discovery and enzyme design.

References

John-Marc Chandonia, Lindsey Guan, Shiangyi Lin, Changhua Yu, Naomi K Fox, and Steven E Brenner. Scope: improvements to the structural classification of proteins–extended database to facilitate variant interpretation and machine learning. *Nucleic acids research*, 50(D1):D553–D559, 2022.

Hyungjin Chung, Jeongsol Kim, Michael T Mccann, Marc L Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. *arXiv preprint arXiv:2209.14687*, 2022a.

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, 2022b.

Valentin De Bortoli. Convergence of denoising diffusion models under the manifold hypothesis. *arXiv preprint arXiv:2208.05314*, 2022.

Valentin De Bortoli, Arnaud Doucet, Jeremy Heng, and James Thornton. Simulating diffusion bridges with score matching. *arXiv preprint arXiv:2111.07243*, 2021a.

Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion Schrödinger bridge with applications to score-based generative modeling. *Advances in Neural Information Processing Systems*, 34:17695–17709, 2021b.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Vincent Dutordoir, Alan Saul, Zoubin Ghahramani, and Fergus Simpson. Neural diffusion processes. In *International Conference on Machine Learning*, pages 8990–9012. PMLR, 2023.

Marc Anton Finzi, Anudhyan Boral, Andrew Gordon Wilson, Fei Sha, and Leonardo Zepeda-Núñez. User-defined event sampling and uncertainty quantification in diffusion models for physical dynamical systems. In *International Conference on Machine Learning*, pages 10136–10152. PMLR, 2023.

Xizewen Han, Huangjie Zheng, and Mingyuan Zhou. Card: Classification and regression diffusion models. *Advances in Neural Information Processing Systems*, 35:18100–18115, 2022.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video Diffusion Models, June 2022.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

Yeqing Lin and Mohammed AlQuraishi. Generating novel, designable, and diverse protein structures by equivariantly diffusing oriented residue clouds. *arXiv preprint arXiv:2301.12485*, 2023.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11461–11471, 2022.

Emile Mathieu, Vincent Dutordoir, Michael J Hutchinson, Valentin De Bortoli, Yee Whye Teh, and Richard E Turner. Geometric neural diffusion processes. *arXiv preprint arXiv:2307.05431*, 2023.

Xiangming Meng and Yoshiyuki Kabashima. Diffusion model based posterior sampling for noisy linear inverse problems. *arXiv preprint arXiv:2211.12343*, 2022.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*, Dec 2008.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8821–8831. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/ramesh21a.html.

L Chris G Rogers and David Williams. *Diffusions, Markov processes and martingales: Volume 2, Itô calculus*, volume 2. Cambridge university press, 2000.

Francois Rozet and Gilles Louppe. Score-based data assimilation. *arXiv preprint arXiv:2306.10574*, 2023.

Mathis Simon V, Julia Komorowska Urszula, Jamnik Mateja, and Lio Pietro. Normal mode diffusion: Towards dynamics-informed protein design. *The 2023 ICML Workshop on Computational Biology. Baltimore, Maryland, USA, 2023. C*, 2023.

Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In *International Conference on Learning Representations*, 2022.

Yang Song, Liyue Shen, Lei Xing, and Stefano Ermon. Solving inverse problems in medical imaging with score-based generative models. *arXiv preprint arXiv:2111.08005*, 2021a.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=PxTIG12RRHS.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021c. URL https://openreview.net/forum?id=PxTIG12RRHS.

Jos Torge, Charles Harris, Simon V. Mathis, and Pietro Lio. Diffhopp: A graph diffusion model for novel drug design via scaffold hopping. *arXiv preprint arXiv:2308.07416*, 2023.

Brian L Trippe, Jason Yim, Doug Tischer, David Baker, Tamara Broderick, Regina Barzilay, and Tommi Jaakkola. Diffusion probabilistic modeling of protein backbones in 3d for the motif-scaffolding problem. *arXiv preprint arXiv:2206.04119*, 2022.

Francisco Vargas, Andrius Ovsianas, David Fernandes, Mark Girolami, Neil D Lawrence, and Nikolas Nüsken. Bayesian learning via neural Schrödinger–Föllmer flows. *Statistics and Computing*, 33 (1):3, 2023.

Joseph L Watson, David Juergens, Nathaniel R Bennett, Brian L Trippe, Jason Yim, Helen E Eisenach, Woody Ahern, Andrew J Borst, Robert J Ragotte, Lukas F Milles, et al. De novo design of protein structure and function with rfdiffusion. *Nature*, pages 1–3, 2023.

Luhuan Wu, Brian L Trippe, Christian A Naesseth, David M Blei, and John P Cunningham. Practical and asymptotically exact conditional sampling in diffusion models. *arXiv preprint arXiv:2306.17775*, 2023.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

A Background on diffusion formulations

A.1 Continuous and discrete diffusion formulations

The discretised DDPM versions with various discrete time schedules amount to the time-dependent OU process

$$\mathrm{d}\mathbf{X}_t = -\frac{\beta(t)}{2}\mathbf{x}_t\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\overline{\mathbf{W}}_t \tag{11}$$

where choosing different time schedules amounts to choosing different functions $\beta(t)$. This process gives rise to the Green's function for transition probabilities

$$\begin{aligned} p(\mathbf{x}, t|\mathbf{x}_0, 0) &= \vec{p}_{t|0}(\mathbf{x}|\mathbf{x}_0) && (12)\\ &= \mathcal{N}\left(\mathbf{x}_0 e^{-\int_0^T \frac{\beta(s)}{2}\mathrm{d}s}, \int_0^T \beta(t) e^{-\int_0^{T-t}\beta(s)\mathrm{d}s}\mathrm{d}t\right) && (13)\\ &= \mathcal{N}\left(\mathbf{x}_0 e^{-\int_0^T \frac{\beta(s)}{2}\mathrm{d}s}, \left(1 - e^{-\int_0^T \beta(s)\mathrm{d}s}\right)\right). && (14) \end{aligned}$$

With $\bar{\alpha}(t) = e^{-\int_0^T \beta(s)\mathrm{d}s}$, this is the familiar form (Ho et al. [2020]):

$$p(\mathbf{x}, t|\mathbf{x}_0, 0) = \vec{p}_{t|0}(\mathbf{x}|\mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_0\sqrt{\bar{\alpha}(t)}, (1 - \bar{\alpha}(t))\right), \tag{15}$$

with $\bar{\alpha}(t)$ time-dependent and we can therefore choose different functional forms for the noise schedule by either choosing the transition parameters $\beta(t)$ or the cumulative parameters $\alpha(t)$.

If we define the noise schedule in terms of $\beta(t)$, the time-dependent OU process is immediately apparent (see (11)).

If we define the noise schedule in terms of $\bar{\alpha}(t)$, the mean and variance of the corresponding OU process can simply be obtained from

$$\beta(t) = -\frac{\mathrm{d}}{\mathrm{d}t}\left[\ln \bar{\alpha}(t)\right]. \tag{16}$$

A.2 Score, noise and mean diffusion formulations

The score-based model used for generation at inference time can be parametrised to model different quantities. The three most common one are the score, the noise and the mean.

When starting from the DDPM formulation of describing the diffusion process as a Gaussian linear Markov chain, it is natural to let the network predict the mean of this Gaussian, with the covariance being a fixed parameter:

$$\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}(t)}}\boldsymbol{\varepsilon}_t) \tag{17}$$

However, we have access to the input $\mathbf{x}_t$ at training time and can therefore reparameterize the Gaussian in order to make our network predict the noise $\boldsymbol{\varepsilon}_t$instead of the mean μ_t:

$$\mathbf{x}_{t-1} = \mathcal{N}(\mathbf{x}_{t-1}; \frac{1}{\sqrt{\alpha_t}}(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}(t)}}\boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, t)) \tag{18}$$

When starting from the score-based SDE formulation, one can instead let the network predict the score term in order to minimise the following score matching loss:

$$\mathcal{L} = \mathbb{E}_{t,\mathbf{x}_0,\mathbf{x}_t}||s_\theta(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t}\ln \vec{p}_{t|0}(\mathbf{x}_t|\mathbf{x}_0)||^2/\sigma_t^2 \tag{19}$$

A.3 Doob's h-transform intuition

As mentioned before Doob's h-transform adds a new drift to the SDE which amounts to two terms (via Bayes Theorem), a conditional and an unconditional score:

$$\nabla \ln \overleftarrow{P}_{0|t}(\boldsymbol{X}_0 \in B|\cdot) = \nabla \ln \overrightarrow{P}_{t|0}(\cdot|\boldsymbol{X}_0 \in B) - \nabla \ln P_t(\cdot) \tag{20}$$

Interestingly, these two terms provide for a unique intuition: the Doob's transform SDE is the time reversal of the forward SDE corresponding to (1), that is the time reversal of the forward SDE

$$\mathrm{d}\boldsymbol{X}_t = \overrightarrow{b}_t(\boldsymbol{X}_t)\,\mathrm{d}t + \sigma_t \overrightarrow{\mathrm{d}\mathbf{W}}_t, \quad \boldsymbol{X}_0 \sim \overrightarrow{P}_0(\cdot|\boldsymbol{X}_0 \in B), \tag{21}$$

coincides with the Doob transformed SDE (2) [De Bortoli et al., 2021a].

Thus we can view Doob's transform as the following series of steps:

1. Time reverse the SDE we want to condition ((2) to (21)).
2. Impose the condition via ancestral sampling from the conditioned distribution/posterior.
3. Time reverse once more to be in the same time direction as we started.

A.4 Examples

Truncated normal Here for illustrative purposes we frame the problem of sampling from a truncated normal distribution as simulating an SDE that is given by Doob's h-transform.

Let's remind that a 1d truncated normal distribution had a density $p(x|a,b) \propto \mathbb{1}_{x\in(a,b)}(x)\mathcal{N}(x|\mu,\sigma^2)$. Now, let's assume a data distribution $p_0(x) = \mathrm{N}(\mu,\sigma^2)$ which is noised with an OU process (11). Thus we have that $p(x_0|x_t) = \mathrm{N}(x_0|\hat{\mu}_{0|t}(x_t), \hat{\sigma}_{0|t}(x_t)^2)$ is Gaussian, and so is $p(x_t) = \mathrm{N}(x_t|\hat{\mu}_t, \hat{\sigma}_t^2)$. Let's add the constraint that the process hit at time $t = 0$ the event $\boldsymbol{X}_0 \in (a,b)$.

$$\mathrm{d}\boldsymbol{H}_t = \beta(t)\left(\frac{\boldsymbol{H}_t}{2} + \nabla_{\boldsymbol{H}_t} \ln \overrightarrow{P}_t(\boldsymbol{H}_t) - \nabla_{\boldsymbol{H}_t} \ln \overleftarrow{P}_{0|t}(\boldsymbol{X}_0 \in (a,b) \mid \boldsymbol{H}_t)\right)\mathrm{d}t + \sqrt{\beta(t)}\,\overleftarrow{\mathrm{d}\mathbf{W}}_t, \tag{22}$$

We have that the h-transform is given by

$$\begin{aligned} h(t,\boldsymbol{H}_t) = \overleftarrow{P}_{0|t}(\boldsymbol{X}_0 \in (a,b)|\boldsymbol{H}_t) &= \int \mathbb{1}_{x\in(a,b)}(\boldsymbol{x}_0)\overleftarrow{p}_{0|t}(\boldsymbol{x}_0|\boldsymbol{H}_t)\mathrm{d}\boldsymbol{x}_0 \\ &= \int \mathbb{1}_{x\in(a,b)}(\boldsymbol{x}_0)\mathcal{N}(x|\hat{\mu}_{0|t}(\boldsymbol{H}_t), \hat{\sigma}_{0|t}(\boldsymbol{H}_t)^2)\mathrm{d}\boldsymbol{x}_0 \\ &= \frac{1}{\hat{\sigma}_{0|t}(\boldsymbol{H}_t)} \frac{\phi\left(\frac{\boldsymbol{H}_t - \hat{\mu}_{0|t}(\boldsymbol{H}_t)}{\hat{\sigma}_{0|t}(\boldsymbol{H}_t)}\right)}{\Phi\left(\frac{b-\hat{\mu}_{0|t}(\boldsymbol{H}_t)}{\hat{\sigma}_{0|t}(\boldsymbol{H}_t)}\right) - \Phi\left(\frac{a-\hat{\mu}_{0|t}(\boldsymbol{H}_t)}{\hat{\sigma}_{0|t}(\boldsymbol{H}_t)}\right)} \end{aligned} \tag{23}$$

where $\phi(\xi) = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}\xi^2\right)$ is the pdf of a standard normal distribution, $\Phi(\xi) = \frac{1}{2}\left(1 + \mathrm{erf}(\xi/\sqrt{2})\right)$ its cumulative function. The corrective drift term due to the h-transform can then be computed via autograd. The unconditional score term can be computed in closed form.

B Algorithms

In this section, we reformulate multiple algorithms from the literature under our common framework as a reference for practitioners. In these algorithms, we use the following conventions: our dataset is drawn from the law $\mathcal{P}_{\text{data}}$, but we can only sample from the simpler law $\mathcal{P}_{\text{sampling}}$ at inference time, which is often chosen as multivariate standard normal $\mathcal{P}_{\text{sampling}} = \mathcal{N}(0,\mathbf{I})$. Therefore, we construct a forward noising process $\mathcal{P}_{\text{data}} \to \mathcal{P}_{\text{sampling}}$ that is parametrised via the noise schedule $\beta_t = \beta(t), \bar{\alpha}_t = \bar{\alpha}(t)$ and try to learn the reverse denoising process $\mathcal{P}_{\text{sampling}} \to \mathcal{P}_{\text{data}}$. Due to this notion of "forward", and to keep consistency with the literature on denoising diffusion models, we explicate the nomenclature $\mathcal{P}_{\text{data}} = \mathcal{P}_0$ and $\mathcal{P}_{\text{sampling}} = \mathcal{P}_T$.

There is an additional law $\mathcal{P}_{\text{noise}}$ that is sometimes confused with $\mathcal{P}_{\text{sampling}}$ since in practice both are often chosen as $\mathcal{N}(0, \mathbf{I})$, but they are two distinct laws that could in principle be different. $\mathcal{P}_{\text{noise}}$ is the law from which the noise added during the forward noising process as well as the during the reverse diffusion process is drawn from.

For reference, we first reiterate the unconditional DDPM training and sampling algorithms, followed by the various conditional methods. For each method, we highlight the differences to standard DDPM sampling or training in gray boxes for clarity.

B.1 Unconditional algorithms

Algorithm 1 | Unconditional training of denoising diffusion models [Ho et al., 2020]

Require: Dataset drawn from law $\mathcal{P}_{\text{data}} = \mathcal{P}_0$ ▷ Dataset law $\mathcal{P}_{\text{data}}$
Require: Noise schedule $\beta_t = \beta(t), \bar{\alpha}_t = \bar{\alpha}(t)$, parametrising process $\mathcal{P}_{\text{data}} \to \mathcal{P}_{\text{sampling}}$
Require: Untrained noise predictor function $\mathbf{f}_\theta(\mathbf{x}, t)$ with parameters θ

repeat
  $\mathbf{x}_0 \sim \mathcal{P}_0 = \mathcal{P}_{\text{data}}$
  $t \sim \text{Uniform}(\{1, ..., T\})$
  ▷ Forward noise sample, $\mathbf{x}_t \sim \bar{p}_{t|0}(\mathbf{x}_0)$ ◁
  $\boldsymbol{\varepsilon}_t \sim \mathcal{P}_{\text{noise}}$ ▷ Often Brownian motion, $\mathcal{P}_{\text{noise}} = \mathcal{N}(0, \mathbf{I})$
  $\mathbf{x}_t \leftarrow \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\varepsilon}_t$
  ▷ Estimate noise of noised sample ◁
  $\hat{\boldsymbol{\varepsilon}}_\theta \leftarrow \mathbf{f}_\theta(\mathbf{x}_t, t)$
  Take gradient descent step on
    $\nabla_\theta L(\boldsymbol{\varepsilon}_t, \hat{\boldsymbol{\varepsilon}}_\theta)$ ▷ Typically, loss $L(x_{\text{true}}, x_{\text{pred}}) = ||x_{\text{true}} - x_{\text{pred}}||^2$
until converged or max epoch reached

Algorithm 2 | Unconditional sampling with denoising diffusion models [Ho et al., 2020]

Require: Unconditionally trained noise predictor $\mathbf{f}_\theta(\mathbf{x}_t, t)$
Require: Noise schedule $\beta_t = \beta(t), \bar{\alpha}_t = \bar{\alpha}(t)$, parametrising process $\mathcal{P}_{\text{data}} \to \mathcal{P}_{\text{sampling}}$

▷ Sample a starting point $\mathbf{x}_T$ ◁
$\mathbf{x}_T \sim \mathcal{P}_T = \mathcal{P}_{\text{sampling}}$ ▷ Often $\mathcal{P}_T = \mathcal{N}(0, \mathbf{I})$
▷ Iteratively denoise for T steps ◁
for t in $(T, T-1, \ldots, 1)$ **do**
  ▷ Predict noise with learned network ◁
  $\hat{\boldsymbol{\varepsilon}}_\theta = \mathbf{f}_\theta(\mathbf{x}_t, t)$
  ▷ Denoise sample with learned reverse process $\mathbf{x}_{t-1} \sim \bar{p}_{t-1|t}(\mathbf{x}_t)$ ◁
  ▷ Perform reverse drift ◁
  $\mathbf{x}_{t-1} \leftarrow \frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\hat{\boldsymbol{\varepsilon}}_\theta\right)$
  ▷ Perform reverse diffusion, which is often Brownian motion in $\mathbb{R}^n$, i.e. $\mathcal{P}_{\text{noise}} = \mathcal{N}(0, \mathbf{I})$ ◁
  $\boldsymbol{\varepsilon}_t \sim \mathcal{P}_{\text{noise}}$ if $t > 1$ else $\boldsymbol{\varepsilon}_t \leftarrow 0$
  $\mathbf{x}_{t-1} \leftarrow \mathbf{x}_{t-1} + \sigma_t \boldsymbol{\varepsilon}_t$ ▷ A common choice is $\sigma_t = \beta(t)$
return $\mathbf{x}_0$

B.2 Conditional training

Algorithm 3 | Classifier-free conditional training [Ho and Salimans, 2022]

Require: Dataset drawn from $\mathcal{P}_{\text{data}}$ ▷ Dataset law $\mathcal{P}_{\text{data}}$ over data and auxiliary variable
Require: Noise schedule $\beta_t = \beta(t), \bar{\alpha}_t = \bar{\alpha}(t)$, parametrising process $\mathcal{P}_{\text{data}} \to \mathcal{P}_{\text{sampling}}$
Require: Untrained noise predictor function $\mathbf{f}_\theta(\mathbf{x}, t)$ with parameters θ

repeat
- $\mathbf{x}_0, \boldsymbol{y} \sim \mathcal{P}_0 = \mathcal{P}_{\text{data}}$
- $\boldsymbol{\varepsilon}_t \sim \mathcal{P}_{\text{noise}}$ ▷ Often Brownian motion, $\mathcal{P}_{\text{noise}} = \mathcal{N}(0, \mathbf{I})$
- $t \sim \text{Uniform}(\{1, ..., T\})$
- $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\varepsilon}_t$
- $\hat{\boldsymbol{\varepsilon}}_\theta = \mathbf{f}_\theta(\mathbf{x}_t, t, \boldsymbol{y})$
- Take gradient descent step on $\nabla_\theta L(\boldsymbol{\varepsilon}_t, \hat{\boldsymbol{\varepsilon}}_\theta)$ ▷ Typically, $L(x_{\text{true}}, x_{\text{pred}}) = ||x_{\text{true}} - x_{\text{pred}}||^2$

until converged or max epoch reached

Algorithm 4 | RFDiffusion conditional training [Watson et al., 2023]

Require: Dataset drawn from $\mathcal{P}_{\text{data}}$ ▷ Dataset law $\mathcal{P}_{\text{data}}$
Require: Noise schedule $\beta_t = \beta(t), \bar{\alpha}_t = \bar{\alpha}(t)$, parametrising process $\mathcal{P}_{\text{data}} \to \mathcal{P}_{\text{sampling}}$
Require: Untrained noise predictor function $\mathbf{f}_\theta(\mathbf{x}, t, M)$ with parameters θ

repeat
- $\mathbf{x}_0 \sim \mathcal{P}_0 = \mathcal{P}_{\text{data}}$
- $t \sim \text{Uniform}(\{1, ..., T\})$
- $\mathbf{x}_0^{[M]} \cup \mathbf{x}_0^{[\setminus M]} \leftarrow \mathbf{x}_0$ ▷ Randomly partition data point into motif and rest
- ▷ Forward noise the non-motif rest via sampling from $\vec{p}_{0|t}(\mathbf{x}_0)$ ◁
- $\boldsymbol{\varepsilon}_t \sim \mathcal{P}_{\text{noise}}$
- $\mathbf{x}_t^{[\setminus M]} \leftarrow \sqrt{\bar{\alpha}_t}\mathbf{x}_0^{[\setminus M]} + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\varepsilon}_t^{[\setminus M]}$
- ▷ Combine unnoised motif with noised rest and set timestep of motif part to 0 ◁
- $\mathbf{x}_t \leftarrow \mathbf{x}_0^{[M]} \cup \mathbf{x}_t^{[\setminus M]}$
- $t^{[M]} \leftarrow 0$
- $\hat{\boldsymbol{\varepsilon}}_\theta \leftarrow \mathbf{f}_\theta(\mathbf{x_t}, t, M)$ ▷ Estimate noise of sample with noised rest
- Take gradient descent step on $\nabla_\theta L(\boldsymbol{\varepsilon}, \hat{\boldsymbol{\varepsilon}}_\theta)$ ▷ Typically, $L(x_{\text{true}}, x_{\text{pred}}) = ||x_{\text{true}} - x_{\text{pred}}||^2$

until converged or max epoch reached

Algorithm 5 | Amortised training – i.e. Doob's h-transform conditional training (new)

Require: Dataset drawn from $\mathcal{P}_{\text{data}}$ ▷ Dataset law $\mathcal{P}_{\text{data}}$
Require: Noise schedule $\beta_t = \beta(t), \bar{\alpha}_t = \bar{\alpha}(t)$, parametrising process $\mathcal{P}_{\text{data}} \to \mathcal{P}_{\text{sampling}}$
Require: Untrained noise predictor function $\mathbf{f}_\theta(\mathbf{x}, t, \mathbf{x}^{[M]}, M)$ with parameters θ

repeat
- $\mathbf{x}_0 \sim \mathcal{P}_0 = \mathcal{P}_{\text{data}}$
- $t \sim \text{Uniform}(\{1, ..., T\})$
- $\mathbf{x}_0^{[M]} \cup \mathbf{x}_0^{[\setminus M]} \leftarrow \mathbf{x}_0$ ▷ Randomly partition data point into motif and rest
- ▷ Forward noise full sample via sampling from $\vec{p}_{0|t}(\mathbf{x}_0)$ ◁
- $\boldsymbol{\varepsilon}_t \sim \mathcal{P}_{\text{noise}}$
- $\mathbf{x}_t \leftarrow \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\varepsilon}_t$
- ▷ Estimate noise of sample with original motif as additional input ◁
- $\hat{\boldsymbol{\varepsilon}}_\theta \leftarrow \mathbf{f}_\theta(\mathbf{x}_t, t, \mathbf{x}_0^{[M]}, M)$
- Take gradient descent step on $\nabla_\theta L(\boldsymbol{\varepsilon}, \hat{\boldsymbol{\varepsilon}}_\theta)$ ▷ Typically, $L(x_{\text{true}}, x_{\text{pred}}) = ||x_{\text{true}} - x_{\text{pred}}||^2$

until converged or max epoch reached

B.3 Conditional sampling

Algorithm 6 | RFDiffusion conditional sampling [Watson et al., 2023]

Require: Conditionally trained noise predictor $\mathbf{f}_\theta(\mathbf{x}, t, M)$
Require: Target motif/context $\mathbf{x}_0^{[M]}$
Require: Noise schedule $\beta_t = \beta(t), \bar{\alpha}_t = \bar{\alpha}(t)$, parametrising process $\mathcal{P}_{\text{data}} \to \mathcal{P}_{\text{sampling}}$

▷ Sample a starting point $\mathbf{x}_T$ ◁
$\mathbf{x}_T \sim \mathcal{P}_T = \mathcal{P}_{\text{sampling}}$

▷ Iteratively denoise for T steps ▷ Often $\mathcal{P}_T = \mathcal{N}(0, \mathbf{I})$ ◁
for t in $(T, T-1, \ldots, 1)$ **do**
- ▷ Overwrite motif variables with target motif and reset their time parameter ◁
- ▷ Note: Original RFDiffusion zero-centers $\mathbf{x}_t$ and $\mathbf{x}_0^{[M]}$ individually for equivariance. ◁
- $\mathbf{x}_t^{[M]} \leftarrow \mathbf{x}_0^{[M]}$ ▷ Set noisy motif to unnoised motif
- $t^{[M]} \leftarrow 0$ ▷ Set timesteps for motif to 0
- $\hat{\boldsymbol{\varepsilon}}_\theta = \mathbf{f}_\theta(\mathbf{x}_t, t, M)$ ▷ Predict noise with learned network
- ▷ Denoise sample with learned reverse process $\mathbf{x}_{t-1} \sim \overleftarrow{p}_{t-1|t}(\mathbf{x}_t)$ ◁
- ▷ Perform reverse drift ◁
- $\mathbf{x}_{t-1} \leftarrow \frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\hat{\boldsymbol{\varepsilon}}_\theta\right)$
- ▷ Perform reverse diffusion, which is often Brownian motion in $\mathbb{R}^n$, i.e. $\mathcal{P}_{\text{noise}} = \mathcal{N}(0, \mathbf{I})$ ◁
- $\boldsymbol{\varepsilon}_t \sim \mathcal{P}_{\text{noise}}$ if $t > 1$ else $\boldsymbol{\varepsilon}_t \leftarrow 0$
- $\mathbf{x}_{t-1} \leftarrow \mathbf{x}_{t-1} + \sigma_t \boldsymbol{\varepsilon}_t$ ▷ A common choice is $\sigma_t = \beta(t)$

return $\mathbf{x}_0$

Algorithm 7 | Replacement conditional sampling

Require: Unconditionally trained noise predictor $\mathbf{f}_\theta(\mathbf{x}_t, t)$
Require: Noise schedule $\beta_t = \beta(t), \bar{\alpha}_t = \bar{\alpha}(t)$, parametrising process $\mathcal{P}_{\text{data}} \to \mathcal{P}_{\text{sampling}}$
Require: Target motif $\mathbf{x}_0^{[M]}$

▷ Sample a starting point $\mathbf{x}_T$ ◁
$\mathbf{x}_T \sim \mathcal{P}_T = \mathcal{P}_{\text{sampling}}$

▷ Iteratively denoise for T steps ▷ Often $\mathcal{P}_T = \mathcal{N}(0, \mathbf{I})$ ◁
for t in $(T, T-1, \ldots, 1)$ **do**
- ▷ Predict noise with learned network ◁
- $\hat{\boldsymbol{\varepsilon}}_\theta \leftarrow \mathbf{f}_\theta(\mathbf{x}_t, t)$
- ▷ Denoise sample with learned reverse process $\mathbf{x}_{t-1} \sim \overleftarrow{p}_{t-1|t}(\mathbf{x}_t)$ ◁
- ▷ Perform reverse drift ◁
- $\mathbf{x}_{t-1} \leftarrow \frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\hat{\boldsymbol{\varepsilon}}_\theta\right)$
- ▷ Perform reverse diffusion, which is often Brownian motion in $\mathbb{R}^n$, i.e. $\mathcal{P}_{\text{noise}} = \mathcal{N}(0, \mathbf{I})$ ◁
- $\boldsymbol{\varepsilon}_t \sim \mathcal{P}_{\text{noise}}$ if $t > 1$ else $\boldsymbol{\varepsilon}_t \leftarrow 0$
- $\mathbf{x}_{t-1} \leftarrow \mathbf{x}_{t-1} + \sigma_t \boldsymbol{\varepsilon}_t$ ▷ A common choice is $\sigma_t = \beta(t)$
- ▷ Forward noise the target motif $\mathbf{x}_{t-1}^{[M]} \sim \overrightarrow{p}_{0|t-1}(\mathbf{x}_0^{[M]})$ ◁
- $\boldsymbol{\eta}_{t-1} \sim \mathcal{P}_{\text{noise}}$ if $t > 1$ else $\boldsymbol{\eta}_{t-1} \leftarrow 0$
- $\mathbf{x}_{t-1}^{[M]} \leftarrow \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0^{[M]} + \sqrt{1-\bar{\alpha}_{t-1}}\boldsymbol{\eta}_{t-1}$
- $\mathbf{x}_{t-1} \leftarrow \mathbf{x}_{t-1}^{[\setminus M]} \cup \mathbf{x}_{t-1}^{[M]}$ ▷ Insert noised motif into current sample

return $\mathbf{x}_0$

Algorithm 8 | Reconstruction Guidance (i.e. Moment Matching (MM) Approximation to h-transform)

Require: Unconditionally trained noise predictor $\mathbf{f}_\theta(\mathbf{x}_t, t)$, target motif/context $\mathbf{x}_0^{[M]}$.
Require: Noise schedule $\beta_t = \beta(t), \bar{\alpha}_t = \bar{\alpha}(t)$, parameterising process $\mathcal{P}_{\text{data}} \to \mathcal{P}_{\text{sampling}}$
Require: Guidance scale (schedule) $\gamma_t = \gamma(t)$
Require: Conditioning loss $l(x_{\text{true}}, x_{\text{pred}})$. e.g, Gaussian MM $l(x_{\text{true}}, x_{\text{pred}}) = ||x_{\text{true}} - x_{\text{pred}}||^2$

▷ Sample a starting point $\mathbf{x}_T$ ◁
$\mathbf{x}_T \sim \mathcal{P}_T = \mathcal{P}_{\text{sampling}}$ ▷ Often $\mathcal{P}_T = \mathcal{N}(0, \mathbf{I})$

▷ Iteratively denoise and condition for T steps ◁
for t in $(T, T-1, \dots, 1)$ **do**
  $\hat{\boldsymbol{\varepsilon}}_\theta = \mathbf{f}_\theta(\mathbf{x}_t, t)$ ▷ Predict noise with learned network

  ▷ Estimate current denoised estimate via Tweedie's formula ◁
  $\hat{\mathbf{x}}_0(\mathbf{x}_t, \hat{\boldsymbol{\varepsilon}}_\theta) \leftarrow \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\hat{\boldsymbol{\varepsilon}}_\theta)$ ▷ c.f. also eq. 15 in Ho et al. [2020]

  ▷ Perform gradient descent step towards condition on motif dimensions M ◁
  $\mathbf{x}_t \leftarrow \mathbf{x}_t - \gamma_t \nabla_\theta l(\mathbf{x}_0^{[M]}, \hat{\mathbf{x}}_0^{[M]}(\mathbf{x}_t, \hat{\boldsymbol{\varepsilon}}_\theta))$ ▷ Requires backprop through $\mathbf{f}_\theta$

  ▷ Denoise sample with learned reverse process $\mathbf{x}_{t-1} \sim \bar{p}_{t-1|t}(\mathbf{x}_t)$ ◁
  $\mathbf{x}_{t-1} \leftarrow (1-\beta_t)^{-1/2}\left(\mathbf{x}_t - \beta_t(1-\bar{\alpha}_t)^{-1/2}\hat{\boldsymbol{\varepsilon}}_\theta\right)$ ▷ Perform reverse drift
  ▷ Perform reverse diffusion, which is often Brownian motion in $\mathbb{R}^n$, i.e. $\mathcal{P}_{\text{noise}} = \mathcal{N}(0, \mathbf{I})$ ◁
  $\boldsymbol{\varepsilon}_t \sim \mathcal{P}_{\text{noise}}$ if $t > 1$ else $\boldsymbol{\varepsilon}_t \leftarrow 0$
  $\mathbf{x}_{t-1} \leftarrow \mathbf{x}_{t-1} + \sigma_t \boldsymbol{\varepsilon}_t$ ▷ A common choice is $\sigma_t = \beta(t)$
return $\mathbf{x}_0$

Algorithm 9 | Replacement conditional Sampling [Lugmayr et al., 2022]

Require: Unconditionally trained noise predictor $\mathbf{f}_\theta(\mathbf{x}_t, t)$
Require: Noise schedule $\beta_t = \beta(t), \bar{\alpha}_t = \bar{\alpha}(t)$, parametrising process $\mathcal{P}_{\text{data}} \to \mathcal{P}_{\text{sampling}}$
Require: Target motif $\mathbf{x}_0^{[M]}$

▷ Sample a starting point $\mathbf{x}_T$ ◁
$\mathbf{x}_T \sim \mathcal{P}_T = \mathcal{P}_{\text{sampling}}$

▷ Iteratively denoise for T steps ▷ Often $\mathcal{P}_T = \mathcal{N}(0, \mathbf{I})$ ◁
for t in $(T, T-1, \dots, 1)$ **do** ▷ T time steps
  for r in $1, \dots, R$ **do** ▷ R repaint steps
    ▷ Predict noise with learned network ◁
    $\hat{\boldsymbol{\varepsilon}}_\theta \leftarrow \mathbf{f}_\theta(\mathbf{x}_t, t)$

    ▷ Denoise sample with learned reverse process $\mathbf{x}_{t-1} \sim \bar{p}_{t-1|t}(\mathbf{x}_t)$ ◁
    $\mathbf{x}_{t-1} \leftarrow (1-\beta_t)^{-1/2}\left(\mathbf{x}_t - \beta_t(1-\bar{\alpha}_t)^{-1/2}\hat{\boldsymbol{\varepsilon}}_\theta\right)$ ▷ Perform reverse drift
    ▷ Perform reverse diffusion, often Brownian motion in $\mathbb{R}^n$, i.e. $\mathcal{P}_{\text{noise}} = \mathcal{N}(0, \mathbf{I})$ ◁
    $\boldsymbol{\varepsilon}_t \sim \mathcal{P}_{\text{noise}}$ if $t > 1$ else $\boldsymbol{\varepsilon}_t \leftarrow 0$
    $\mathbf{x}_{t-1} \leftarrow \mathbf{x}_{t-1} + \sigma_t \boldsymbol{\varepsilon}_t$ ▷ A common choice is $\sigma_t = \beta(t)$

    ▷ Forward noise the target motif $\mathbf{x}_{t-1}^{[M]} \sim \vec{p}_{0|t-1}(\mathbf{x}_0^{[M]})$ ◁
    $\boldsymbol{\eta}_{t-1} \sim \mathcal{P}_{\text{noise}}$ if $t > 1$ else $\boldsymbol{\eta}_{t-1} \leftarrow 0$
    $\mathbf{x}_{t-1}^{[M]} \leftarrow \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0^{[M]} + \sqrt{1-\bar{\alpha}_{t-1}}\boldsymbol{\eta}_{t-1}$
    $\mathbf{x}_{t-1} \leftarrow \mathbf{x}_{t-1}^{[\setminus M]} \cup \mathbf{x}_{t-1}^{[M]}$ ▷ Insert noised motif into current sample
    if $r < R$ and $t > 1$ **then** ▷ Forward noise sample from $t-1$ to t, $\mathbf{x}_t \sim \vec{p}_{t|t-1}(\mathbf{x}_{t-1})$
      $\boldsymbol{\zeta}_{t-1} \sim \mathcal{P}_{\text{noise}}$
      $\mathbf{x}_t \leftarrow \sqrt{1-\beta_{t-1}}\mathbf{x}_{t-1} + \sqrt{\beta_{t-1}}\boldsymbol{\zeta}_{t-1}$
return $\mathbf{x}_0$

C Amortised learning of Doob's transform

C.1 Proof of proposition 2.5

Proof. (Informal) Via the mean squared error property of the conditional expectation the minimiser is given by:

$$f_t^*(\boldsymbol{h}, \boldsymbol{y}, \mathcal{A}) = \mathbb{E}\left[\nabla_{\boldsymbol{X}_t} \ln \vec{p}_{t|0}(\boldsymbol{X}_t|\boldsymbol{X}_0)|\boldsymbol{Y} = \boldsymbol{y}, \boldsymbol{X}_t = \boldsymbol{h}\right] \tag{24}$$

Then:

$$\begin{aligned} f_t^*(\boldsymbol{h}, \boldsymbol{y}, \mathcal{A}) &= \int \nabla_{\boldsymbol{h}} \ln \vec{p}_{t|0}(\boldsymbol{h}|\boldsymbol{X}_0)\vec{p}_{0|t}(\boldsymbol{X}_0|\boldsymbol{X}_t = \boldsymbol{h}, \boldsymbol{Y} = \boldsymbol{y})\mathrm{d}\boldsymbol{X}_0 \\ &= \int \frac{\nabla_{\boldsymbol{h}}\vec{p}_{t|0}(\boldsymbol{h}|\boldsymbol{X}_0)}{\vec{p}_{0|t}(\boldsymbol{h}|\boldsymbol{X}_0)} \frac{\vec{p}_{t|0}(\boldsymbol{X}_t = \boldsymbol{h}|\boldsymbol{X}_0, \boldsymbol{Y} = \boldsymbol{y})p(\boldsymbol{X}_0|\boldsymbol{Y} = \boldsymbol{y})}{p(\boldsymbol{X}_t = \boldsymbol{h}|\boldsymbol{Y} = \boldsymbol{y})}\mathrm{d}\boldsymbol{X}_0 \\ &= \frac{1}{p(\boldsymbol{X}_t = \boldsymbol{h}|\boldsymbol{Y} = \boldsymbol{y})} \int \frac{\nabla_{\boldsymbol{h}}\vec{p}_{t|0}(\boldsymbol{h}|\boldsymbol{X}_0)}{\vec{p}_{0|t}(\boldsymbol{h}|\boldsymbol{X}_0)}\vec{p}_{t|0}(\boldsymbol{X}_t = \boldsymbol{h}|\boldsymbol{X}_0)p(\boldsymbol{X}_0|\boldsymbol{Y} = \boldsymbol{y})\mathrm{d}\boldsymbol{X}_0 \\ &= \frac{1}{p(\boldsymbol{X}_t = \boldsymbol{h}|\boldsymbol{Y} = \boldsymbol{y})}\nabla_{\boldsymbol{h}} \int \vec{p}_{t|0}(\boldsymbol{h}|\boldsymbol{X}_0)p(\boldsymbol{X}_0|\boldsymbol{Y} = \boldsymbol{y})\mathrm{d}\boldsymbol{X}_0 \\ &= \frac{1}{\vec{p}(\boldsymbol{X}_t = \boldsymbol{h}|\boldsymbol{Y} = \boldsymbol{y})}\nabla_{\boldsymbol{h}}\vec{p}(\boldsymbol{X}_t = \boldsymbol{h}|\mathcal{A}\boldsymbol{X}_0 = \boldsymbol{y}) = \nabla_{\boldsymbol{h}} \ln \vec{p}(\boldsymbol{X}_t = \boldsymbol{h}|\boldsymbol{Y} = \boldsymbol{y}), \end{aligned}$$

□

D Experimental details: image experiments

In the image experiment, we use the DDPM [Ho et al., 2020] formulation for the diffusion model with $N = 1000$ steps, a linear β-schedule with $\beta_0 = 10^{-4}$ and $\beta_N = 2 \cdot 10^{-2}$.

Data We focus on the CELEBA [Liu et al., 2015] and FLOWERS [Nilsback and Zisserman, 2008] image datasets. For each of these datasets, we follow the same preprocessing procedure consisting of centrally cropping the image to size 64×64, and rescaling to pixel values $[-1, 1]$. We use this information to also clip our model's prediction.

Noise model The noise model ϵ_θ consists of a UNET architecture with four downsampling blocks consisting of 2d convolutional layers of dimensionality 128, 256, 384 and 512, respectively. We apply attention in the middle layers of the UNet with four heads. Throughout the network, we use the SiLU activation function, no dropout and group normalisation layers. The amortised network differs from the unconditional network in the fact that it accepts as input twice the number of channels (six instead of only three RGB channels). The unconditional models operate directly on the three RGB channels while the amortised network operates on the RBG channels, the mask and the condition. We can represent the mask and the condition information, however, into a single input with the same dimension as the image. The values of this input will be equal to the condition when the mask is 1 and set to a padding value of -2 where the mask is 0. We concatenate the image $\mathbb{R}^{3\times H\times W}$ with the condition and mask input of size $\mathbb{R}^{3\times H\times W}$ into an image with six channels. Due to this minor difference, our amortised network has 68.159M parameters while the unconditional networks have 68.156M parameters (roughly 0.005% fewer).

Methods In the amortised setting we follow Alg. 5. In 90% of the training steps, we pass a condition to the network. The other 10% contains a mask consisting of only 0's. For the reconstruction guidance method, we use a guidance term of $\gamma = 10.0$.

Metrics We measure the performance of the methods using mean squared error (MSE) and the perceptual metric LPIPS. Both these metrics compare the similarity between the original image (from which a patch was taken) and the conditional sample. For each metric, we compute the mean across 64 test images and repeat the experiment 5 times to get error estimates.