A Multi-Resolution Framework for U-Nets with Applications to Hierarchical VAEs

Fabian Falck *[1,3,4] **Christopher Williams** *[,1] **Dominic Danks** [2,4] **George Deligiannidis** [1]
Christopher Yau [1,3,4] **Chris Holmes** [1,3,4] **Arnaud Doucet** [1] **Matthew Willetts** [4]
[1]University of Oxford [2]University of Birmingham
[3]Health Data Research UK [4]The Alan Turing Institute
`{fabian.falck, williams, deligian, cholmes, doucet}@stats.ox.ac.uk,`
`{ddanks, cyau, mwilletts}@turing.ac.uk}`

Abstract

U-Net architectures are ubiquitous in state-of-the-art deep learning, however their regularisation properties and relationship to wavelets are understudied. In this paper, we formulate a multi-resolution framework which identifies U-Nets as finite-dimensional truncations of models on an infinite-dimensional function space. We provide theoretical results which prove that average pooling corresponds to projection within the space of square-integrable functions and show that U-Nets with average pooling implicitly learn a Haar wavelet basis representation of the data. We then leverage our framework to identify state-of-the-art hierarchical VAEs (HVAEs), which have a U-Net architecture, as a type of two-step forward Euler discretisation of multi-resolution diffusion processes which flow from a point mass, introducing sampling instabilities. We also demonstrate that HVAEs learn a representation of time which allows for improved parameter efficiency through weight-sharing. We use this observation to achieve state-of-the-art HVAE performance with half the number of parameters of existing models, exploiting the properties of our continuous-time formulation.

1 Introduction

U-Net architectures are extensively utilised in modern deep learning models. First developed for image segmentation in biomedical applications [1], U-Nets have been widely applied for text-to-image models [2], image-to-image translation [3], image restoration [4, 5], super-resolution [6], and multiview learning [7], amongst other tasks [8]. They also form a core building block as the neural architecture of choice in state-of-the-art generative models, particularly for images, such as HVAEs [9, 10, 11, 12] and diffusion models [2, 13, 14, 15, 16, 17, 18, 19, 20]. In spite of their empirical success, it is poorly understood why U-Nets work so well, and what regularisation they impose.

In likelihood-based generative modelling, various model classes are competing for superiority, including normalizing flows [21, 22], autoregressive models [23, 24], diffusion models, and hierarchical variational autoencoders (HVAEs), the latter two of which we focus on in this work. HVAEs form groups of latent variables with a conditional dependence structure, use a U-Net neural architecture, and are trained with the typical VAE ELBO objective (for a detailed introduction to HVAEs, see Appendix B). HVAEs show impressive synthesis results on facial images, and yield competitive likelihood performance, consistently outperforming the previously state-of-the-art autoregressive models, VAEs and flow models on computer vision benchmarks [9, 10]. HVAEs have undergone a journey

---

[*] Equal contribution.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

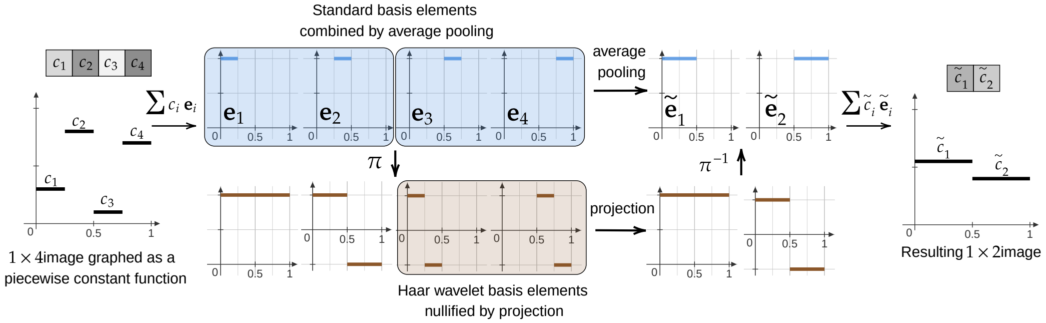


Figure 1: U-Nets with average pooling learn a Haar wavelet basis representation of the data.

of design iterations and architectural improvements in recent years, for example the introduction a deterministic backbone [25, 26, 27] and ResNet elements [28, 29] with shared parameters between the inference and generative model parts. There has also been a massive increase in the number of latent variables and overall stochastic depth, as well as the use of different types of residual cells in the decoder [9, 10] (see §4 and Fig. A.1 for a detailed discussion). However, a theoretical understanding of these choices is lacking. For instance, it has not been shown why a residual backbone may be beneficial, or what the specific cell structures in VDVAE [9] and NVAE [10] correspond to, or how they could be improved.

In this paper we provide a theoretical framework for understanding the latent spaces in U-Nets, and apply this to HVAEs specifically. Doing so allows us to relate HVAEs to diffusion processes, and also to motivate a new type of piecewise time-homogenous model which demonstrates state-of-the-art performance with approximately half the number of parameters of a VDVAE [9]. More formally, our contributions are as follows: **(a)** We provide a multi-resolution framework for U-Nets. We formally define U-Nets as as acting over a multi-resolution hierarchy of $L^2([0,1]^2)$. We prove that average pooling is a conjugate operation to projection in the Haar wavelet basis within $L^2([0,1]^2)$. We use this insight to show how U-Nets with average pooling implicitly learn a Haar wavelet basis representation of the data (see Fig. 1), helping to characterise the regularisation within U-Nets. **(b)** We apply this framework to state-of-the-art HVAEs as an example, identifying their residual cell structure as a type of two-step forward Euler discretisation of a multi-resolution diffusion bridge. We uncover that this diffusion process flows from a point mass, which causes instabilities, for instance during sampling, and identify parameter redundancies through our continuous-time formulation. Our framework both allows us to understand the heuristic choices of existing work in HVAEs and enables future work to optimise their design, for instance their residual cell. **(c)** In our experiments, we demonstrate these sampling instabilities and train HVAEs with the largest stochastic depth ever, achieving state-of-the-art performance with half the number of parameters by exploiting our theoretical insights. We explain these results by uncovering that HVAEs secretly represent time in their state and show that they use this information during training. We finally provide extensive ablation studies which, for instance, rule out other potential factors which correlate with stochastic depth, show the empirical gain of multiple resolutions, and find that Fourier features (which discrete-time diffusion models strongly benefit from [19]) do not improve performance in the HVAE setting.

2 The Multi-Resolution Framework

A grayscale image with infinite resolution can be thought of as the graph[2] of a two-dimensional function over the unit square. To store these infinitely-detailed images in computers, we project them to some finite resolution. These projections can still be thought of as the graphs of functions with support over the unit square, but they are piecewise constant on finitely many intervals or 'pixels', e.g. 512^2 pixels, and we store the function values obtained at these pixels in an array or 'grid'. The relationship between the finite-dimensional version and its infinitely-fine counterpart depends entirely on how we construct this projection to preserve the details we wish to keep. One approach is to prioritise preserving the large-scale details of our images, so unless closely inspected, the projection

---

[2] For a function $f(\cdot)$, its graph is the set $\bigcup_{x\in[0,1]^2}\{x, f(x)\}$.

is indistinguishable from the original. This can be achieved with a multi-resolution projection [30] of the image. In this section we introduce a *multi-resolution framework* for constructing neural network architectures that utilise such projections, prove what regularisation properties they impose, and show as an example how HVAEs with a U-Net [1] architecture can be interpreted in our framework. Proofs of all theorems in the form of an extended exposition of our framework can be found in Appendix A.

2.1 Multi-Resolution Framework: Definitions and Intuition

What makes a multi-resolution projection good at prioritising large-scale details can be informally explained through the following thought experiment. Imagine we have an image, represented as the graph of a function, and its finite-dimensional projection drawn on the wall. We look at the wall, start walking away from it and stop when the image and its projection are indistinguishable by eye. The number of steps we took away from the wall can be considered our measure of 'how far away' the approximation is from the underlying function. The goal of the multi-resolution projection is therefore to have to take as few steps away as possible. The reader is encouraged to physically conduct this experiment with the examples provided in Appendix B.1. We can formalise the aforementioned intuitions by defining a *multi-resolution hierarchy* [30] of sub-spaces we may project to:

Definition 1. [*Daubechies (1992)* [30]] Given a nested sequence of *approximation spaces* $\cdots \subset V_1 \subset V_0 \subset V_{-1} \subset \cdots$, $\{V_{-j}\}_{j\in\mathbb{Z}}$ is a *multi-resolution hierarchy* of the function space $L^2(\mathbb{R}^m)$ if: **(A1)** $\overline{\bigcup_{j\in\mathbb{Z}} V_{-j}} = L^2(\mathbb{R}^m)$; **(A2)** $\bigcap_{j\in\mathbb{Z}} V_{-j} = \{0\}$; **(A3)** $f(\cdot) \in V_{-j} \Leftrightarrow f(2^j \cdot) \in V_0$; **(A4)** $f(\cdot) \in V_0 \Leftrightarrow f(\cdot - n) \in V_0$ for $n \in \mathbb{Z}$. For a compact set $\mathbb{X} \subset \mathbb{R}^m$, a *multi-resolution hierarchy* of $L^2(\mathbb{X})$ is $\{V_{-j}\}_{j\in\mathbb{Z}}$ as defined above, restricting functions in V_{-j} to be supported on $\mathbb{X}$.

In Definition 1, the index j references how many steps we took in our thought experiment, so negative j corresponds to 'zooming in' on the images. The original image[3] is a member of $L^2([0,1]^2)$, the space of square-integrable functions on the unit square, and its finite projection to $2^j \cdot 2^j$ many pixels is a member of V_{-j}. Images can be represented as piecewise continuous functions in the subspaces $V_{-j} = \{f \in L^2([0,1]) \mid f|_{[2^{-j}\cdot k, 2^{-j}\cdot(k+1))} = c_k,\, k \in \{0,\ldots,2^j-1\},\, c_k \in \mathbb{R}\}$. The nesting property $V_{-j+1} \subset V_{-j}$ ensures that any image with $(2^{j-1})^2$ pixels can also be represented by $(2^j)^2$ pixels, but at a higher resolution. Assumption (**A1**) states that with infinitely many pixels, we can describe any infinitely detailed image. In contrast, (**A2**) says that with no pixels, we cannot approximate any images. Assumptions (**A3**) and (**A4**) allow us to form a basis for images in any V_{-j} if we know the basis of V_0. One basis made by extrapolating from V_0 in this way is known as a *wavelet basis* [30]. Wavelets have proven useful for representing images, for instance in the JPEG standard [31], and are constructed to be orthonormal.

Now suppose we have a probability measure ν_∞ over infinitely detailed images represented in $L^2([0,1]^2)$ and wish to represent it at a lower resolution. Similar to how we did for infinitely detailed images, we want to project the measure ν_∞ to a lower dimensional measure ν_j on the finite dimensional space V_{-j}. In extension to this, we want the ability to reverse this projection so that we may sample from the lower dimensional measure and create a generative model for ν_∞. We would like to again prioritise the presence of large-scale features of the original image within the lower dimensional samples. We do this by constructing a *multi-resolution bridge* from ν_∞ to ν_j, as defined below.

Definition 2. Let $\mathbb{X} \subset \mathbb{R}^m$ be compact, $\{V_{-j}\}_{j=0}^{\infty}$ be a multi-resolution hierarchy of scaled so $L^2(\mathbb{X}) = \overline{\bigcup_{j\in\mathbb{N}_0} V_{-j}}$ and $V_0 = \{0\}$. If $\mathbb{D}(L^2(\mathbb{X}))$ is the space of probability measures over $L^2(\mathbb{X})$, then a family of probability measures $\{\nu_t\}_{t\in[0,1]}$ on $L^2(\mathbb{X})$ is a *multi-resolution bridge* if:

(i) there exist increasing times $\mathcal{I} := \{t_j\}_{j\in\mathbb{N}_0}$ where $t_0 = 0$, $\lim_{j\to\infty} t_j = 1$, such that $s \in [t_j, t_{j+1})$ implies $\mathrm{supp}(\nu_s) \subset V_{-j}$, i.e $\nu_s \in \mathbb{D}(V_{-j})$; and,

(ii) for $s \in (0,1)$, the mapping $s \mapsto \nu_s$ is continuous for $s \in (t_j, t_{j+1})$ for some j.

The continuous time dependence in Definition 2 plays a movie of the measure ν_0 supported on V_0 growing to ν_∞, a measure on images with infinite resolution. At a time interval $[t_j, t_{j+1})$, the space V_{-j} which the measure is supported on is fixed. We may therefore define a finite-dimensional model

---

[3] We here focus on grayscale, squared images for simplicity, but note that our framework can be seamlessly extended to colour images with a Cartesian product $L^2([0,1]^2) \times L^2([0,1]^2) \times L^2([0,1]^2)$, and other continuous signals such as time series.

transporting probability measures within V_{-j}, but at t_{j+1} the support flows over to V_{-j-1}. Given a multi-resolution hierarchy, we may glue these finite models, each acting on a disjoint time interval, together in a unified fashion. In Theorem 1 we show this for the example of a continuous-time multi-resolution diffusion process truncated up until some time $t_J = T \in (0,1)$ and in the *standard basis* discussed in §2.2, which will be useful when viewing HVAEs as discretisations of diffusion processes on functions in §2.3.

Theorem 1. *Let $B_j : [t_j, t_{j+1}) \times \mathbb{D}(V_{-j}) \mapsto \mathbb{D}(V_{-j})$ be a linear operator (such as a diffusion transition kernel, see Appendix A) for $j < J$ with coefficients $\mu^{(j)}, \sigma^{(j)} : [t_j, t_{j+1}) \times V_{-j} \mapsto V_{-j}$, and define the natural extensions within V_{-J} in bold, i.e. $\boldsymbol{B}_j := B_j \oplus \boldsymbol{I}_{V_{-j}^{\perp}}$. Then the operator $\boldsymbol{B} : [0,T] \times \mathbb{D}(V_{-J}) \mapsto \mathbb{D}(V_{-J})$ and the coefficients $\boldsymbol{\mu}, \boldsymbol{\sigma} : [0,T] \times V_{-J} \mapsto V_{-J}$ given by*

$$\boldsymbol{B} := \sum_{j=0}^{J} \mathbb{1}_{[t_j, t_{j+1})} \cdot \boldsymbol{B}_j, \quad \boldsymbol{\mu} := \sum_{j=0}^{J} \mathbb{1}_{[t_j, t_{j+1})} \cdot \boldsymbol{\mu}^{(j)}, \quad \boldsymbol{\sigma} := \sum_{j=0}^{J} \mathbb{1}_{[t_j, t_{j+1})} \cdot \boldsymbol{\sigma}^{(j)},$$

induce a multi-resolution bridge of measures from the dynamics for $t \in [0,T]$ and on the standard basis as $dZ_t = \boldsymbol{\mu}_t(Z_t)dt + \boldsymbol{\sigma}_t(Z_t)dW_t$ (see Appendix A.4 for details) for $Z_t \in V_{-j}$ for $t \in [t_j, t_{j+1})$, i.e. a multi-resolution diffusion process.

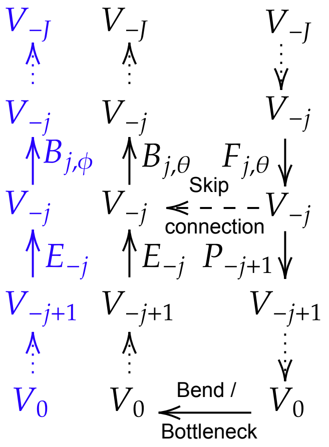


Figure 2: A U-Net in our multi-resolution framework. See Appendix B.2 for details.

The concept of a multi-resolution bridge will become important in Section 2.2 where we will show that current U-Net bottleneck structures used for unconditional sampling impose a multi-resolution bridge on the modelled densities. To preface this, we here provide a description of a U-Net within our framework, illustrated in 2. Consider $B_{j,\theta}, F_{j,\theta} : \mathbb{D}(V_{-j}) \to \mathbb{D}(V_{-j})$ as the forwards and backwards passes of a U-Net on resolution j. Further, let $P_{-j+1} : \mathbb{D}(V_{-j}) \to \mathbb{D}(V_{-j+1})$ and $E_{-j} : \mathbb{D}(V_{-j+1}) \to \mathbb{D}(V_{-j})$ be the projection (here: average pooling) and embedding maps (e.g. interpolation), respectively. When using an L^2-reconstruction error, a U-Net [1] architecture implicitly learns a sequence of models $\mathbf{B}_{j,\phi} : \mathbb{D}(V_{-j+1}) \times \mathbb{D}(V_{-j+1}^{\perp}) \mapsto \mathbb{D}(V_{-j})$ due to the orthogonal decomposition $V_{-j} = V_{-j+1} \oplus U_{-j+1}$ where $U_{-j+1} := V_{-j} \cap V_{-j+1}^{\perp}$. The backwards operator for the U-Net has a (bottleneck) input from $\mathbb{D}(V_{-j+1})$ and a (skip) input yielding information from $\mathbb{D}(V_{-j+1}^{\perp})$. A simple *bottleneck* map $U_{j,\theta} : \mathbb{D}(V_{-j}) \to \mathbb{D}(V_{-j})$ (without skip connection) is given by

$$U_{j,\theta} := B_{j,\theta} \circ E_{-j} \circ P_{-j+1} \circ F_{j,\theta}, \tag{1}$$

and a U-Net bottleneck with skip connection is

$$\mathbf{U}_{j,\phi} := B_{j,\phi}(E_{-j} \circ P_{-j+1} \circ F_{j,\theta}, F_{j,\theta}). \tag{2}$$

In HVAEs, the map $\mathbf{U}_{j,\phi} : \mathbb{D}(V_{-j}) \to \mathbb{D}(V_{-j})$ is trained to be the identity by minimising reconstruction error, and further shall approximate $U_{j,\theta} \approx \mathbf{U}_{j,\phi}$ via a KL divergence. The L^2-reconstruction error for $\mathbf{U}_{j,\phi}$ has an orthogonal partition of the inputs from $V_{-j+1} \times V_{-j}$, hence the only new subspace added is U_{-j+1}. As each orthogonal U_{-j+1} is added sequentially in HVAEs, the skip connections induce a multi-resolution structure of this hierarchical neural network structure. What we will investigate in Theorem 3 is the regularisation imposed on this partitioning by enforcing $U_{j,\theta} \approx \mathbf{U}_{j,\phi}$, as is often enforced for generative models with VAEs.

2.2 The regularisation property imposed by U-Net architectures with average pooling

Having defined U-Net architectures within our multi-resolution framework, we are now interested in the regularisation they impose. We do so by analysing a U-Net when skip connections are absent, so that we may better understand what information is transferred through each skip connection when they are present. In practice, a pixel representation of images is used when training U-Nets, which we henceforth call the *standard basis* (see A.2, Eq. (A.9)). The standard basis is not convenient to derive theoretical results. It is instead preferable to use a basis natural to the multi-resolution bridge imposed by a U-Net with a corresponding projection operation, which for average pooling is the *Haar (wavelet) basis* [32] (see Appendix A.2). The Haar basis, like a Fourier basis, is an orthonormal basis of $L^2(\mathbb{X})$ which has desirable L^2-approximation properties. We formalise this in Theorem 2 which states that the dimension reduction operation of average pooling in the standard basis is a conjugate operation to co-ordinate projection within the Haar basis (details are provided in Appendix A.2).

Theorem 2. *Given V_{-j} as in Definition 1, let $x \in V_{-j}$ be represented in the standard basis $\mathbf{E}_j$ and Haar basis $\mathbf{\Psi}_j$. Let $\pi_j : \mathbf{E}_j \mapsto \mathbf{\Psi}_j$ be the change of basis map illustrated in Fig. 3, then we have the conjugacy $\pi_{j-1} \circ pool_{-j,-j+1} = proj_{V_{-j+1}} \circ \pi_j$.*

Theorem 2 means that if we project an image from V_{-j} to V_{-j+1} in the Haar wavelet basis, we can alternatively view this as changing to the standard basis via π_j^{-1}, performing average pooling, and reverting back via π_{j-1} (see Figure 3). This is important because the Haar basis is orthonormal, which in Theorem 3 allows us to precisely quantify what information is lost with average pooling.

$$\begin{tikzcd}
(V_{-j}, \mathbf{E}_j) \arrow[r, "\text{pool}_{-j,-j+1}"] & (V_{-j+1}, \mathbf{E}_{j-1}) \arrow[d, "\pi_{j-1}"'] \\
(V_{-j}, \mathbf{\Psi}_j) \arrow[u, "\pi_j^{-1}"'] \arrow[r, dashed, "\text{proj}_{V_{-j+1}}"'] & (V_{-j+1}, \mathbf{\Psi}_{j-1})
\end{tikzcd}$$

Figure 3: The function space V_{-j} remains the same, but the basis changes under π_j.

Theorem 3. *Let $\{V_{-j}\}_{j=0}^J$ be a multi-resolution hierarchy of V_{-J} where $V_{-j} = V_{-j+1} \oplus U_{-j+1}$, and further, let $F_{j,\phi}$, $B_{j,\theta} : \mathbb{D}(V_{-j}) \mapsto \mathbb{D}(V_{-j})$ be such that $B_{j,\theta}F_{j,\phi} = I$ with parameters ϕ and θ. Define $\boldsymbol{F}_{j_1|j_2,\phi} := \boldsymbol{F}_{j_1,\phi} \circ \cdots \circ \boldsymbol{F}_{j_2,\phi}$ by $\boldsymbol{F}_{j,\phi} : \mathbb{D}(V_{-j}) \mapsto \mathbb{D}(V_{-j+1})$ where $\boldsymbol{F}_{j,\phi} := proj_{V_{-j+1}} \circ F_{j,\phi}$, and analogously define $\boldsymbol{B}_{j_1|j_2,\theta}$ with $\boldsymbol{B}_{j,\theta} := B_{j,\theta} \circ embd_{V_{-j}}$. Then, the sequence $\{\boldsymbol{B}_{1|j,\theta}(\boldsymbol{F}_{1|J,\phi}\nu_J)\}_{j=0}^J$ forms a discrete multi-resolution bridge between $\boldsymbol{F}_{1|J,\phi}\nu_J$ and $\boldsymbol{B}_{1|J,\theta}\boldsymbol{F}_{1|J,\phi}\nu_J$ at times $\{t_j\}_{j=1}^J$, and*

$$\sum_{j=0}^{J} \mathbb{E}_{X_{t_j} \sim \nu_j} \left\| \text{proj}_{U_{-j+1}} X_{t_j} \right\|_2^2 / \left\| \boldsymbol{F}_{j|J,\phi} \right\|_2^2 \leq (\mathcal{W}_2(\boldsymbol{B}_{1|J,\theta}\boldsymbol{F}_{1|J,\phi}\nu_J, \nu_J))^2, \tag{3}$$

where $\mathcal{W}_2$ is the Wasserstein-2 metric and $\left\|\boldsymbol{F}_{j|J,\phi}\right\|_2$ is the Lipschitz constant of $\boldsymbol{F}_{j|J,\phi}$.

Theorem 3 states that the bottleneck component of a U-Net pushes the latent data distribution to a finite multi-resolution basis, specifically a Haar basis when average pooling is used. To see this, note that the RHS of Eq. (A.65) is itself upper-bounded by the L^2-reconstruction error. This is because the Wasserstein-2 distance finds the infinimum over all possible couplings between the data and the 'reconstruction' measure, hence any coupling (induced by the learned model) bounds it. Note that models using a U-Net, for instance HVAEs or diffusion models, either directly or indirectly optimise for low reconstruction error in their loss function. The LHS of Eq. (A.65) represents what percentage of our data enters the orthogonal subspaces $\{U_{-j}\}_{j=0}^J$ which are (by Theorem 2) discarded by the bottleneck structure when using a U-Net architecture with average pooling. Theorem 3 thus shows that as we minimise the reconstruction error during training, we minimise the percentage of our data transported to the orthogonal sub-spaces $\{U_{-j}\}_{j=0}^J$. Consequently, the bottleneck architecture implicitly decomposes our data into a Haar wavelet decomposition, and when the skip connections are absent (like in a traditional auto-encoder) our network learns to compress the discarded subspaces U_{-j}. This characterises the regularisation imposed by a U-Net in the absence of skip connections.

These results suggest that U-Nets with average pooling provide a direct alternative to Fourier features [19, 33, 34, 35] which impose a Fourier basis, an alternative orthogonal basis on $L^2(\mathbb{X})$, as with skip connections the U-Net adds each subspace U_{-j} sequentially. However, unlike Fourier bases, there are in fact a multitude of wavelet bases which are all encompassed by the multi-resolution framework, and in particular, Theorem 3 pertains to all of them for the bottleneck structure. This opens the door to exploring conjugacy operations beyond average pooling induced by other wavelet bases optimised for specific data types.

2.3 Example: HVAEs as Diffusion Discretisations

To show what practical inferences we can derive from our multi-resolution framework, we apply it to analyse state-of-the-art HVAE architectures (see Appendix B.3 for an introduction), identifying parameter redundancies and instabilities. Here and in our experiments, we focus on VDVAEs [9]. We provide similar results for Markovian HVAEs [36, 37] and NVAEs [10] (see § 4) in Appendix A.5.

We start by inspecting VDVAEs. As we show next, we can tie the computations in VDVAE cells to the (forward and backward) operators $F_{j,\phi}$ and $B_{j,\theta}$ within our framework and identify them as a type of two-step forward Euler discretisation of a diffusion process. When used with a U-Net, as is done in VDVAE [9], this creates a *multi-resolution diffusion bridge* by Theorem 4.

Theorem 4. *Let $t_J := T \in (0,1)$ and consider (the p_θ backward pass) $\boldsymbol{B}_{\theta,1|J} : \mathbb{D}(V_{-J}) \mapsto \mathbb{D}(V_0)$ given in multi-resolution Markov process in the standard basis:*

$$dZ_t = (\overleftarrow{\mu}_{1,t}(Z_t) + \overleftarrow{\mu}_{2,t}(Z_t))dt + \overleftarrow{\sigma}_t(Z_t)dW_t, \tag{4}$$

where $\mathit{proj}_{U_{-j}} Z_{t_j} = 0$, $\|Z_t\|_2 > \|Z_s\|_2$ with $0 \leq s < t \leq T$ and for a measure $\nu_J \in \mathbb{D}(V_{-J})$ we have X_T, $Z_0 \sim \boldsymbol{F}_{\phi,J|1}\nu_J = \delta_{\{0\}}$. Then, VDVAEs approximates this process, and its residual cells are a type of two-step forward Euler discretisation of this Stochastic Differential Equation (SDE).

To better understand Theorem 4, we visualise its residual cell structure of VDVAEs and the corresponding discretisation steps in Fig. 4, and together those of NVAEs and Markovian HVAEs in Appendix A.5, Fig. A.1. Note that this process is Markov and increasing in the Z_i variables. Similar processes have been empirically observed as efficient first-order approximates to higher-order chains, for example the memory state in LSTMs [38]. Further, VDVAEs and NVAEs are even claimed to be high-order chains (see Eqs. (2,3) in [9] and Eq. (1) in [10]), despite only approximating this with a accumulative process.

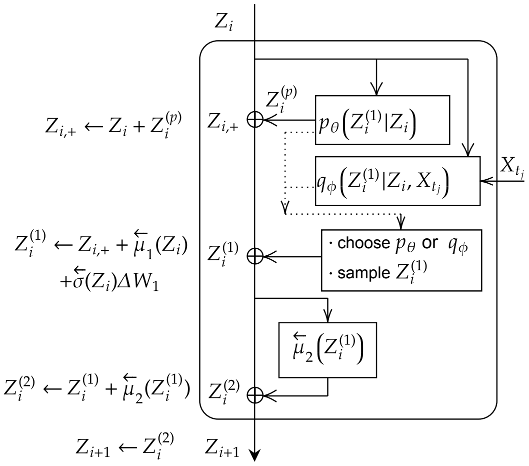

Z_i
$Z_{i,+} \leftarrow Z_i + Z_i^{(p)}$
$Z_{i,+}$
$Z_i^{(p)}$
$p_\theta\left(Z_i^{(1)}|Z_i\right)$
$q_\phi\left(Z_i^{(1)}|Z_i, X_{t_j}\right)$
X_{t_j}
$Z_i^{(1)} \leftarrow Z_{i,+} + \overleftarrow{\mu}_1(Z_i)$
$+\overleftarrow{\sigma}(Z_i)\Delta W_1$
$Z_i^{(1)}$
· choose p_θ or q_ϕ
· sample $Z_i^{(1)}$
$\overleftarrow{\mu}_2\left(Z_i^{(1)}\right)$
$Z_i^{(2)} \leftarrow Z_i^{(1)} + \overleftarrow{\mu}_2(Z_i^{(1)})$
$Z_i^{(2)}$
$Z_{i+1} \leftarrow Z_i^{(2)}$
Z_{i+1}

Figure 4: The VDVAE [9] cell is a type of two-step forward Euler discretisations of the continuous-time diffusion process in Eq. A.76. See Fig. A.1 for similar schemas on NVAE [10] and Markovian HVAE [36, 37].

To show how VDVAEs impose the growth of the Z_t, we prove that the bottleneck component of VDVAE's U-Net enforces $Z_0 = 0$. This is done by identifying that the measure ν_0, which a VDVAE connects to the data ν_∞ via a multi-resolution bridge, is a point mass on the zero function. Consequently the backward pass must grow from this, and the network learns this in a monotonic manner as we later confirm in our experiments (see §3.2).

Theorem 5. *Consider the SDE in Eq. (A.76), trained through the ELBO in Eq. B.101. Let $\tilde{\nu}_J$ denote the data measure and $\nu_0 = \delta_{\{0\}}$ be the initial multi-resolution bridge measure imposed by VDVAEs. If $q_{\phi,j}$ and $p_{\theta,j}$ are the densities of $B_{\phi,1|j}\boldsymbol{F}_{J|1}\tilde{\nu}_J$ and $B_{\theta,1|j}\nu_0$ respectively, then a VDVAE optimises the boundary condition $\min_{\theta,\phi} KL(q_{\phi,0,1}||q_{\phi,0}p_{\theta,1})$, where a double index indicates the joint distribution.*

Theorem 5 states that the VDVAE architecture forms multi-resolution bridge with the dynamics of Eq. (A.76), and connects our data distribution to the trivial measure on V_0: a Dirac mass at 0 as the pooling here cascades completely to V_0. From this insight, we can draw conclusions on instabilities and on parameter redundancies of this HVAE cell. There are two major instabilities in this discretisation. First, the imposed ν_0 is disastrously unstable as it enforces a data set, with potentially complicated topology to derive from a point-mass in U_{-j} at each $t = t_j$, and we observe the resulting sampling instability in our experiments in §3.3. We note that similar arguments are applicable in settings without a latent hierarchy imposed by a U-Net, see for instance [39]. The VDVAE architecture does, however, bolster this rate through the $Z_{i,+}^{(\sigma)}$ term, which is absent in NVAEs [10], in the discretisation steps of the residual cell. We empirically observe this controlled backward error in Fig. 6 [Right]. We refer to Fig. A.1 for a detailed comparison of HVAE cells and their corresponding discretisation of the coupled SDE in Eq. (A.76).

Moreover, the current form of VDVAEs is over-parameterised and not informed by this continuous-time formulation. The continuous time analogue of VDVAEs [9] in Theorem 4 has time dependent coefficients $\overleftarrow{\mu}_{t,1}$, $\overleftarrow{\mu}_{t,2}$, $\overleftarrow{\sigma}_t$. We hypothesise that the increasing diffusion process in Z_i implicitly encodes time. Hence, explicitly representing this in the model, for instance via ResNet blocks with independent parameterisations at every time step, is redundant, and a time-homogeneous model (see Appendix A.6 for a precise formulation)—practically speaking, performing weight-sharing across time time steps/layers—has the same expressivity, but requires far fewer parameters than the state-of-the-art VDVAE. It is worth noting that such a time-homogeneous model would make the parameterisation of HVAEs more similar to the recently popular (score-based) diffusion models [40, 41] which perform weight-sharing across all time steps.

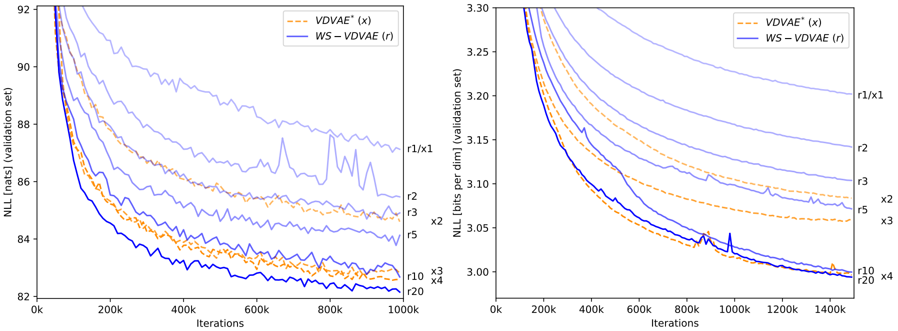


Figure 5: A small-scale study on parameter efficiency of HVAEs. We compare models with with 1,2,3 and 4 parameterised blocks per resolution ($\{x1, x2, x3, x4\}$) against models with a single parameterised block per resolution weight-shared $\{2, 3, 5, 10, 20\}$ times ($\{r2, r3, r5, r10, r20\}$). We report NLL (↓) measured on the validation set of MNIST [left] and CIFAR10 [right]. NLL performance increases with more weight-sharing repetitions and surpasses models without weight-sharing but with more parameters.

3 Experiments

In the following we probe the theoretical understanding of HVAEs gained through our framework, demonstrating its utility in four experimental analyses: (*a*) Improving parameter efficiency in HVAEs, (*b*) Time representation in HVAEs and how they make use of it, (*c*) Sampling instabilities in HVAEs, and (*d*) Ablation studies.

Table 1: A large-scale study of parameter efficiency in HVAEs. We compare our runs of VDVAE with original hyperparameters [9] (`VDVAE*`) against our weight-shared VDVAE (`WS-VDVAE`). While `WS-VDVAE`s have improved parameter efficiency by a factor of 2, they reach similar NLL as `VDVAE*` with the simple modification inspired by our framework (weight sharing). We note that a parameter count cannot be provided for VDM [19] as the code is not public and the manuscript does not specify it.

Dataset	Method	Type	#Params	NLL ↓
MNIST 28×28	`WS-VDVAE` (ours)	VAE	**232k**	≤ 79.98
	`VDVAE*` (ours)	VAE	339k	≤ 80.14
	NVAE [10]	VAE	33m	≤ 78.01
CIFAR10 32×32	`WS-VDVAE` (ours)	VAE	**25m**	≤ 2.88
	`WS-VDVAE` (ours)	VAE	39m	≤ 2.83
	`VDVAE*` (ours)	VAE	39m	≤ 2.87
	NVAE [10]	VAE	131m	≤ 2.91
	VDVAE [9]	VAE	39m	≤ 2.87
	VDM [19]	Diff	–	≤ 2.65
ImageNet 32×32	`WS-VDVAE` (ours)	VAE	**55m**	≤ 3.68
	`WS-VDVAE` (ours)	VAE	85m	≤ 3.65
	`VDVAE*` (ours)	VAE	119m	≤ 3.67
	NVAE [10]	VAE	268m	≤ 3.92
	VDVAE [9]	VAE	119m	≤ 3.80
	VDM [19]	Diff	–	≤ 3.72
CelebA 64×64	`WS-VDVAE` (ours)	VAE	**75m**	≤ 2.02
	`VDVAE*` (ours)	VAE	125m	≤ 2.02
	NVAE [10]	VAE	153m	≤ 2.03

We train HVAEs using VDVAE [9] as the basis model on five datasets: MNIST [42], CIFAR10 [43], two downsampled versions of ImageNet [44, 45], and CelebA [46], splitting each into a training, validation and test set (see Appendix D for details). In general, reported numeric values refer to Negative Log-Likelihood (NLL) in nats (MNIST) or bits per dim (all other datasets) on the test set at model convergence, if not stated otherwise. We note that performance on the validation and test set have similar trends in general. An optional *gradient checkpointing* implementation to trade in GPU memory for compute is discussed in Appendix F. Appendices F and G define the HVAE models we train, i.e. $p_\theta(\mathbf{z}_L), p_\theta(\mathbf{z}_l|\mathbf{z}_{>l}), q_\phi(\mathbf{z}_L|\mathbf{x}), q_\phi(\mathbf{z}_l|\mathbf{z}_{>l}, \mathbf{x})$ and $p_\theta(\mathbf{x}|\vec{\mathbf{z}})$, and present additional experimental details and results. We provide our PyTorch code base at https://github.com/FabianFalck/unet-vdvae (see Appendix C for details).

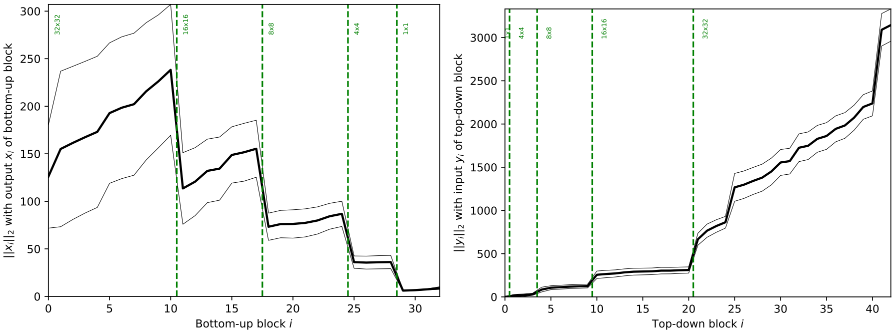


Figure 6: HVAEs secretly represent a notion of time: We measure the L_2-norm of the residual state for the [Left] forward/bottom-up pass and the [Right] backward/top-down pass over 10 batches with 100 data points each. In both plots, the thick, central line refers to the average and the thin, outer lines refer to ± 2 standard deviations.

3.1 "More from less": Improving parameter efficiency in HVAEs

In §2.3, we hypothesised that a time-homogeneous model has the same expressivity as a model with time-dependent coefficients, yet uses much less parameters. We start demonstrating this effect by weight-sharing ResNet blocks across time on a small scale. In Fig. 5, we train HVAEs on MNIST and CIFAR10 with $\{1, 2, 3, 4\}$ ResNet blocks (referred to as {`x1`, `x2`, `x3`, `x4`}) in each resolution with spatial dimensions $\{32^2, 16^2, 8^2, 4^2, 1^2\}$ (`VDVAE`*), and compare their performance when weight-sharing a single parameterised block per resolution $\{2, 3, 5, 10, 20\}$ times (referred to as {`r2`,`r3`,`r5`,`r10`,`r20`}; `WS-VDVAE`), excluding projection and embedding blocks. As hypothesised by our framework, yet very surprising in HVAEs, NLL after 1m iterations measured on the validation set gradually increases the more often blocks are repeated even though all weight-sharing models have an identical parameter count to the $x1$ model (MNIST: 107k, CIFAR10: 8.7m). Furthermore, the weight-sharing models often outperform or reach equal NLLs compared to `x2`, `x3`, `x4`, all of which have more parameters (MNIST: 140k; 173k; 206k. CIFAR10: 13.0m; 17.3m; 21.6m), yet fewer activations, latent variables, and number of timesteps at which the coupled SDE in Eq. (A.76) is discretised.

We now scale these findings up to large-scale hyperparameter configurations. We train VDVAE closely following the state-of-the-art hyperparameter configurations in [9], specifically with the same number of parameterised blocks and without weight-sharing (`VDVAE`*), and compare them against models with weight-sharing (`WS-VDVAE`) and fewer parameters, i.e. fewer parameterised blocks, in Table 1. On all four datasets, the weight-shared models achieve similar NLLs with fewer parameters compared to their counterparts without weight-sharing: We use 32%, 36%, 54%, and 40% less parameters on the four datasets reported in Table 1, respectively. For the larger runs, weight-sharing has diminishing returns on NLL as these already have many discretisation steps. To the best of our knowledge, our models achieve a new state-of-the-art performance in terms of NLL compared to any HVAE on CIFAR10, ImageNet32 and CelebA. Furthermore, our `WS-VDVAE` models have stochastic depths of 57, 105, 235, 125, respectively, the highest ever trained. In spite of these results, it is worth noting that current HVAEs, and VDVAE in particular remains notoriously unstable to train, partly due to the instabilities identified in Theorem 5, and finding the right hyperparameters helps, but cannot solve this.

3.2 HVAEs secretly represent time and make use of it

In §3.1, we showed how we can exploit insight on HVAEs through our framework to make HVAEs more parameter efficient. We now want to explain and understand this behavior further. In Fig. 6, we measure $\|Z_i\|_2$, the L_2-norm of the residual state at every backward/top-down block with index i, over several batches for models trained on MNIST (see Appendix G.2 for the corresponding figure of the forward/bottom-up pass, and similar results on CIFAR10 and ImageNet32). On average, we experience an increase in the state norm across time in every resolution, interleaved by discontinuous 'jumps' at the resolution transitions (projection or embedding) where the dimension of the residual state changes. This supports our claim in §2 that HVAEs discretise multi-resolution diffusion

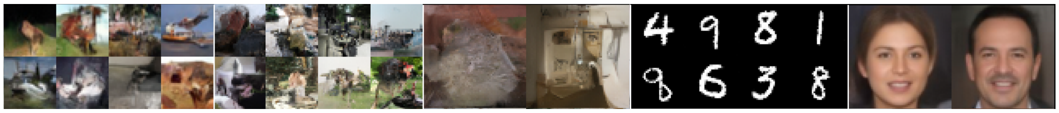

CIFAR10 $(t = 0.9)$ ImageNet32 $(t = 1)$ ImageNet64 $(t = 0.9)$ MNIST $(t = 0.9)$ CelebA $(t = 0.5)$

Figure 7: Unconditional samples (not cherry-picked) of `VDVAE`*. While samples on MNIST and CelebA demonstrate high fidelity and diversity, samples on CIFAR10, ImageNet32 and ImageNet64 are diverse, but are unrecognisable, demonstrating the instabilities identified by Theorem 1. Temperatures t are tuned for maximum fidelity.

processes which are increasing in the Z_i variables, and hence learn to represent a notion of time in their residual state.

It is now straightforward to ask how HVAEs benefit from this time representation during training: As we show in Table 2, when normalising the state by its norm at every forward and backward block during training, i.e. forcing a “flat line” in Fig. 6 [Left], learning deteriorates after a short while, resulting in poor NLL results compared to the runs with a regular, non-normalised residual state. This evidence confirms our earlier stated hypothesis: The time representation in ResNet-based HVAEs encodes information which recent HVAEs heavily rely on during learning.

3.3 Sampling instabilities in HVAEs

High fidelity unconditional samples of faces, e.g. from models trained on CelebA, cover the front pages of state-of-the-art HVAE papers [9, 10]. Here, we question whether face datasets are an appropriate benchmark for HVAEs. In Theorem 5, we identified the aforementioned state-of-the-art HVAEs as flow from a point mass, hypothesising instabilities during sampling. And indeed, when sampling from our trained `VDVAE`* with state-of-the-art configurations, we observe high fidelity and diversity samples on MNIST and CelebA, but unrecognisable, yet diverse samples on CIFAR10, ImageNet32 and ImageNet64, in spite of state-of-the-art test set NLLs (see Fig. 7 and Appendix G.3). We argue that MNIST and CelebA, i.e. numbers and faces, have a more uni-modal nature, and are in this sense easier to learn for a discretised multi-resolution process flowing to a point mass, which is uni-modal, than the other “in-the-wild”, multi-modal datasets. Trying to approximate the latter with the, in this case unsuitable, HVAE model leads to the sampling instabilities observed.

Table 2: NLL of HVAEs with and without normalisation of the residual state Z_i.

Residual state	NLL
MNIST	
Normalised (✗)	≤ 464.68
Non-normalised	≤ 81.69
CIFAR10	
Normalised (✗)	≤ 6.80
Non-normalised	≤ 2.93
ImageNet	
Normalised	≤ 6.76
Non-normalised	≤ 3.68

3.4 Ablation studies

We conducted several ablation studies which support our experimental results and further probe our multi-resolution framework for HVAEs. In this section we note key findings—a detailed account of all ablations can be found in Appendix G.4. In particular, we find that the number of latent variables, which correlates with stochastic depth, does not explain the performance observed in §3.1, supporting our claims. We further show that Fourier features do not provide a performance gain in HVAEs, in contrast to state-of-the-art diffusion models, where they significantly improve performance [19]. This is consistent with our framework’s finding that a U-Net architecture with pooling is already forced to learn a Haar wavelet basis representation of the data, hence introducing another basis does not add value. We also demonstrate that residual cells are crucial for the performance of HVAEs as they are able to approximate the dynamics of a diffusion process and impose an SDE structure into the model, empirically compare a multi-resolution bridge to a single-resolution model, and investigate synchronous vs. asynchronous processing in time between the forward and backward pass.

4 Related work

U-Nets. A U-Net [1] is an autoencoding architecture with multiple resolutions where skip connections enable information to pass between matched layers on opposite sides of the autoencoder's bottleneck. These connections also smooth out the network's loss landscape [47]. In the literature, U-Nets tend to be convolutional, and a wide range of different approaches have been used for up-sampling and down-sampling between resolutions, with many using average pooling for the down-sampling operation [13, 14, 16, 17, 19]. In this work, we focus on U-Nets as operators on measures interleaved by average pooling as the down-sampling operation (and a corresponding inclusion operation for up-sampling), and we formally characterise U-Nets in Section 2.1 and Appendix B.2. Prior to our work, the dimensionality-reducing bottleneck structure of U-Nets was widely acknowledged as being useful, however it was unclear what regularising properties a U-Net imposes. We provided these in §2.

HVAEs. The evolution of HVAEs can be seen as a quest for a parameterisation with more expressiveness than single-latent-layer VAEs [48], while achieving stable training dynamics that avoid common issues such as posterior collapse [36, 49] or exploding gradients. Early HVAEs such as LVAE condition each latent variable directly on only the previous one by taking samples forward [36, 37]. Such VAEs suffer from stability issues even for very small stochastic depths. *Nouveau VAEs (NVAE)* [10] and *Very Deep VAEs (VDVAE)* [9] combine the improvements of several earlier HVAE models (see Appendix B for details), while scaling up to larger stochastic depths. Both use ResNet-based backbones, sharing parameters between the generative and recognition parts of the model. VDVAE is the considerably simpler approach, in particular avoiding common tricks such as a warm-up deterministic autoencoder training phase or data-specific initialisation. VDVAE achieves a stochastic depth of up to 78, improving performance with more ResNet blocks. Worth noting is that while LVAE and NVAE use convolutions with appropriately chosen stride to jump between resolutions, VDVAE use average pooling. In all HVAEs to date, a theoretical underpinning which explains architectural choices, for instance the choice of residual cell, is missing, and we provided this in Section §2.3.

5 Conclusion

In this work, we introduced a multi-resolution framework for U-Nets. We provided theoretical results which uncover the regularisation property of the U-Nets bottleneck architecture with average pooling as implicitly learning a Haar wavelet representation of the data. We applied our framework to HVAEs, identifying them as multi-resolution diffusion processes flowing to a point mass. We characterised their backward cell as a type of two-step forward Euler discretisations, providing an alternative to score-matching to approoximate a continuous-time diffusion process [16, 18], and observed parameter redundancies and instabilities. We verified the latter theoretical insights in both small- and large-scale experiments, and in doing so trained the deepest ever HVAEs. We explained these results by showing that HVAEs learn a representation of time and performed extensive ablation studies.

An important limitation is that the proven regularisation property of U-Nets is limited to using average pooling as the down-sampling operation. Another limitation is that we only applied our framework to HVAEs, though it is possible to apply it to other model classes. It could also be argued that the lack of exhaustive hyperparameter optimisation performed is a limitation of the work as it may be possible to obtain improved results. We demonstrate, however, that simply adding weight-sharing to the hyperparameter settings given in the original VDVAE paper [9] leads to state-of-the-art performance with improved parameter efficiency, and hence view it as a strength of our results.

Acknowledgments and Disclosure of Funding

Fabian Falck acknowledges the receipt of studentship awards from the Health Data Research UK-The Alan Turing Institute Wellcome PhD Programme in Health Data Science (Grant Ref: 218529/Z/19/Z), and the Enrichment Scheme of The Alan Turing Institute under the EPSRC Grant EP/N510129/1. Chris Williams acknowledges support from the Defence Science and Technology (DST) Group and from a ESPRC DTP Studentship. Dominic Danks is supported by a Doctoral Studentship from The Alan Turing Institute under the EPSRC Grant EP/N510129/1. Christopher Yau is funded by a UKRI Turing AI Fellowship (Ref: EP/V023233/1). Chris Holmes acknowledges support from the Medical Research Council Programme Leaders award MC_UP_A390_1107, The Alan Turing Institute, Health Data Research, U.K., and the U.K. Engineering and Physical Sciences Research Council through the Bayes4Health programme grant. Arnaud Doucet acknowledges support of the UK Defence Science and Technology Laboratory (Dstl) and EPSRC grant EP/R013616/1. This is part of the collaboration between US DOD, UK MOD and UK EPSRC under the Multidisciplinary University Research Initiative. Arnaud Doucet also acknowledges support from the EPSRC grant EP/R034710/1. Matthew Willetts is grateful for the support of UCL Computer Science and The Alan Turing Institute.

The authors report no competing interests.

The three compute clusters used in this work were provided by the Alan Turing Institute, the Oxford Biomedical Research Computing (BMRC) facility, and the Baskerville Tier 2 HPC service (https://www.baskerville.ac.uk/) which we detail in the following. First, this research was supported in part through computational resources provided by The Alan Turing Institute under EPSRC grant EP/N510129/1 and with the help of a generous gift from Microsoft Corporation. Second, we used the Oxford BMRC facility, a joint development between the Wellcome Centre for Human Genetics and the Big Data Institute supported by Health Data Research UK and the NIHR Oxford Biomedical Research Centre. The views expressed are those of the author(s) and not necessarily those of the NHS, the NIHR or the Department of Health. Third, Baskerville was funded by the EPSRC and UKRI through the World Class Labs scheme (EP/T022221/1) and the Digital Research Infrastructure programme (EP/W032244/1) and is operated by Advanced Research Computing at the University of Birmingham.

We thank Tomas Lazauskas, Jim Madge and Oscar Giles from the Alan Turing Institute's Research Engineering team for their help and support. We thank Adam Huffman, Jonathan Diprose, Geoffrey Ferrari and Colin Freeman from the Biomedical Research Computing team at the University of Oxford for their help and support. We thank Haoting Zhang (University of Cambridge) for valuable comments on the implementation; Huiyu Wang (Johns Hopkins University) for a useful discussion on gradient checkpointing; and Ruining Li and Hanwen Zhu (University of Oxford) for kindly proofreading the manuscript.

References

[1] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[2] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022.

[3] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[4] Xiaojiao Mao, Chunhua Shen, and Yu-Bin Yang. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[5] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021.

[6] Yuyang Shi, Valentin De Bortoli, George Deligiannidis, and Arnaud Doucet. Conditional simulation using diffusion schrödinger bridges. *arXiv preprint arXiv:2202.13460*, 2022.

[7] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision – ECCV 2020*. Springer International Publishing, 2020.

[8] Zoe Landgraf, Fabian Falck, Michael Bloesch, Stefan Leutenegger, and Andrew J. Davison. Comparing view-based and map-based semantic labelling in real-time slam. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6884–6890, 2020.

[9] Rewon Child. Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images. In *International Conference on Learning Representations*, 2021.

[10] Arash Vahdat and Jan Kautz. NVAE: A deep hierarchical variational autoencoder. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[11] Louay Hazami, Rayhane Mama, and Ragavan Thurairatnam. Efficient-vdvae: Less is more. *arXiv preprint arXiv:2203.13751*, 2022.

[12] Simon AA Kohl, Bernardino Romera-Paredes, Klaus H Maier-Hein, Danilo Jimenez Rezende, SM Eslami, Pushmeet Kohli, Andrew Zisserman, and Olaf Ronneberger. A hierarchical probabilistic u-net for modeling multi-scale ambiguities. *arXiv preprint arXiv:1905.13077*, 2019.

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

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

[15] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. *arXiv preprint arXiv:2102.09672*, 2021.

[16] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[17] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[18] Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion schrödinger bridge with applications to score-based generative modeling. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[19] Diederik P. Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational Diffusion Models. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[20] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[21] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International Conference on Machine Learning*, 2019.

[22] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in Neural Information Processing Systems*, 31, 2018.

[23] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

[24] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in Neural Information Processing Systems*, 29, 2016.

[25] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, 2014.

[26] Shengjia Zhao, Jiaming Song, and Stefano Ermon. Learning hierarchical features from deep generative models. In *International Conference on Machine Learning*, 2017.

[27] Fabian Falck, Haoting Zhang, Matthew Willetts, George Nicholson, Christopher Yau, and Chris C Holmes. Multi-facet clustering variational autoencoders. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[28] Diederik P. Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, 2016.

[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[30] Ingrid Daubechies. *Ten lectures on wavelets*. SIAM, 1992.

[31] David Taubman and Michael Marcellin. *JPEG2000 image compression fundamentals, standards and practice: image compression fundamentals, standards and practice*, volume 642. Springer Science & Business Media, 2012.

[32] Alfred Haar. *Zur theorie der orthogonalen funktionensysteme*. Georg-August-Universitat, Gottingen., 1909.

[33] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[34] Haohan Wang, Xindi Wu, Zeyi Huang, and Eric P Xing. High-frequency component helps explain the generalization of convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8684–8694, 2020.

[35] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, volume 20, 2007.

[36] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[37] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

[38] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[39] Rob Cornish, Anthony Caterini, George Deligiannidis, and Arnaud Doucet. Relaxing bijectivity constraints with continuously indexed normalising flows. In *International conference on machine learning*, pages 2133–2143. PMLR, 2020.

[40] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[41] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, 2020.

[42] Yann LeCun, Corinna Cortes, and C. J. Burges. MNIST handwritten digit database. `http://yann.lecun.com/exdb/mnist/`, 2010.

[43] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[44] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[45] Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819*, 2017.

[46] 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.

[47] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[48] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

[49] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

[50] D La Torre and F Mendivil. The monge–kantorovich metric on multimeasures and self–similar multimeasures. *Set-Valued and Variational Analysis*, 23(2):319–331, 2015.

[51] Svetlozar T Rachev. *Probability metrics and the stability of stochastic models*, volume 269. Wiley, 1991.

[52] Stephane G Mallat. Multiresolution approximations and wavelet orthonormal bases of ^{2} (). *Transactions of the American mathematical society*, 315(1):69–87, 1989.

[53] Naftali Tishby, Fernando C. Pereira, and William Bialek. The Information Bottleneck Method. 2000.

[54] Lars Maaløe, Marco Fraccaro, Valentin Liévin, and Ole Winther. BIVA: A very deep hierarchy of latent variables for generative modeling. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[55] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016.

[56] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019.

[57] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, et al. Array programming with NumPy. *Nature*, 585(7825):357–362, 2020.

[58] Lukas Biewald. Experiment tracking with weights and biases, 2020. Software available from wandb.com.

[59] NVIDIA. Nvidia apex. `https://github.com/NVIDIA/apex`.

[60] Guido Van Rossum. *The Python Library Reference, release 3.8.2*. Python Software Foundation, 2020.

[61] J. D. Hunter. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[62] Almar et al. Klein. Imageio. `https://zenodo.org/record/6551868#.Yolo_5PMIhg`.

[63] Lisandro Dalcin and Yao-Lung L Fang. Mpi4py: Status update after 12 years of development. *Computing in Science & Engineering*, 23(4):47–54, 2021.

[64] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[65] P Umesh. Image processing in python. *CSI Communications*, 23, 2012.

[66] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. 80 million tiny images. `http://groups.csail.mit.edu/vision/TinyImages/`.

[67] Vinay Uday Prabhu and Abeba Birhane. Large image datasets: A pyrrhic win for computer vision? *arXiv preprint arXiv:2006.16923*, 2020.

[68] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Large-scale celebfaces attributes (celeba) dataset. 2022.

[69] Tomás Mikolov et al. Statistical language models based on neural networks. *Presentation at Google, Mountain View, 2nd April*, 80(26), 2012.

[70] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European Conference on Computer Vision*, pages 646–661. Springer, 2016.

[71] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] We state our theoretical and experimental contributions in §1. We develop our multi-resolution framework with corresponding Theorems 1 to 5 in §2 (proofs in Appendix A). We provide our experimental results in §3.
   (b) Did you describe the limitations of your work? [Yes] See §5.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Appendix E.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] We stated the assumptions of all theoretical results in §2 and refer to Appendices A and B for further details.
   (b) Did you include complete proofs of all theoretical results? [Yes] The complete proofs of Theorems 1 to 5, as well as of all additional experimental results are provided in Appendix A.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We provide our code, and instructions on how to download the data and reproduce the main results in the supplementary material. In particular, we refer to the `README.md` file in the code repository for further details which follow the NeurIPS Code Completeness Checklist.

    (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendices F and D.
    (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Due to the significant computational cost of training extremely deep HVAEs (multiple Nvidia A100 graphic cards with 40GB of GPU memory each running for 3 weeks per run), we did not perform the multiple runs per hyperparameter setting required to provide error bars for our runs. We note that this is common practice in large-scale HVAE research (compare [9, 10]). Furthermore, in their code base, VDVAE [9], which we directly base our architecture on, reported highly stable test NLL when varying the random seed, varying only in the second decimal place in bits per dim.
    (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We discuss our computational resources used and an estimate of the total amount of compute required to reproduce our results briefly in §3, and in detail in Appendix C.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
    (a) If your work uses existing assets, did you cite the creators? [Yes] We discuss all existing assets used in our code base in Appendix C.
    (b) Did you mention the license of the assets? [Yes] We mention the license of all assets used in Appendix C.
    (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We provide our source code with instructions on how to reproduce our results in the supplementary material. Further details on our code are provided in Appendix C.
    (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] We address this question in Appendix D.
    (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] We likewise address this question in Appendix D.

5. If you used crowdsourcing or conducted research with human subjects...
    (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
    (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
    (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Appendix for *A Multi-Resolution Framework for U-Nets with Applications to Hierarchical VAEs*

A Framework Details and Technical Proofs

Here we provide proofs for the theorems in the main paper and additional theoretical results supporting these.

A.1 Definitions and Notations

The following provides an index of commonly used notation throughout this manuscript for reference.

The *function space* of interest in this work is $L^2(\mathbb{X})$, the space of square integrable functions, where $\mathbb{X}$ is a compact subset of $\mathbb{R}^m$ for some integer m, for instance, $\mathbb{X} = [0, 1]$. This set of functions is defined as

$$L^2(\mathbb{X}) = \{f : \mathbb{X} \to \mathbb{R} \,|\, \|f\|_2 < \infty,\ f \text{ Borel measurable}\}. \tag{A.1}$$

$L^2(\mathbb{X})$ forms a vector space with the standard operations.

We denote $V_{-j} \subset L^2(\mathbb{X})$ as a finite-dimensional approximation space. With the nesting property, $V_{-j+1} \subset V_{-j}$, the space U_{-j+1} is the orthogonal compliment of V_{-j+1} within V_{-j}, i.e. $V_{-j} = U_{-j+1} \oplus V_{-j+1}$.

The *integration* shorthand notations used are as follows. For an integrable function $t \mapsto f(t)$, we use

$$f(t)dt := \int_0^t f(s)ds. \tag{A.2}$$

The function f may be multi-dimensional in which case we mean the multi-dimensional integral in whichever basis is being used. For *stochastic* integrals, we only analyse dynamics within the truncation V_{-J} of $L^2(\mathbb{X})$. In this case, W_t refers to a Brownian motion on the same amount of dimensions as V_{-J} in the *standard*, or 'pixel', basis of V_{-J}. The shorthand

$$g(W_t)dW_t := \int_0^t g(W_s)dW_s, \tag{A.3}$$

is used for the standard Itô integral. Last, for a stochastic process X_t on V_{-J} we use

$$dX_t := X_t - X_0, \tag{A.4}$$

if not specified otherwise.

For *measures*, we use $\mathbb{D}$ to prefix a set for which we consider the space of probability measures over: for instance, $\mathbb{D}(\mathbb{X})$ denotes the space of probability measures over $\mathbb{X}$. We often refer to measures over functions (i.e. images): recall that V_{-J} is an L^2-function space and we take $\mathbb{D}(V_{-J})$ to be probability measures over this space.

When referenced in Definition 2, the distance metric between two measures ν_1 and ν_2 which yields the topology of *weak continuity* is the *Monge-Kantorovich* metric [50, 51]

$$d_{\mathbb{P}}(\nu_1, \nu_2) = \sup_{f \in \mathrm{Lip}_1(\mathbb{X})} \int f d(\nu_1 - \nu_2), \tag{A.5}$$

where

$$\mathrm{Lip}_1(\mathbb{X}) = \{f : \mathbb{X} \to \mathbb{R} \big| \ |f(x) - f(y)| \leq d(x, y), \forall x, y \in \mathbb{X}\}. \tag{A.6}$$

Further, we use the Wasserstein-2 metric which in comparison to the weak convergence above has additional moment assumptions. It is given by

$$\mathcal{W}_2(\nu_1, \nu_2) = \left(\inf_{\gamma \in \Gamma(\nu_1, \nu_2)} \mathbb{E}\|X_1 - X_2\|_2^2 \right)^{1/2}. \tag{A.7}$$

where $(X_1, X_2) \sim \gamma$ and $\Gamma(\nu_1, \nu_2)$ is the space of measures on $\mathbb{D}(V_{-J} \times V_{-J})$ with marginals ν_1 and ν_2.

A.2 Dimension Reduction Conjugacy

Assume momentarily the one dimensional case where $\mathbb{X} = [0, 1]$. Let V_{-j} be an *approximation space* contained in $L^2(\mathbb{X})$ (see Definition 1) pertaining to image pixel values

$$V_{-j} = \{f \in L^2([0, 1]) \mid f_{[2^{-j} \cdot k, 2^{-j} \cdot (k+1))} = c_k,\ k \in \{0, \dots, 2^j - 1\},\ c_k \in \mathbb{R}\}. \tag{A.8}$$

For a function $f \in V_{-j}$, there are several ways to express f in different bases. Consider the *standard (or 'pixel') basis* for a fixed V_{-j} given via

$$e_{j,k} = \mathbb{1}_{[2^{-j} \cdot k, 2^{-j} \cdot (k+1)]}. \tag{A.9}$$

Clearly, the family $\mathbf{E}_j := \{e_{j,k}\}_{k=0}^{2^j-1}$ is an orthogonal basis of V_{-j}, hence full rank with dimension 2^j. Functions in V_{-j} may be expressed as

$$f = \sum_{k=0}^{2^j-1} c_k \cdot e_{j,k}, \tag{A.10}$$

for $c_k \in \mathbb{R}$.

First, let us recall the average *pooling* operation in these bases $\mathbf{E}_j$ and $\mathbf{E}_{j-1}$ of V_{-j} and V_{-j+1}, where $\text{pool}_{-j,-j+1} : V_{-j} \to V_{-j+1}$. Its operation is given by

$$\text{pool}_{-j,-j+1}(f) = \text{pool}_{-j,-j+1}\left(\sum_{k=0}^{2^j-1} c_k \cdot e_{j,k}\right) = \sum_{i=0}^{2^{j-1}-1} \tilde{c}_i \cdot e_{j-1,i}, \tag{A.11}$$

where for $i \in \{0, \dots, 2^{j-1} - 1\}$ we have the coefficient relation

$$\tilde{c}_i = \frac{c_{2i} + c_{2i+1}}{2} = \frac{1}{2^{-j}} \int_{[2^{-j} \cdot (2i), 2^{-j} \cdot (2i+1))} f(x) dx. \tag{A.12}$$

Average pooling and its imposed basis representation are commonly used in U-Net architectures [1], for instance in state-of-the-art diffusion models [13] and HVAEs [9].

Note that across approximation spaces of two resolutions V_{-j} and V_{-j+1}, the standard bases $\mathbf{E}_j$ and $\mathbf{E}_{j-1}$ share no basis elements. As basis elements change at each resolution, it is difficult to analyse V_{-j} embedded in V_{-j+1}. What we seek is a basis for all V_{-j} such that any basis element in this set at resolution j is also a basis element in V_{-J}, the approximation space of highest resolution J we consider. This is where wavelets serve their purpose: We consider a *multi-resolution (or 'wavelet') basis* of V_{-J} [52]. For the purpose of our theoretical results below, we are here focusing on a *Haar wavelet* basis [32] which we introduce in the following, but note that our framework straight-forwardly generalises to other wavelet bases. Begin with $\phi_1 = \mathbb{1}_{[0,1)}$ as L^2-basis element for V_{-1}, the space of constant functions on $[0, 1)$. For V_{-2} we have the space of L^2 functions which are constant on $[0, 1/2)$ and $[1/2, 1)$, which we receive by adding the basis element $\psi = \sqrt{2}(\mathbb{1}_{[0,1/2)} - \mathbb{1}_{[1/2,1)})$. Here ϕ_1 is known as the father wavelet, and ψ as the mother wavelet. To make a basis for general V_{-j} we localise these two wavelets with scaling and translation, i.e

$$\psi_{i,k} = 2^{-i/2} \cdot \psi(2^i(\cdot - k)) \quad \text{where } i \in \{0, j\}, k \in \{0, 2^{-i+1}\}. \tag{A.13}$$

It is straight-forward to check that $\mathbf{\Psi}_j := \{\psi_{i,k}\}_{i=0,k=0}^{j,2^{i-1}}$ is an orthonormal basis of V_{-j} on $[0, 1]$. Further, the truncated basis $\mathbf{\Psi}_{j-1}$, which is a basis for V_{-j+1}, is contained in the basis $\mathbf{\Psi}_j$. This is in contrast to $\mathbf{E}_{j-1}$ which has basis elements distinct from the elements in the basis $\mathbf{E}_j$ on a higher resolution.

The collections $\mathbf{E}_j$ and $\mathbf{\Psi}_j$ both constitute full-rank bases for V_{-j}. They further have the same dimension and so there is a linear isomorphism $\pi_j : V_{-j} \to V_{-j}$ for change of basis, i.e.

$$\pi_j(e_{j,i}) = \psi_{j,i}. \tag{A.14}$$

This can be normalised to be an isometry. We now analyse the pooling operation in our basis $\mathbf{\Psi}_j$, restating Theorem 2 from the main text and providing a proof.

Theorem 2. Given V_{-j} as in Definition 1, let $x \in V_{-j}$ be represented in the standard basis $\mathbf{E}_j$ and Haar basis $\mathbf{\Psi}_j$. Let $\pi_j : \mathbf{E}_j \mapsto \mathbf{\Psi}_j$ be the change of basis map illustrated in Fig. 3, then we have the conjugacy $\pi_{j-1} \circ \text{pool}_{-j,-j+1} = \text{proj}_{V_{-j+1}} \circ \pi_j$.

Proof. Define the conjugate pooling map in the wavelet basis, $\text{pool}^*_{j,j+1} : V_{-j} \to V_{-j+1}$ computed on the bases $\mathbf{\Psi}_j$ and $\mathbf{\Psi}_{j-1}$,

$$\text{pool}^*_{-j,-j+1} \coloneqq \pi_{j-1} \circ \text{pool}_{-j,-j+1} \circ \pi_j^{-1}. \tag{A.15}$$

$$\begin{tikzcd}
(V_{-j}, \mathbf{E}_j) \arrow[r, "\text{pool}_{-j,-j+1}"] & (V_{-j+1}, \mathbf{E}_{j-1}) \arrow[d, "\pi_{j-1}"'] \\
(V_{-j}, \mathbf{\Psi}_j) \arrow[u, "\pi_j^{-1}"'] \arrow[r, dashed, "\text{pool}^*_{-j,-j+1}"'] & (V_{-j+1}, \mathbf{\Psi}_{j-1})
\end{tikzcd}$$

Due to the scaling and translation construction in Eq. (A.13) and because the pooling operation is local, we need only consider the case for $\text{pool}_{-2,-1}$. This is because one can view pooling between the higher-resolution spaces as multiple localised pooling operations between V_{-2} and V_{-1}. Now note that $\text{pool}_{-2,-1}$ maps V_{-2} to V_{-1}. Further,

$$\int_{\mathbb{X}} \psi(x) dx = 0, \tag{A.16}$$

where $\psi = \sqrt{2}(\mathbb{1}_{[0,1/2)} - \mathbb{1}_{[1/2,1)})$ is the mother wavelet. For $v \in V_{-2}$ let v have the wavelet representation $v = \tilde{c}_2\psi + \tilde{c}_1\phi_1$, where $\phi_1 = \mathbb{1}_{[0,1)}$ is the father wavelet. To pool we compute the average of the two coefficients ('pixel values')

$$\text{pool}_{-2,-1}(v) = \int_{\mathbb{X}} v(x)dx = \int_{\mathbb{X}} \tilde{c}_2\psi(x) + \tilde{c}_1\phi_1(x)dx = \tilde{c}_1. \tag{A.17}$$

Thus average pooling here corresponds to truncation of the wavelet basis for V_{-2} to the wavelet basis for V_{-1}. As this basis is orthonormal over $L^2(\mathbb{X})$, truncation corresponds to L^2 projection, i.e. $\text{pool}^*_{-j,-j+1} = \text{proj}_{V_{-j+1}}$, as claimed. ☐

Theorem 2 shows that the pooling operation is conjugate to projection in the Haar wavelet approximation space, and computed by truncation in the Haar wavelet basis. The only quantity we needed for our basis over the V_{-j} was the vanishing moment quantity

$$\int_{\mathbb{X}} \psi(x) dx = 0. \tag{A.18}$$

To extend this property to higher dimensions, such as the two dimensions of gray-scale images, we use the tensor product of $[0, 1]$, and further, the tensor product of basis functions. This property is preserved, and hence the associated average pooling operation is preserved on the tensor product wavelet basis, too. To further extend it to color images, one may consider the cartesian product of several L2 spaces.

A.3 Average pooling Truncation Error

In this section we prove Theorem 3, which quantifies the regularisation imposed by an average pooling bottleneck trained by minimising the reconstruction error. The proof structure is as follows: First we give an intuition for autoencoders with an average pooling bottleneck, then derive the relevant assumptions for Theorem 3. We next prove our result under strong assumptions. Last, we weaken our assumptions so that our theorem is relevant to HVAE architectures.

Suppose we train an autoencoder on V_{-j} without dimension reduction, calling the parameterised forward (or encoder/bottom-up) and backward (or decoder/top-down) passes $F_{j,\phi}$, $B_{j,\theta} : V_{-j} \mapsto V_{-j}$ respectively. We can optimise $F_{j,\phi}$ and $B_{j,\theta}$ w.r.t. ϕ and θ to find a perfect reconstruction, i.e. $x = B_{j,\theta}F_{j,\phi}x$ for all x in our data as there is no bottleneck (no dimensionality reduction): $B_{j,\theta}$ need only be a left inverse of $F_{j,\phi}$, as in

$$B_{j,\theta}F_{j,\phi} = I. \tag{A.19}$$

Importantly, we can choose $F_{j,\phi}$ and $B_{j,\theta}$ satisfying A.19 *independent* of our data. For instance, they could both be the identity operator and achieve perfect reconstruction, but contain no information about the generative characteristics of our data. Compare this to a *bottleneck* with average pooling, i.e. an autoencoder with dimension reduction. Here, we consider the dimension reduction from V_{-j}

to V_{-j+1}, where we split $V_{-j} = V_{-j+1} \oplus U_{-j+1}$. As we have seen in Theorem 2, through average pooling, we keep information in V_{-j+1}, and discard the information in U_{-j+1}. For simplicity, let $\text{embd}_{V_{-j}}$ be the inclusion of the projection $\text{proj}_{V_{-j+1}}$. Now to achieve perfect reconstruction

$$x = (B_{j,\theta} \circ \text{embd}_{V_{-j}} \circ \text{proj}_{V_{-j+1}} \circ F_{j,\phi})x, \tag{A.20}$$

we require $(\text{proj}_{U_{-j+1}} F_{j,\phi})x = 0$. Simply put, the encoder $F_{j,\phi}$ should make sure that the discarded information in the bottleneck is nullified.

We may marry this observation with a simple U-Net structure (without skip connection) with L^2-reconstruction and average pooling dimension reduction. Let V_{-j} be one of our multi-resolution approximation spaces and $\mathbb{D}(V_{-j})$ be the space of probability measures over V_{-j}. Recall in a multi-resolution basis we have $V_{-j} = V_{-j+1} \oplus U_{-j+1}$ where U_{-j+1} is the $-j+1$ orthogonal compliment within V_{-j}. For any $v \in V_{-j}$ we may write $v = \text{proj}_{V_{-j+1}} v \oplus \text{proj}_{U_{-j+1}} v$ and analyse the truncation error in V_{-j+1}, i.e. the discarded information, via

$$\|v - \text{embd}_{V_{-j}} \circ \text{proj}_{V_{-j+1}} v\|_2^2 = \|\text{proj}_{U_{-j+1}} v\|_2^2. \tag{A.21}$$

If we normalise this value to

$$\frac{\|v - \text{embd}_{V_{-j}} \circ \text{proj}_{V_{-j+1}} v\|_2^2}{\|v\|_2^2} = \frac{\|\text{proj}_{U_{-j+1}} v\|_2^2}{\|v\|_2^2} \in [0,1], \tag{A.22}$$

then this is zero when v is non-zero only within V_{-j+1} and zero everywhere within U_{-j+1}. Suppose now that we have a measure $\nu_j \in \mathbb{D}(V_{-j})$, we could quantify *how much* of the norm of a sample from ν_j comes from the U_{-j+1} components by computing

$$\mathbb{E}_{v\sim\nu_j} \frac{\|\text{proj}_{U_{-j+1}} v\|_2^2}{\|v\|_2^2} = \int \frac{\|\text{proj}_{U_{-j+1}} v\|_2^2}{\|v\|_2^2} d\nu_j(v) \in [0,1]. \tag{A.23}$$

This value forms a convex sum with its complement projection to $\text{proj}_{V_{-j+1}}$, demonstrating the splitting of mass across V_{-j+1} and U_{-j+1}, as we show in Lemma 1.

Lemma 1. Let $\nu_j \in \mathbb{D}(V_{-j})$ be atom-less at 0, then

$$\mathbb{E}_{v\sim\nu_j} \frac{\|\text{proj}_{V_{-j+1}} v\|_2^2}{\|v\|_2^2} + \mathbb{E}_{v\sim\nu_j} \frac{\|\text{proj}_{U_{-j+1}} v\|_2^2}{\|v\|_2^2} = 1. \tag{A.24}$$

Proof. For any $v \in V_{-j}$ we have $\|v\|_2^2 = \|\text{proj}_{V_{-j+1}} v\|_2^2 + \|\text{proj}_{U_{-j+1}} v\|_2^2$ due to orthogonality of V_{-j+1} and U_{-j+1}. As both $\|\text{proj}_{V_{-j+1}} v\|_2^2$ and $\|\text{proj}_{U_{-j+1}} v\|_2^2$ are projections, they are bounded by $\|v\|_2^2$ giving that the integrands in Eq. (A.24) are bounded by one, and so for all $v \neq 0$ (no point mass at 0) the expectation is bounded. ☐

From the splitting behaviour of masses in the L^2-norm observed in Lemma 1 we see that

1. if $\mathbb{E}_{v\sim\nu_j} \|\text{proj}_{U_{-j+1}} v\|_2^2 / \|v\|_2^2$ is large, then, on average, samples from ν_j have most of their size in the U_{-j+1} subspace; or,
2. if $\mathbb{E}_{v\sim\nu_j} \|\text{proj}_{U_{-j+1}} v\|_2^2 / \|v\|_2^2$ is small, then, on average, samples from ν_j have most of their size in the V_{-j+1} subspace.

In the latter case, $\|\text{proj}_{U_{-j+1}} v\|_2^2 \approx 0$, i.e. $\text{embd}_{V_{-j}} \circ \text{proj}_{V_{-j+1}} v \approx v$. We get the heuristic $\text{embd}_{V_{-j}} \circ \text{proj}_{V_{-j+1}} \approx I$ on the measure ν_j, yielding a perfect reconstruction.

Let $\text{embd}_{V_{-j}} \circ \text{proj}_{V_{-j+1}}, I : V_{-j} \to V_{-j}$, then this heuristic performs the operator approximation

$$\mathbb{E}_{v\sim\nu_j} \|(\text{embd}_{V_{-j}} \circ \text{proj}_{V_{-j+1}} - I)v\|_2^2, \tag{A.25}$$

quantifying 'how close' these operators are on ν_j. For many measures, this (near) equivalence between operators will not hold. But what if instead, we had an operator $D : V_{-j} \to V_{-j}$ such that the push-forward of ν_j through this operator had this quality. Practically, this push-forward operator will be parameterised by neural networks, for instance later in the context of U-Nets. For simplicity, we will initially consider the case where D is linear on V_{-j}, then we consider when D is Lipschitz.

Lemma 2. Given V_{-j} with the L^2-orthogonal decomposition $V_{-j} = V_{-j+1} \oplus U_{-j+1}$, let $D_{-j} : V_{-j} \to V_{-j}$ be an invertible linear operator and define $F_j : V_{-j} \to V_{-j+1}$ and $B_j : V_{-j+1} \to V_{-j}$ through

$$F_j = \mathrm{proj}_{V_{-j+1}} \circ D_j, \qquad\qquad B_j = D_j^{-1} \circ \mathrm{embd}_{V_{-j}}. \tag{A.26}$$

Then $B_j F_j \equiv I$ on V_{-j}, or otherwise, we have the truncation bound

$$\frac{\left\| \mathrm{proj}_{U_{-j+1}} F_j v \right\|_2^2}{\|D_j\|_2^2} \leq \frac{\left\| \mathrm{proj}_{U_{-j+1}} F_j v \right\|_2^2}{\|F_j\|_2^2} \leq \|(I - B_j F_j)v\|_2^2 . \tag{A.27}$$

Proof. Consider the operator $D_j(I - B_j F_j) : V_{-j} \to V_{-j}$ which is linear and obeys the multiplicative bound $\|D_j(I - B_j F_j)\| \leq \|D_j\| \, \|I - B_j F_j\|$. This implies for any $v \in V_{-j}$,

$$\frac{\|D_j(I - B_j F_j)v\|_2^2}{\|D_j\|_2^2} \leq \|(I - B_j F_j)v\|_2^2 . \tag{A.28}$$

The numerator is equal to

$$\|D_j(I - B_j F_j)v\|_2^2 = \left\| (D_j - \mathrm{embd}_{V_{-j}} \circ \mathrm{proj}_{V_{-j+1}} \circ D_j)v \right\|_2^2 . \tag{A.29}$$

As we have the orthogonal decomposition $V_{-j} = V_{-j+1} \oplus U_{-j+1}$, we know

$$\begin{aligned} I &= \mathrm{proj}_{V_{-j+1}} \oplus \mathrm{proj}_{U_{-j+1}} && \text{(A.30)} \\ &= \mathrm{embd}_{V_{-j}} \circ \mathrm{proj}_{V_{-j+1}} + \mathrm{embd}_{V_{-j}} \circ \mathrm{proj}_{U_{-j+1}}, && \text{(A.31)} \end{aligned}$$

and as $\left\| \mathrm{embd}_{V_{-j}} \right\|_2 = 1$, we get

$$\left\| (I - \mathrm{embd}_{V_{-j}} \circ \mathrm{proj}_{V_{-j+1}} \circ D_j)v \right\|_2^2 = \left\| \mathrm{proj}_{U_{-j+1}} \circ D_j v \right\|_2^2 . \tag{A.32}$$

So now as $\|D_j(I - B_j F_j)v\|_2^2 = \left\| \mathrm{proj}_{U_{-j+1}} \circ D_j v \right\|_2^2$, we may use $\|D_j(I - B_j F_j)v\|_2^2 \leq \|D_j\|^2 \, \|I - B_j F_j v\|_2^2$ to get the desired result. ☐

The quantity $\left\| \mathrm{proj}_{U_{-j+1}} F_j v \right\|_2^2 / \|F_j\|_2^2$ is analogous to the in Lemma 1 discussed quantity $\|\mathrm{proj}_{U_{-j+1}} v\|_2^2 / \|v\|_2^2$, but we now have a 'free parameter', the operator D_j.

Next, suppose D_j is trainable with parameters θ. We do so by minimising the reconstruction cost

$$\mathbb{E}_{v \sim \nu_j} \|(I - B_j F_j)v\|_2^2 , \tag{A.33}$$

which upper-bounds our 'closeness metric' in Lemma 2.

In the linear case (D_j is linear), to ensure that $D_{j,\theta}$ is invertible we may parameterise it by an (unnormalised) LU-decomposition of the identity

$$I = D_{j,\theta}^{-1} D_{j,\theta} = L_{j,\theta} U_{j,\theta}, \tag{A.34}$$

where the diagonal entries of $L_{j,\theta}$ and $U_{j,\theta}$ are necessarily inverses of one-another. This is a natural parameterisation when considering a U-Net with dimensionality reduction. Building from Lemma 2, we can now consider the stacked U-Net (without skip connections), i.e. a U-Net with multiple downsampling/upsampling and forward/backward operators stacked on top of each other, in the linear setting. In Proposition 1, we show that this LU-parameterisation forces the pivots of $U_{j,\theta}$ to tend toward zero.

Proposition 1. Let $\{V_{-j}\}_{j=0}^{J}$ be a multi-resolution hierarchy of V_{-J} with the orthogonal decompositions $V_{-j} = V_{-j+1} \oplus U_{-j+1}$ and $F_{j,\phi}$, $B_{j,\theta} : V_{-j} \to V_{-j}$ be bounded linear operators such that $B_{j,\theta} F_{j,\phi} = I$. Define $\boldsymbol{F}_{j,\phi} : V_{-j} \to V_{-j+1}$ and $\boldsymbol{B}_{j,\theta} : V_{-j+1} \to V_{-j}$ by

$$\boldsymbol{F}_{j,\phi} := \mathrm{proj}_{V_{-j+1}} \circ F_{j,\phi}, \qquad\qquad \boldsymbol{B}_{j,\theta} := B_{j,\theta} \circ \mathrm{embd}_{V_{-j}}, \tag{A.35}$$

with compositions

$$\boldsymbol{F}_{j_1|j_2,\phi} \coloneqq \boldsymbol{F}_{j_1,\phi} \circ \cdots \circ \boldsymbol{F}_{j_2,\phi}, \qquad \boldsymbol{B}_{j_1|j_2,\phi} \coloneqq \boldsymbol{B}_{j_1,\phi} \circ \cdots \circ \boldsymbol{B}_{j_2,\phi}. \tag{A.36}$$

Then

$$\sum_{j=1}^{J} \frac{\left\| \text{proj}_{U_{-j+1}} F_j v \right\|_2^2}{\|F_j\|_2^2} \leq \left\| (I - \boldsymbol{B}_{1|J,\theta} \boldsymbol{F}_{1|J,\phi}) v \right\|_2^2. \tag{A.37}$$

Proof. The operator $\mathbf{F}_{1|J}$ is linear, and decomposes into a block operator form with pivots $\mathbf{F}_{j|J}$ for each $j \in \{1, \ldots, J\}$. Each $\mathbf{F}_{j|J}$ is L^2-operator norm bounded by $\|F_j\|_2$, so if

$$\lambda_{1|J} \coloneqq \text{diag}(\|F_1\|_2, \ldots, \|F_J\|_2), \tag{A.38}$$

then $\|\lambda_{1|J}^{-1} \mathbf{F}_{1|J}\|_2 \leq 1$. Last, as the spaces $\{U_{-j}\}_{j=0}^{J}$ are orthogonal and $\mathbf{F}_{1|J}$ has triangular form:

$$\|\lambda_{1|J}^{-1} (F_{1|J} - \mathbf{F}_{1|J}) v\|_2^2 = \sum_{j=1}^{J} \frac{\left\| \text{proj}_{U_{-j+1}} F_j v \right\|_2^2}{\|F_j\|_2^2}, \tag{A.39}$$

and $\|\lambda_{1|J}^{-1} (F_{1|J} - \mathbf{F}_{1|J}) v\|_2^2 \leq \|(I - \mathbf{B}_{1|J} \mathbf{F}_{1|J}) v\|_2^2$. $\square$

Here in the linear case, a U-Net's encoder is a triangular matrix where the basis vectors are the Haar wavelets. Proposition 1 states that the pivots of this matrix are minimised. Adversely, this diminishes the rank of the autoencoder and pushes our original underdetermined problem to a singular one. In other words, the U-Net is in this case demanding to approximate the identity (via an LU-like-decomposition), a linear operator, with an operator of diminishing rank.

Proposition 2. Let $\mathbb{D}(\mathbb{X})$ be the space of probability measures over $\mathbb{X}$, and assume for $\overline{F}_j, \overline{B}_j : \mathbb{D}(\mathbb{X}) \to \mathbb{D}(\mathbb{X})$ that these are inverses of one-another and $\overline{F}_j$ is Lipschitz, that is

$$\overline{F}_j \overline{B}_j = I, \qquad \mathcal{W}_2(\overline{F}_j \nu_1, \overline{F}_j \nu_2) \leq \|\overline{F}_j\|_2 W_2(\nu_1, \nu_2). \tag{A.40}$$

Then for any $\nu \in \mathbb{D}(\mathbb{X})$ with bounded second moment,

$$\mathbb{E}_{X_j \sim \overline{F}_j \nu} \frac{\|\text{proj}_{U_{-j}} X_j\|_2^2}{\|\overline{F}_j\|_2^2} \leq \mathcal{W}_2(\nu, \overline{B}_j \circ P_{V_{-j}} \circ \overline{F}_j \nu). \tag{A.41}$$

Proof. First as $\overline{F}_j \overline{B}_j = I$ we know that

$$\mathcal{W}_2(\overline{F}_j \nu, P_{V_{-j}} \overline{F}_j \nu) = \mathcal{W}_2(\overline{F}_j \overline{B}_j \overline{F}_j \nu, \overline{F}_j \overline{B}_j P_{V_{-j}} \overline{F}_j \nu). \tag{A.42}$$

But for any $X \in V_{-j}$ we have the orthogonal decomposition

$$X = \text{proj}_{V_{-j}} X \oplus \text{proj}_{U_{-j}} X, \tag{A.43}$$

which respects the L^2-norm by

$$\|X\|_2^2 = \|\text{proj}_{V_{-j}} X\|_2^2 + \|\text{proj}_{U_{-j}} X\|^2, \tag{A.44}$$

and in particular,

$$\|X - \text{proj}_{V_{-j}} X\|_2^2 = \|\text{proj}_{U_{-j}} X\|^2. \tag{A.45}$$

This grants

$$\begin{aligned}
(W_2(\overline{F}_j \nu, P_{V_{-j}} \overline{F}_j \nu))^2 &= \inf_{\gamma \in \Gamma(\overline{F}_j \nu, P_{V_{-j}} \overline{F}_j \nu)} \mathbb{E} \|X - Y\|_2^2 && \text{(A.46)} \\
&= \inf_{\gamma \in \Gamma(\overline{F}_j \nu, P_{V_{-j}} \overline{F}_j \nu)} \mathbb{E} \|\text{proj}_{V_{-j}} X - \text{proj}_{V_{-j}} Y\|_2^2 + \|\text{proj}_{U_{-j}} X\|_2^2 && \text{(A.47)} \\
&= \inf_{\gamma \in \Gamma(\overline{F}_j \nu, P_{V_{-j}} \overline{F}_j \nu)} \mathbb{E} \|\text{proj}_{U_{-j}} X\|_2^2 && \text{(A.48)} \\
&= (\mathcal{W}_2(\text{proj}_{U_{-j}} \overline{F}_j \nu, \delta_{\{0\}}))^2. && \text{(A.49)}
\end{aligned}$$

Now the Lipschitz of $\overline{F}_j$ yields

$$\frac{\mathcal{W}_2(\overline{F}_j\overline{B}_j\overline{F}_j\nu, \overline{F}_j\overline{B}_jP_{V_{-j}}\overline{F}_j\nu)}{\|\overline{F}_j\|_2} \leq \mathcal{W}_2(\overline{B}_j\overline{F}_j\nu, \overline{B}_jP_{V_{-j}}\overline{F}_j\nu) = \mathcal{W}_2(\nu, \overline{B}_jP_{V_{-j}}\overline{F}_j\nu). \quad \text{(A.50)}$$

Squaring and substituting grants

$$\frac{(\mathcal{W}_2(\text{proj}_{U_{-j}}\overline{F}_j\nu, \delta_{\{0\}}))^2}{\|\overline{F}_j\|_2^2} \leq \mathcal{W}_2(\nu, \overline{B}_jP_{V_{-j}}\overline{F}_j\nu). \quad \text{(A.51)}$$

□

To work on multiple resolution spaces, we need to define what the triangular operator over our space of measures is. For a cylinder set B on $V_{-J} = V_0 \oplus \bigoplus_{j=0}^{J} U_{-j}$ we can assume it has the form $\bigotimes_j B_j$ where B_j is a cylinder on U_j. Break ν_J into the multi-resolution sub-spaces by defining projection onto $\mathbb{D}(U_{-j})$ through

$$\text{proj}_{U_{-j}}(\nu_J)(B_j) := \nu_J(B_j \otimes U_{-j}^{\perp}), \quad \text{(A.52)}$$

where B_j is a cylinder for U_{-j}. This projection of measures is respected by evaluation in that

$$\mathbb{E}_{X_j\sim\text{proj}_{U_{-j}}\nu_J}X_j = \int v_j d\text{proj}_{U_{-j}}\nu_J(v_j) = \int \text{proj}_{U_{-j}}v d\nu_J(v) = \mathbb{E}_{X_j\sim\nu_J}\text{proj}_{U_{-j}}X. \quad \text{(A.53)}$$

As $\|X\|_2^2 = \sum_j \text{proj}_{U_{-j}}\|\text{proj}_{U_{-j}}X\|_2^2$ due to the orthogonality of the spaces, then

$$\mathbb{E}_{X_j\sim\text{proj}_{U_{-j}}\nu_J}\|X_j\|_2^2 = \sum_j \mathbb{E}_{X_j\sim\text{proj}_{U_{-j}}\nu_J}\|X_j\|_2^2 = \sum_j \mathbb{E}_{X\sim\nu_J}\|\text{proj}_{U_{-j}}X\|_2^2. \quad \text{(A.54)}$$

Define the extension, with a convenient abuse of notation, of $\text{proj}_{V_{-j+1}}$ on $\mathbb{D}(V_{-J})$ to be

$$\text{proj}_{V_{-j+1}}(\nu_J) := \text{proj}_{V_{-j+1}}(\nu_J) \otimes \text{proj}_{V_{-j+1}^{\perp}}(\nu_J). \quad \text{(A.55)}$$

If $F_{-j} : \mathbb{D}(V_{-j}) \to \mathbb{D}(V_{-j})$ are linear operators for $j \in \{0, \dots, J\}$, extend each $F_j : \mathbb{D}(V_{-j}) \to \mathbb{D}(V_{-j})$ to $\mathbb{D}(V_{-j}) \times \mathbb{D}(V_{-j}^{\perp})$ through

$$\overline{F}_j := F_j \oplus I. \quad \text{(A.56)}$$

For a measure $\nu_J \in \mathbb{D}(V_{-J})$ we can split it into $\mathbb{D}(V_{-j}) \times \mathbb{D}(V_{-j}^{\perp})$ via

$$\text{proj}_{V_{-j}}\nu_J \times \text{proj}_{V_{-j}^{\perp}}\nu_J, \quad \text{(A.57)}$$

which also remains a measure in $\mathbb{D}(V_{-J})$ as $\mathbb{D}(V_{-j}) \times \mathbb{D}(V_{-j}^{\perp}) \subset \mathbb{D}(V_{-J})$. Now the operator $\overline{F}_j$ acts on the product measure $\nu_j \otimes \nu_j^{\perp}$ by

$$\overline{F}_j(\nu_j \otimes \nu_j^{\perp}) = F_j\,\nu_j \otimes I\,\nu_j^{\perp}. \quad \text{(A.58)}$$

Now we may define the map $\boldsymbol{F}_j : \mathbb{D}(V_{-J}) \to \mathbb{D}(V_{-j}) \times \mathbb{D}(V_{-j}^{\perp})$ through

$$\boldsymbol{F}_j := \overline{F}_j\text{proj}_{V_{-j}}, \quad \text{(A.59)}$$

and its compositions by

$$\boldsymbol{F}_{j_1|j_2} = \boldsymbol{F}_{j_1} \circ \cdots \circ \boldsymbol{F}_{j_2}, \quad \text{(A.60)}$$

which too is an operator on $\mathbb{D}(V_{-J})$.

Further if we have a measure ν_j on V_{-j} we can form the embedding map

$$\text{embd}_j\nu_j = \nu_j \otimes \bigotimes_{i=j}^{J} \delta_{\{0\}}, \quad \text{(A.61)}$$

which we extend to $\mathbb{D}(V_{-J})$ by a convenient abuse of notation

$$\mathrm{proj}_j \nu_J = \mathrm{proj}_j(\nu_J) \otimes \bigotimes_{i=j}^{J} \delta_{\{0\}}. \tag{A.62}$$

Let $B_{-j} : \mathbb{D}(V_{-j}) \to \mathbb{D}(V_{-j})$ be the linear operator which is the inverse of F_{-j}. Now if we extend $B_j : \mathbb{D}(V_{-j}) \to \mathbb{D}(V_{-j})$ to $\mathbb{D}(V_{-j}) \times \mathbb{D}(V_{-j}^{\perp})$ like before through

$$\overline{B}_j \coloneqq B_j \oplus I, \tag{A.63}$$

so the map $\overline{B}_j \mathrm{embd}_j$ is well defined on $\mathbb{D}(V_{-J})$. Now analogously define $\boldsymbol{B}_j$ and its compositions by

$$\boldsymbol{B}_j \coloneqq \overline{B}_j \mathrm{embd}_{V_{-j}} \qquad\qquad \boldsymbol{B}_{j_1|j_2} = \boldsymbol{B}_{j_2} \circ \cdots \circ \boldsymbol{B}_{j_1}. \tag{A.64}$$

In an analogous way, the operator $\boldsymbol{F}_{j_1|j_2}$ is 'upper triangular' and $\boldsymbol{B}_{j_1|j_2}$ is 'lower triangular'. In this way, we are again seeking a lower/upper (LU-) decomposition of the identity on $\mathbb{D}(V_{-J})$. Now we may prove Theorem 3.

Theorem 3. Let $\{V_{-j}\}_{j=0}^{J}$ be a multi-resolution hierarchy of V_{-J} where $V_{-j} = V_{-j+1} \oplus U_{-j+1}$, and further, let $F_{j,\phi}, B_{j,\theta} : \mathbb{D}(V_{-j}) \mapsto \mathbb{D}(V_{-j})$ be such that $B_{j,\theta}F_{j,\phi} = I$ with parameters ϕ and θ. Define $\boldsymbol{F}_{j_1|j_2,\phi} \coloneqq \boldsymbol{F}_{j_1,\phi} \circ \cdots \circ \boldsymbol{F}_{j_2,\phi}$ by $\boldsymbol{F}_{j,\phi} : \mathbb{D}(V_{-j}) \mapsto \mathbb{D}(V_{-j+1})$ where $\boldsymbol{F}_{j,\phi} \coloneqq \mathrm{proj}_{V_{-j+1}} \circ F_{j,\phi}$, and analogously define $\boldsymbol{B}_{j_1|j_2,\theta}$ with $\boldsymbol{B}_{j,\theta} \coloneqq B_{j,\theta} \circ \mathrm{embd}_{V_{-j}}$. Then, the sequence $\{\boldsymbol{B}_{1|j,\theta}(\boldsymbol{F}_{1|J,\phi}\nu_J)\}_{j=0}^{J}$ forms a discrete multi-resolution bridge between $\boldsymbol{F}_{1|J,\phi}\nu_J$ and $\boldsymbol{B}_{1|J,\theta}\boldsymbol{F}_{1|J,\phi}\nu_J$ at times $\{t_j\}_{j=1}^{J}$, and

$$\sum_{j=0}^{J} \mathbb{E}_{X_{t_J}\sim\nu_J} \left\| \mathit{proj}_{U_{-j+1}} X_{t_j} \right\|_2^2 / \left\| \boldsymbol{F}_{j|J,\phi} \right\|_2^2 \leq (\mathcal{W}_2(\boldsymbol{B}_{1|J,\theta}\boldsymbol{F}_{1|J,\phi}\nu_J, \nu_J))^2, \tag{A.65}$$

where $\mathcal{W}_2$ is the Wasserstein-2 metric and $\left\|\boldsymbol{F}_{j|J,\phi}\right\|_2$ is the Lipschitz constant of $\boldsymbol{F}_{j|J,\phi}$.

Proof. All we must show is that successively chaining the projections from Proposition 2 decomposes like in Proposition 1. For $X_1, X_2 \sim \nu$, $\mathcal{W}_2(F_jF_{j+1}\nu, P_{-j+2}F_{j-1}P_{-j+1}F_j\nu)$ consider f_j, f_{j-1} as realised paths for our kernel and write $\|f_{j-1}f_jX_1 - \mathrm{proj}_{V_{-j+2}}f_{j-1}\mathrm{proj}_{V_{-j+1}}f_jX_2\|_2^2$

$$\begin{aligned} &= \|\mathrm{proj}_{V_{-j+1}}(f_jf_{j+1}X_1 - \mathrm{proj}_{V_{-j+2}}f_{j-1}\mathrm{proj}_{V_{-j+1}}f_jX_2)\|_2^2 \\ &\quad + \|\mathrm{proj}_{U_{-j+1}}(f_jf_{j+1}X_1 - \mathrm{proj}_{V_{-j+2}}f_{j-1}\mathrm{proj}_{V_{-j+1}}f_jX_2)\|_2^2 \end{aligned}$$

due to the triangular form and the orthogonality of the multi-resolution basis. Let $\nu_{-j+1} = \mathrm{proj}_{V_{-j+1}}\nu_{-j}$, then as $\mathrm{proj}_{V_{-j+1}}$ commutes with any term equivalent to the identity operator on V_{-j+1}, the first term becomes

$$\|f_{j-1}X_{1,t_{j+1}} - \mathrm{proj}_{V_{-j+2}}f_{j-1}X_{2,t_{j+1}})\|_2^2, \tag{A.66}$$

where $X_{1,t_{j+1}}, X_{2,t_{j+1}} \sim \nu_{-j+1}$. When an optimal coupling is made, this term becomes $\|\mathrm{proj}_{U_{-j+2}}X_{1,t_{j+1}}\|_2^2$. The second term has $\mathrm{proj}_{U_{-j+1}}\mathrm{proj}_{V_{-j+2}}$ nullified, and again commutes where appropriate making this

$$\|\mathrm{proj}_{U_{-j+1}}X_{1,t_{j+1}}\|_2^2. \tag{A.67}$$

We may again use the triangular form to utilise the identify

$$\|\mathrm{proj}_{U_{-j+1}}F_j\|_2^2 \leq \|F_j\|_2^2, \tag{A.68}$$

to define

$$\lambda_{j|J} \coloneqq \mathrm{diag}(\|\mathrm{proj}_{U_{-j+1}}F_{j|J}\|_2^2, \ldots, \|\mathrm{proj}_{U_{-J+1}}F_{J|J}\|_2^2) \tag{A.69}$$

so that

$$\mathcal{W}_2(\lambda_{j|J}^{-1}(F_{j|J}\nu_1), \lambda_{j|J}^{-1}(F_{j|J}\nu_2)) \leq \mathcal{W}_2(\nu_1, \nu_2). \tag{A.70}$$

Piecing the decomposition and scaling together, we yield

$$\mathbb{E}_{\nu_{-j+2}}\|\mathrm{proj}_{U_{-j+2}}X_{1,t_{j+1}}\|_2^2/\|F_{j-2|j}\|_2^2+\mathbb{E}_{\nu_{-j+1}}\|\mathrm{proj}_{U_{-j+1}}X_{1,t_{j+1}}\|_2^2/\|F_{j-2|j}\|_2^2 \tag{A.71}$$

$$\leq(\mathcal{W}_2(\nu,\mathbf{B}_{j-2|j}\mathbf{F}_{j-2|j}))^2. \tag{A.72}$$

Iterating over j in the fashion given yields the result. Last, measures within

$$\mathcal{U}_{\boldsymbol{BF}}:=\{\nu_J\mid\boldsymbol{F}_{j|J}\gamma_J=\overline{F}_{j|J}\otimes\bigotimes_{i=j}^{J}\delta_{\{0\}}\}, \tag{A.73}$$

are invariant under $\boldsymbol{B}_{J|1}\boldsymbol{F}_{1|J}$, further, $\boldsymbol{B}_{J|1}\boldsymbol{F}_{1|J}$ projects onto this set. To see this, take any measure $\nu_J\in\mathbb{D}(V_{-J})$ and apply $\boldsymbol{F}_{j|J}$. The information in $V_{-j}^{\perp}$ split by $\boldsymbol{P}_j$ is replaced by $\delta_{\{0\}}$ in the backward pass. Thus $\boldsymbol{B}_{J|1}\boldsymbol{F}_{1|J}\boldsymbol{B}_{J|1}\boldsymbol{F}_{1|J}=\boldsymbol{B}_{J|1}\boldsymbol{F}_{1|J}$. □

A.4 U-Nets in V_{-J}

Here we show how U-Nets can be seen as only computing the non-truncated components of a multi-resolution diffusion bridge on V_{-J} — the computations are performed in V_{-j} for $j<J$ at various layers. This amounts to showing the embedding presented in Theorem 1.

Theorem 1. Let $B_j:[t_j,t_{j+1})\times\mathbb{D}(V_{-j})\mapsto\mathbb{D}(V_{-j})$ be a linear operator (such as a diffusion transition kernel, see Appendix A) for $j<J$ with coefficients $\mu^{(j)},\sigma^{(j)}:[t_j,t_{j+1})\times V_{-j}\mapsto V_{-j}$, and define the natural extensions within V_{-J} in bold, i.e. $\boldsymbol{B}_j:=B_j\oplus\boldsymbol{I}_{V_{-j}^{\perp}}$. Then the operator $\boldsymbol{B}:[0,T]\times\mathbb{D}(V_{-J})\mapsto\mathbb{D}(V_{-J})$ and the coefficients $\boldsymbol{\mu},\boldsymbol{\sigma}:[0,T]\times V_{-J}\mapsto V_{-J}$ given by

$$\boldsymbol{B}:=\sum_{j=0}^{J}\mathbb{1}_{[t_j,t_{j+1})}\cdot\boldsymbol{B}_j,\quad\boldsymbol{\mu}:=\sum_{j=0}^{J}\mathbb{1}_{[t_j,t_{j+1})}\cdot\boldsymbol{\mu}^{(j)},\quad\boldsymbol{\sigma}:=\sum_{j=0}^{J}\mathbb{1}_{[t_j,t_{j+1})}\cdot\boldsymbol{\sigma}^{(j)},$$

induce a multi-resolution bridge of measures from the dynamics for $t\in[0,T]$ and on the standard basis as $dX_t=\boldsymbol{\mu}_t(X_t)dt+\boldsymbol{\sigma}_t(X_t)dW_t$ (see Appendix A.4 for the details of this integration) for $X_t\in V_{-J}$, i.e. a (backward) multi-resolution diffusion process.

Proof. At time $t=0$ we have that $\mathrm{supp}\nu_0\subset V_0=\{0\}$, so $\mathbb{D}(V_0)=\delta_{\{0\}}$. For the s in the first time interval $[t_0,t_1)$ it must be the case $\nu_s=\delta_{\{0\}}$, so $\mu_s^{(j)},\sigma_s^{(j)}=0$ and $B_0(s)\equiv I$. The extension is thus $\boldsymbol{B}_0(s)\equiv I$ on V_{-J}. At $t=t_1$, the operator $\boldsymbol{B}_1\equiv I$ on $V_1^{\perp}$ grants $\boldsymbol{\mu}_s,\boldsymbol{\sigma}_s=0$ here, On V_1, $\boldsymbol{B}_1$, it is an operator with domain in V_1, granting $\mathrm{supp}\nu_{t_1}\subset V_1$. For s within the interval (t_1,t_2) we maintain $\mathrm{supp}\nu_s\subset V_1$, and by induction we can continue this for any $s\in[t_j,t_{j+1})\subsetneq[0,1]$. Let E_j be a basis of V_{-j}, then as $\boldsymbol{\mu}_s,\boldsymbol{\sigma}_s=0$ on $V_{-j}^{\perp}$ the diffusion SDE on $[t_j,t_{j+1})\times V_{-j}$ given in the basis E_j by

$$dX_t^{(j)}=\mu_t^{(j)}(X_t)dt+\sigma_t^{(j)}(X_t)dW_t \tag{A.74}$$

embeds into an SDE on V_{-J} with basis E_J by

$$dX_t=\boldsymbol{\mu}_t(X_t)dt+\boldsymbol{\sigma}_t(X_t)dW_t, \tag{A.75}$$

which maintains $X_t\in V_{-j}$ as $\boldsymbol{\sigma}_t\equiv0$ on the complement. □

In practice, we will compute the sample paths made from Equation A.74, but we can in theory think of this as Equation A.75. The U-Net sequential truncation of spaces, then sequential inclusion of these spaces is what forms the multi-dimensional bridge with our sampling models.

A.5 Forward Euler Diffusion Approximations

Here we show that the backward cell structure of state-of-the-art HVAEs approximates an SDE evolution within each resolution.

Theorem 4. Let $t_J:=T\in(0,1)$ and consider (the p_θ backward pass) $\boldsymbol{B}_{\theta,1|J}:\mathbb{D}(V_{-J})\mapsto\mathbb{D}(V_0)$ given in multi-resolution Markov process in the standard basis:

$$dZ_t=(\overleftarrow{\mu}_{1,t}(Z_t)+\overleftarrow{\mu}_{2,t}(Z_t))dt+\overleftarrow{\sigma}_t(Z_t)dW_t, \tag{A.76}$$

where $\mathrm{proj}_{U_{-j+1}}Z_{t_j}=0,\ \|Z_t\|_2>\|Z_s\|_2$ with $0\leq s<t\leq T$ and for a measure $\nu_J\in\mathbb{D}(V_{-J})$ we have $Z_0\sim\boldsymbol{F}_{\phi,J|1}\nu_J$. Then, VDVAEs approximates this process, and its residual cells are a type of two-step forward Euler discretisation of this Stochastic Differential Equation (SDE).

Proof. The evolution

$$dZ_t = (\overleftarrow{\mu}_{1,t}(Z_t) + \overleftarrow{\mu}_{2,t}(Z_t))dt + \overleftarrow{\sigma}_t(Z_t)dW_t, \tag{A.77}$$

subject to $Z_0 = 0, \|Z_t\|_2 > \|Z_s\|_2$ and $X_T,\ Z_0 \sim \boldsymbol{F}_{\phi,J|1}\nu_J$. By Theorem 3 we know $\boldsymbol{F}_{\phi,J|1}\nu_J$ enforces the form

$$\text{proj}_{V_1}\overline{F}_{\phi,J|1}\nu_J \otimes \bigotimes_{j=1}^{J-1}\delta_{\{0\}} \tag{A.78}$$

when ϕ is trained with a reconstruction loss. By Theorem 5, the full cost used imposes $\text{proj}_{V_1}\overline{F}_{\phi,J|1}\nu_J = \delta_{\{0\}}$, further, VDVAE initialises $Z_0 = \delta_{\{0\}}$. This enforces $Z_0 = 0$ as $Z_0 \sim \delta_{\{0\}}$. For the backward SDE, consider the splitting

$$dZ_t^{(1)} = \overleftarrow{\mu}_{1,t}(Z_t^{(1)})dt + \overleftarrow{\sigma}_t(Z_t^{(1)})dW_t, \qquad dZ_t^{(2)} = \overleftarrow{\mu}_{2,t}(Z_t^{(2)})dt, \tag{A.79}$$

where $dZ_t = dZ_t^{(1)} + dZ_t^{(2)}$ when $Z_t = Z_t^{(1)} = Z_t^{(2)}$. For the split SDE make the forward-Euler discretisation

$$Z_{i+1}^{(1)} = Z_i^{(1)} + \int_i^{i+1} \overleftarrow{\mu}_{1,t}(Z_t^{(1)})dt + \int_i^{i+1} \overleftarrow{\sigma}_t(Z_i^{(1)})dW_t \approx Z_i^{(1)} + \overleftarrow{\mu}_{1,i}(Z_i^{(1)}) + \overleftarrow{\sigma}_i(Z_i^{(1)})(W_1). \tag{A.80}$$

Now the second deterministic component can also be approximated with a forward-Euler discretisation

$$Z_{i+1}^{(2)} = Z_i^{(2)} + \int_i^{i+1} \overleftarrow{\mu}_{2,t}(Z_t^{(2)})dt \approx Z_i^{(2)} + \overleftarrow{\mu}_{2,i}(Z_i^{(2)}). \tag{A.81}$$

As $Z_0 = 0$, we need only show the update at a time i, so assume we have Z_i. First we update in the SDE step, so make the assignment and update

$$Z_i^{(1)} \leftarrow Z_i, \qquad Z_{i+1}^{(1)} = Z_i^{(1)} + \overleftarrow{\mu}_{i,1}(Z_i^{(1)}) + \overleftarrow{\sigma}_i(Z_i^{(1)})\Delta W_1. \tag{A.82}$$

Now assign $Z_i^{(2)} \leftarrow Z_{i+1}^{(1)}$ so we may update in the mean direction with

$$Z_{i+1}^{(2)} = Z_i^{(2)} + \overleftarrow{\mu}_{i,2}(Z_i^{(2)}), \tag{A.83}$$

with the total update $Z_{i+1} \leftarrow Z_{i+1}^{(2)}$. This gives the cell update for NVAE in Figure A.1. To help enforce the growth $\|Z_t\|_2 > \|Z_s\|_2$, VDVAE splits $Z_i^{(1)} = Z_i + Z_{i,+}$ where $Z_{i,+}$ increases the norm of the latent process Z_t. This connection and the associated update are illustrated in Figure A.1 [left]. Note here that if no residual connection through the cell was used (just the re-parameterisation trick in a VAE), then we degenerate to a standard Markovian diffusion process and yield the Euler-Maruyama VAE structure in Figure A.1 [right].

Remark 1. To simplify the stepping notation in the HVAE backward cells (Figures 4 and A.1), we use $Z_i^{(1)} = Z_i + \overleftarrow{\mu}_{1,i}(Z_i) + \overleftarrow{\sigma}_i(Z_i)(W_1)$ and $Z_i^{(2)} = Z_i^{(1)} + \overleftarrow{\mu}_{i,2}(Z_i^{(1)})$ so that the index i refers to all computations of the i^{th} backward cell.

□

A.6 Time-homogenuous model

Recall VDVAE has the continuous time analogue

$$dZ_t = (\overleftarrow{\mu}_{1,t}(Z_t) + \overleftarrow{\mu}_{2,t}(Z_t))dt + \overleftarrow{\sigma}_t(Z_t)dW_t, \tag{A.84}$$

where $Z_0 = 0,\ \|Z_t\|_2 > \|Z_s\|_2$ with $0 \le s < t \le T$ and for a measure $\nu_J \in \mathbb{D}(V_{-J})$. Due to Theorem 5, we know that the initial condition of VDVAE's U-Net is the point mass $\delta_{\{0\}}$. As the backwards pass flows from zero to positive valued functions, this direction is increasing and the equation is stiff with few layers. The distance progression from zero is our proxy for time, and we can use its 'position' to measure this. Thus, the coefficients $\overleftarrow{\mu}_{t,1}$, $\overleftarrow{\mu}_{t,2}$, $\overleftarrow{\sigma}_t$ need not have a time dependence as this is already encoded in the norm of the Z_t processes. Thus, the time-homogeneous model postulated in the main text is:

$$dZ_t = (\overleftarrow{\mu}_1(Z_t) + \overleftarrow{\mu}_2(Z_t))dt + \overleftarrow{\sigma}(Z_t)dW_t, \tag{A.85}$$

$$Z_0 = 0, \; \|Z_t\|_2 > \|Z_s\|_2. \tag{A.86}$$

In practice, the loss of time dependence in the components corresponds to weight sharing the parameters across time, as explored in the experimental section. Weight sharing, or a time-homogeneous model, is common for score based diffusion models [40, 41], and due to our identification we are able to utilise this for HVAEs.

A.7 HVAE Sampling

Here we use our framework to comment on the sampling distribution imposed by the U-Net within VDVAE.

Theorem 5. Consider the SDE in Eq. (A.76), trained through the ELBO in Eq. B.101. Let $\tilde{\nu}_J$ denote the data measure and $\nu_0 = \delta_{\{0\}}$ be the initial multi-resolution bridge measure imposed by VDVAEs. If $q_{\phi,j}$ and $p_{\theta,j}$ are the densities of $B_{\phi,1|j}\boldsymbol{F}_{J|1}\tilde{\nu}_J$ and $B_{\theta,1|j}\nu_0$ respectively, then a VDVAE optimises the boundary condition $\min_{\theta,\phi} KL(q_{\phi,0,1}||q_{\phi,0}p_{\theta,1})$, where a double index indicates the joint distribution.

Proof. We need to only show two things. First, due to Theorems 3 and 4, we know that the architecture imposes

$$\mathrm{proj}_{V_1}\overline{F}_{\phi,J|1}\nu_J \otimes \bigotimes_{j=1}^{J-1} \delta_{\{0\}}, \tag{A.87}$$

so we must analyse how $\mathrm{proj}_{V_1}\overline{F}_{\phi,J|1}\nu_J$ is trained. Second, we use Theorem 4 to view the discretised version of the continuous problem, and identify the error in the two-step forward Euler splitting.

On the first point, VDVAE uses an ELBO reconstruction with a KL divergence between the backwards pass of the data $\overline{B}_{\phi,J|1}\overline{F}_{\phi,J|1}\tilde{\nu}_J$ (the 'q_ϕ-distribution'), and the backwards pass of the model imposed by the U-Net $\overline{B}_{\phi,J|1}\nu_0$ (the 'p_θ distribution'). As Z_0 is zero initialised, we know $\nu_0 = \delta_{\{0\}}$. We need to show the cost function used imposes this initialisation on $\overline{B}_{\phi,1|0}\overline{F}_{\phi,J|0}\tilde{\nu}_J$. Let $X_T \sim \overline{F}_{\phi,J|0}\tilde{\nu}_J$, call the distribution of this $q_{\phi,0}$. We also use $Z_1 \sim q_{\phi,1}$ for a sample from $\overline{B}_{\phi,1|0}\overline{F}_{\phi,J|0}\tilde{\nu}_J$ and $Z_1 \sim p_{\theta,1}$ for a sample from $\overline{B}_{\phi,1|0}\nu_0$. For a realisation x of X_T, VDVAE computes

$$KL(q_{\phi,1|0}(\,\cdot\,|X_T = x)||p_{\theta,1|0}(\,\cdot\,|Z_0 = 0)) = KL(q_{\phi,1|0}(\,\cdot\,|X_T = x)||p_{\theta,1}(\,\cdot\,)), \tag{A.88}$$

which in training is weighted by each datum, so the total cost in this term is

$$\int KL(q_{\phi,1|0}(\,\cdot\,|X_T = x)||p_{\theta,1}(\,\cdot\,))q_{\phi,0}(X_T = x)dx. \tag{A.89}$$

But this is equal to,

$$\int\int \log\left(\frac{q_{\phi,1|0}(Z_1 = z|X_T = x)}{p_{\theta,1}(Z_1 = z_1)}\right) q_{\phi,1|0}(Z_1 = z|X_T = x)q_{\phi,0}(X_T = x)dzdx \tag{A.90}$$

$$= \int\int \log\left(\frac{q_{\phi,0,1}(Z_1 = z, X_T = x)}{p_{\theta,1}(Z_1 = z_1)q_{\phi,0}(X_T = x)}\right) q_{\phi,0,1}(Z_1 = z, X_T = x)dzdx \tag{A.91}$$

$$= KL(q_{\phi,0,1}(Z_1, X_T)||p_{\theta,1}(Z_1)q_{\phi,0}(X_T)) = KL(q_{\phi,0,1}||p_{\theta,1}q_{\phi,0}). \tag{A.92}$$

The distribution of $p_{\theta,1}$ is Gaussian as a one time step diffusion evolution from the initial point mass $\nu_0 = \delta_{\{0\}}$.

□

Theorem 1 states that the choice of the initial latent variable in VDVAE imposes a boundary condition on the continuous SDE formulation. Further, this boundary condition is enforced into the final output X_T of the encoder within VDVAE.

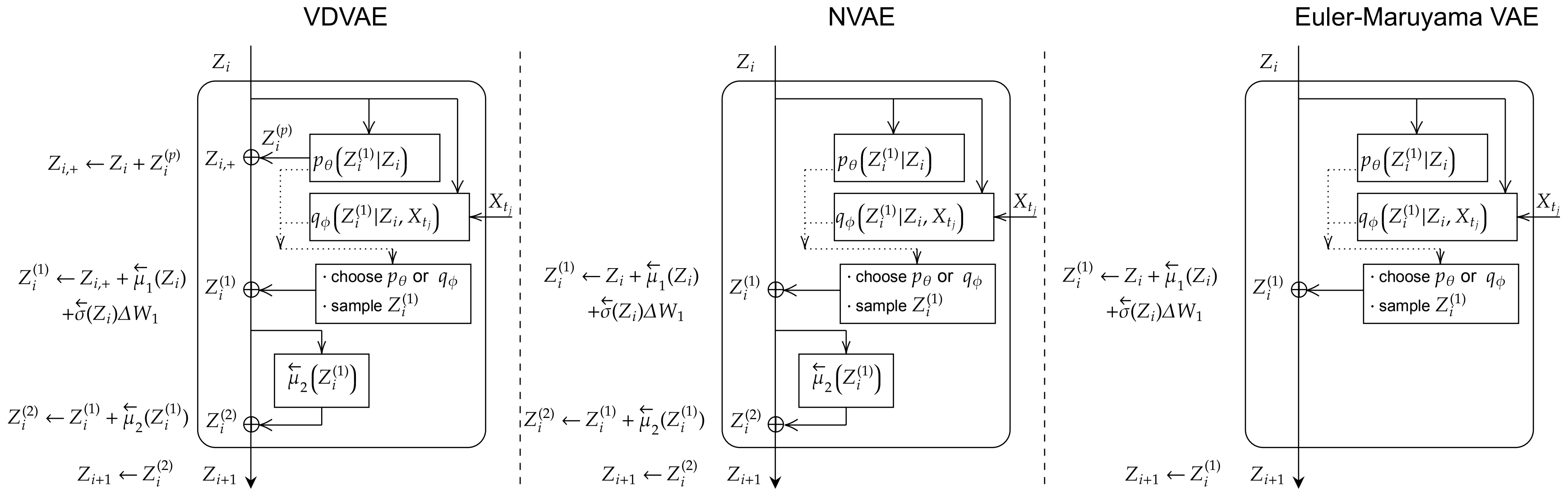

VDVAE
NVAE
Euler-Maruyama VAE
Z_i
$Z_{i,+} \leftarrow Z_i + Z_i^{(p)}$
$Z_{i,+}$
$Z_i^{(p)}$
$p_\theta\left(Z_i^{(1)}|Z_i\right)$
$q_\phi\left(Z_i^{(1)}|Z_i, X_{t_j}\right)$
X_{t_j}
$Z_i^{(1)} \leftarrow Z_{i,+} + \overleftarrow{\mu}_1(Z_i) + \overleftarrow{\sigma}(Z_i)\Delta W_1$
$Z_i^{(1)}$
· choose p_θ or q_ϕ
· sample $Z_i^{(1)}$
$\overleftarrow{\mu}_2\left(Z_i^{(1)}\right)$
$Z_i^{(2)} \leftarrow Z_i^{(1)} + \overleftarrow{\mu}_2(Z_i^{(1)})$
$Z_i^{(2)}$
$Z_{i+1} \leftarrow Z_i^{(2)}$
Z_{i+1}
$Z_i^{(1)} \leftarrow Z_i + \overleftarrow{\mu}_1(Z_i) + \overleftarrow{\sigma}(Z_i)\Delta W_1$
$Z_{i+1} \leftarrow Z_i^{(1)}$

Figure A.1: HVAE top-down cells are resembling two-step forward Euler discretisations of a continuous-time diffusion process in Eq. A.76. We here provide the residual cell structures of [left] VDVAE [9], [middle] NVAE [10] and [right] a Euler-Maruyama VAE. Either q_ϕ (conditional) or p_θ (unconditional) are used in the sampling step (indicated by the dotted lines) during training and generation, respectively. X_{t_j} is an input from the in effect non-stochastic bottom-up pass, Y_i is the input from the previous, and Y_{i+1} the output to the next residual cell. $\oplus$ indicates element-wise addition.

B Background

B.1 Multi-Resolution Hierarchy and thought experiment

Let $\mathbb{X} \subset \mathbb{R}^m$ be compact and $L^2(\mathbb{X})$ be the space of square-integrable functions over this set. We are interested in decomposing $L^2(\mathbb{X})$ across multiple resolutions.

Definition 1 (abbreviated). A *multi-resolution hierarchy* is one of the form

$$\cdots \subset V_1 \subset V_0 \subset V_{-1} \subset \cdots \tag{B.93}$$

$$\overline{\bigcup_{j\in\mathbb{Z}} V_{-j}} = L^2(\mathbb{R}^m) \tag{B.94}$$

$$\bigcap_{j\in\mathbb{Z}} V_{-j} = \{0\} \tag{B.95}$$

$$f(\cdot) \in V_{-j} \iff f(2^j \cdot\,) \in V_0 \tag{B.96}$$

$$f(\cdot) \in V_0 \iff f(\,\cdot - n) \in V_0, \text{ for } n \in \mathbb{Z}. \tag{B.97}$$

Each V_{-j} is a finite truncation of $L^2(\mathbb{X})$. What we are interested in is to consider a function $f \in L^2(\mathbb{X})$ and finding a finite dimensional approximation in V_{-J}, say, for $J > 0$. Further, for gray-scale images, $\mathbb{X} = [0,1]^2$, the space of pixel-represented images. To simplify notation, we just consider $\mathbb{X} = [0,1]$ for the examples below, but we can extend this to gray-scale images, and to colour images with a Cartesian product.

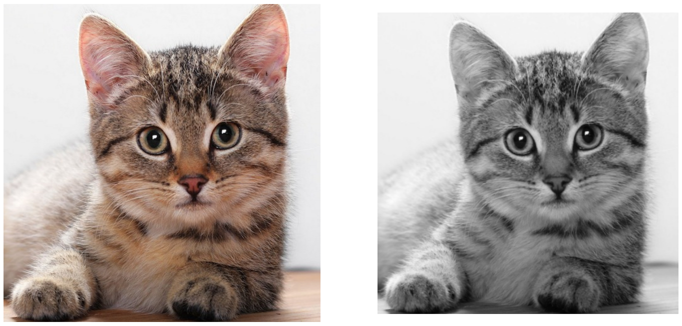
The 'pixel' multi-resolution hierarchy is given by the collection of sub-spaces

$$V_{-j} = \{f \in L^2([0,1]) \mid f|_{[2^{-j}\cdot k, 2^{-j}\cdot(k+1))} = c_k,\ k \in \{0,\dots,2^j-1\},\ c_k \in \mathbb{R}\}. \tag{B.98}$$

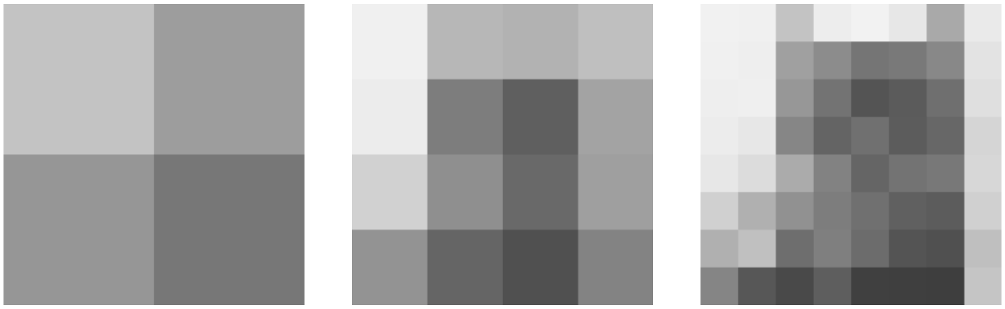
It can be readily checked that these sub-spaces obey the assumptions of Definition 1. An example image projected into such sub-spaces, obtained from a discrete Haar wavelet transform, is illustrated in Fig. B.2. We call it the pixel space of functions as elements of this set are piece-wise constant on dyadically split intervals of resolution 2^j, i.e a pixelated image. For each V_{-j} there is an obvious basis of size 2^j where we store the coefficients $(c_0, c_1, \dots, c_{2^j-1}) \in \mathbb{R}^{2^j}$. The set of basis vectors for it is the *standard basis* $\{e_i\}_{i=0}^{2^j-1}$ which are 0 for all co-ordinates except for the $i^{th}+1$ entry which is 1. This basis is not natural to the multi-resolution structure of V_{-j}. This is because all the basis functions change when we project down to V_{-j+1}. We want to use the multi-resolution structure to create a basis which naturally relates V_{-j}, V_{-j+1}, and any other sub-space. To do this consider $V_{-j} \cap V_{-j+1}^{\perp} \subset V_{-j}$. Define this orthogonal compliment to be $U_{-j+1} := V_{-j+1}^{\perp}$, then see $V_{-j} = V_{-j+1} \oplus U_{-j+1}$. Doing this recursively finds $V_{-j} = V_0 \oplus \bigoplus_{i=0}^{-j+1} U_i$, and taking the limit

$$L^2(\mathbb{X}) = \bigoplus_{i=0}^{-\infty} U_i \oplus V_0. \tag{B.99}$$

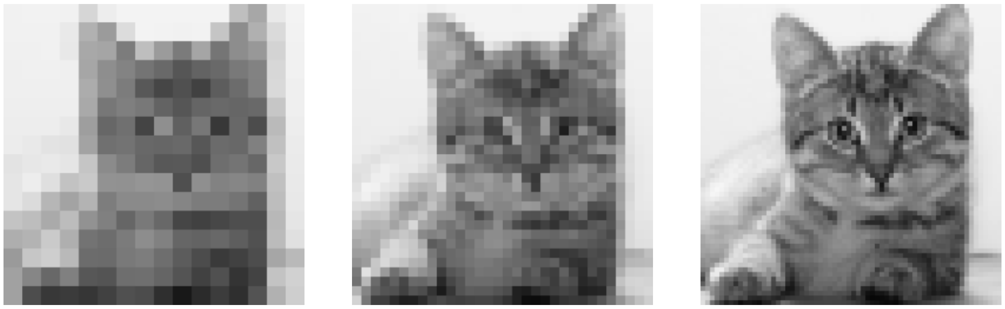
Each of the sub-spaces $\{U_{-j}\}_{j=0}^{\infty}$ are mutually orthogonal as each $V_{-j} \perp U_{-j}$. Now suppose we had a basis set $\mathbf{\Psi}_j$ for each U_{-j} and $\mathbf{\Phi}_0$ for V_0. As these spaces are orthogonal, so are the basis sets to each other, too. We can make a basis for V_{-j} with span$(\mathbf{\Phi}_0, \mathbf{\Psi}_0, \cdots, \mathbf{\Psi}_{-j+1})$. For the above examples, V_0 needs only a single basis function $\phi_{0,k} = \mathbb{1}_{[k,k+1)}$, further if $\psi = \sqrt{2}(\mathbb{1}_{[0,1/2)} - \mathbb{1}_{[1/2,1)})$, then given the functions $x \mapsto \psi_{j,k}(x) := 2^{j/2} \cdot \psi(2^{-j}(x-k))$ we have $\{\psi_{j,k}\}$ is a basis for V_{-j}.



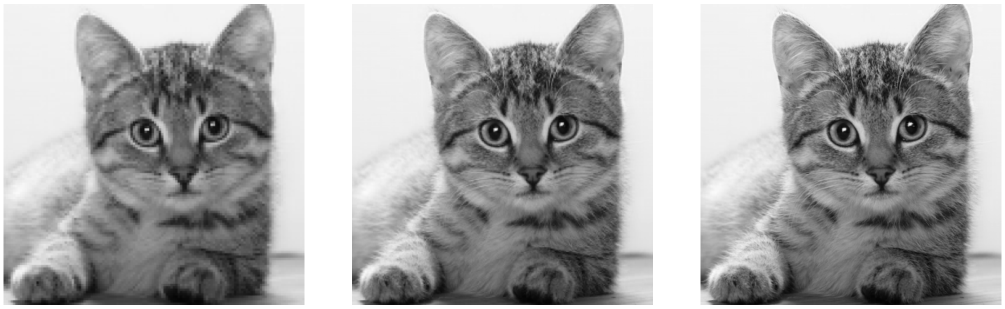
original (512×512) gray-scaled original (512×512)



V_{-1} (2×2) V_{-2} (4×4) V_{-3} (8×8)



V_{-4} (16×16) V_{-5} (32×32) V_{-6} (64×64)



V_{-7} (128×128) V_{-8} (256×256) V_{-9} (512×512)

Figure B.2: The thought experiment discussed in §2. The original colour image [top-left], its gray-scale version [top-right], and its Haar wavelet projections to the approximation spaces V_{-j} for $j \in \{1, \ldots, 9\}$.

B.2 U-Net

In practice, a U-Net [1] is a neural network structure which consists of a forward pass (encoder) and backward pass (decoder), wherein layers in the forward pass have some form of dimension reduction, and layers in the backward pass have some form of dimension embedding. Furthermore, there are 'skip connection' between corresponding layers on the same level of the forward and backward pass.

We now formalise this notion, referring to an illustration of a U-Net in Fig. B.3. In black, we label the latent spaces to be V_{-j} for all j, where the original data is in V_{-J} and the U-Net 'bend' (bottleneck) occurs at V_0. We use $f_{j,\theta}$ to be the forward component, or encoder, of the U-Net, and similarly $b_{j,\theta}$ as the backward component or decoder, operating on the latent space V_{-j}. P_{-j+1} refers to the dimension reduction operation between latent space V_{-j} and V_{-j+1}, and E_{-j} refers to its corresponding dimension embedding operation between latent spaces V_{-j+1} and V_{-j}. A standard dimension reduction operation in practice is to take P_{-j+1} as average pooling, reducing the resolution of an image. Similarly, the embedding step may be some form of deterministic interpolation of the image to a higher resolution. We note that the skip connection in Fig. B.3 occur before the dimension reduction step, in this sense, lossless information is fed from the image of $f_{j,\theta}$ into the domain of $b_{j,\theta}$.

In blue, we show another backward process $b_{j,\phi}$ that is often present in U-Net architectures for generative models. This second backward process is used for unconditional sampling. In the context of HVAEs, we may refer to it as the (hierarchical) prior (and likelihood model). It is trained to match its counterpart in black, without any information from the forward process. In HVAEs, this is enforced by a KL-divergence between distributions on the latent spaces V_{-j}. The goal of either backward process is as follows:

$$\begin{tikzcd}
V_{-J} \arrow[d, dash, dotted] & V_{-J} \arrow[d, dash, dotted] & V_{-J} \arrow[d, dash, dotted] \\
\text{out} & \text{out} & \text{in} \arrow[d] \\
V_{-j} \arrow[u] & V_{-j} \arrow[u] & V_{-j} \\
V_{-j} \arrow[u, "B_{j,\phi}"] & V_{-j} \arrow[u, "B_{j,\theta}"] & V_{-j} \arrow[u, leftarrow, "F_{j,\theta}"'] \arrow[l, dashed, "\text{Skip connection}"'] \arrow[d, "P_{-j+1}"] \\
V_{-j+1} \arrow[u] & V_{-j+1} \arrow[u, "E_{-j}"] & V_{-j+1} \arrow[d] \\
\text{out} \arrow[u] & \text{out} \arrow[u] & \text{in} \\
V_0 \arrow[u, dash, dotted] & V_0 \arrow[u, dash, dotted] & V_0 \arrow[u, dash, dotted] \arrow[l, "\text{Bend / Bottleneck}"']
\end{tikzcd}$$

Figure B.3: The repeated structure in a U-Net, where V_{-j+1} is a lower dimensional latent space compared to V_{-j}. $f_{j,\theta}, b_{j,\theta}$ are in practice typically parameterised by neural networks (e.g. convolutional neural networks); P_{-j+1} is a dimension reduction operation (e.g. average pooling) to a lower-dimensional latent space; and, E_{-j} is a dimension embedding operation (e.g. deterministic interpolation) to a higher-dimensional latent space. This structure is repeated to achieve a desired dimension of the latent space at the U-Net bottleneck.

1. $b_{j,\theta}$ must be able to reconstruct the data from $f_{j,\theta}$, and in this sense it is reasonable to require $b_{j,\theta} f_{j,\theta} = I$;
2. $b_{j,\phi}$ must connect the data to a known sampling distribution.

The second backward process can be absent when the backward process $b_{j,\theta}$ is imposed to be the inverse of $f_{j,\theta}$, such as in Normalising Flow based models, or reversible score-based diffusion models. In this case the invertibility is assured, and the boundary condition that the encoder connects to a sampling distribution must be enforced. For the purposes of our study, we will assume that in the absence of dimension reduction, the decoder is constrained to be an inverse of the encoder. This is a reasonable assumption: for instance, in HVAEs near perfect data reconstructions are readily achieved.

For variational autoencoders, the encoder and decoder are not necessarily deterministic and involve resampling. To encapsulate this, we will work with the data as a measure and have $F_{\theta,j}$ and $B_{\theta,j}$ as the corresponding kernels imposed by $f_{j,\theta}$ and $b_{j,\theta}$, respectively.

With all of these considerations in mind, for the purposes of our framework we provide a definition of an idealised U-Net which is an approximate encapsulation of all models using a U-Net architecture.

Definition 3. (Idealised U-Net for generative modelling)
For each $j \in \{0, \ldots, J\}$, let $F_{j,\theta}, B_{j,\theta} : \mathbb{D}(V_{-j}) \mapsto \mathbb{D}(V_{-j})$ such that $B_{j,\theta} F_{j,\theta} \equiv I_{V_{-j}}$. A U-Net

with (average pooling) dimension reduction P_{-j+1} and dimension embedding E_{-j} is the operator $\mathbf{U} : \mathbb{D}(V_{-J}) \mapsto \mathbb{D}(V_{-J})$ given by

$$\mathbf{U} \coloneqq B_{J,\theta}E_{-J} \circ \cdots \circ B_{1,\theta}E_{-1} \circ P_0F_{1,\theta} \circ \cdots \circ P_{-J+1}F_{J,\theta}, \qquad B_jF_j \equiv I. \tag{B.100}$$

Remark 2. The condition $B_{j,\theta}F_{j,\theta} \equiv I_{V_{-j}}$ in our idealised U-Net (for unconditional sampling here) is either imposed directly (reversible flow based model), or approximated via skip connections. For instance, in our HVAE case, we have both a U-Net without skip connections (the p distribution) and a U-Net with skip connections (the q distribution). The U-Net related to the q distribution learns how to reconstruct our data from the reconstruction term in the ELBO cost function. The U-Net related to the p distribution learns to mimic the q distribution via the KL term in the ELBO of the HVAE, whose decoder is trained to invert its encoder — $B_{j,\phi}F_{j,\phi} \equiv I_{V_{-j}}$ — but the p U-Net lacks skip connections. Thus, in the HVAE context, we are analysing U-Nets which must simultaneously reconstruct our data and lose their reliance on their skip connections due to the condition that the q U-Net must be approximately equal to the p U-Net.

B.3 Hierarchical VAEs

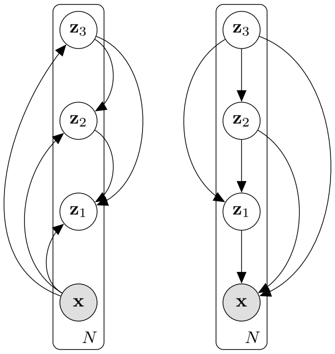


Figure B.4: Conditioning structure in state-of-the-art HVAE models (VDVAE[9] / NVAE[10]) with $L = 3$. [Left] Amortised variational posterior $q_\theta(\vec{\mathbf{z}} \mid \mathbf{x})$. [Right] Generative model $p_\phi(\mathbf{x}, \vec{\mathbf{z}})$.

A *hierarchical Variational Autoencoder (HVAE)* [4] is a VAE [48] where latent variables are separated into L groups $\vec{\mathbf{z}} = (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_L)$ which conditionally depend on each other. L is often referred to as stochastic depth. For convenience, we refer to the observed variable $\mathbf{x}$ as $\mathbf{z}_0$, so $\mathbf{x} \equiv \mathbf{z}_0$. In HVAEs, latent variables typically follow a 'bow tie', U-Net [1] type architecture with an information bottleneck [53], so $\dim(\mathbf{z}_{l+1}) \leq \dim(\mathbf{z}_l)$ for all $l = 0, \ldots, L-1$. Latent variables live on multiple *resolutions*, either decreasing steadily [36, 54] or step-wise every few stochastic layers [10, 9] in dimension. We consider this multi-resolution property an important characteristic of HVAEs. It distinguishes HVAEs from other deep generative models, in particular vanilla diffusion models where latent and data variables are of equal dimension [13].

As in a plain VAE with only a single group of latent variables, an HVAE has a likelihood $p_\phi(\mathbf{x}|\vec{\mathbf{z}})$, a prior $p_\phi(\vec{\mathbf{z}})$ and an approximate posterior $q_\theta(\vec{\mathbf{z}}|\mathbf{x})$. To train the HVAE, one optimises the ELBO w.r.t. parameters ϕ and θ via stochastic gradient descent using the reparametrization trick

$$\log p(\mathcal{D}) \geq \mathcal{L}(\mathcal{D}; \theta, \phi) = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}} \underbrace{[\mathbb{E}_{\vec{\mathbf{z}}\sim q_\theta(\vec{\mathbf{z}}|\mathbf{x})}\left[\log p_\phi(\mathbf{x}|\vec{\mathbf{z}})\right]}_{\text{Reconstruction loss}} - \underbrace{\mathrm{KL}[q_\theta(\vec{\mathbf{z}}|\mathbf{x})||p_\phi(\vec{\mathbf{z}})]]}_{\text{Prior loss}}. \tag{B.101}$$

Numerous conditioning structures of the latent variables in HVAEs exist, and we review them in §4. In this work, we follow [9, 10, 28]: the latent variables in the prior and approximate posterior are estimated in the same order, from $\mathbf{z}_L$ to $\mathbf{z}_1$, conditioning 'on all previous latent variables', i.e.

$$p_\phi(\vec{\mathbf{z}}) = p_\phi(\mathbf{z}_L) \prod_{l=1}^{L-1} p_\phi(\mathbf{z}_l|\mathbf{z}_{>l}) \quad \text{(B.102)} \qquad q_\theta(\vec{\mathbf{z}}|\mathbf{x}) = q_\theta(\mathbf{z}_L|\mathbf{x}) \prod_{l=1}^{L-1} q_\theta(\mathbf{z}_l|\mathbf{z}_{>l}, \mathbf{x}) \quad \text{(B.103)}$$

We visualise the graphical model of this HVAE in Fig. B.4. Recent HVAEs [9, 10] capture this

---

[4] We closely follow the introduction of hierarchical VAEs in [9, §2.2].

dependence on all previous latent variables $\mathbf{z}_{>l}$ in their residual state as shown in §2.3, imposing this conditional structure. This implies a 1st-order Markov chain conditional on the previous residual state, not the previous $\mathbf{z}_l$. Such 1st-order Markov processes have shown great success empirically, such as in LSTMs [38]. Further, note that in all previous work on HVAEs, the neural networks estimating the inference and generative distributions of the l-th stochastic layer are *not* sharing parameters with those estimating other stochastic layers.

Intuitively, HVAEs' conditional structure together with a U-Net architecture imposes an inductive bias on the model to learn a *hierarchy* of latent variables where each level corresponds to a different degree of abstraction. In this work, we characterise this intuition via the regularisation property of U-Nets in §2.2.

The distributions over the latent variables in both the inference and generative model are Gaussian with mean $\boldsymbol{\mu}$ and a diagonal covariance matrix $\boldsymbol{\Sigma}$, i.e. for all $l = 1, \dots, L$,

$$q_\theta(\mathbf{z}_l|\mathbf{z}_{>l}, \mathbf{x}) \sim \mathcal{N}(\boldsymbol{\mu}_{l,\theta}, \boldsymbol{\Sigma}_{l,\theta}), \tag{B.104}$$

$$p_\phi(\mathbf{z}_l|\mathbf{z}_{>l}) \sim \mathcal{N}(\boldsymbol{\mu}_{l,\theta}, \boldsymbol{\Sigma}_{l,\theta}), \tag{B.105}$$

where mean and variances are estimated by neural networks with parameters ϕ and θ corresponding to stochastic layer l. Note that $p_\phi(\mathbf{z}_L|\mathbf{z}_{>l}) = p_\phi(\mathbf{z}_L)$, where the top-down block estimating $p_\phi(\mathbf{z}_L)$ receives the zero-vector as input, and $q_\theta(\mathbf{z}_L|\mathbf{z}_{>L}, \mathbf{x}) = q_\theta(\mathbf{z}_L|\mathbf{x})$, meaning that we infer without conditioning on other latent groups at the L-th step. Further, VDVAE chooses $p_\phi(\mathbf{x}|\vec{\mathbf{z}})$ to be a discretized Mixture-of-Logistics likelihood.

B.4 Sampling of Time Steps in HVAEs

Monte Carlo sampling of time steps in ELBO of HVAEs.

We here provide one additional theoretical result. We show that the ELBO of an HVAE can be written as an expected value over uniformly distributed time steps.

Previous work [19] [41] (Eq. (13), respectively) showed that the diffusion loss term $\mathcal{L}_T(\mathbf{x})$ in the ELBO of discrete-time diffusion models can be written as

$$\mathcal{L}_T(\mathbf{x}) = \frac{T}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}), i\sim U\{1,T\}}\left[(\mathrm{SNR}(s) - \mathrm{SNR}(t))\left\|\mathbf{x} - \hat{\mathbf{x}}_\theta\left(\mathbf{z}_t; t\right)\right\|_2^2\right] \tag{B.106}$$

which allows maximizing the variational lower-bound via a Monte Carlo estimator of Eq. B.106, sampling time steps.

Inspired by this result for diffusion models, we provide a similar form of the ELBO for an HVAE with factorisation as in Eqs. (B.102)-(B.103) (and the graphical model in Fig. B.4). An HVAE's ELBO can be written as

$$\log p(\mathbf{x}) \geq \mathbb{E}_{\vec{\mathbf{z}}\sim q(\vec{\mathbf{z}}|\mathbf{x})}\left[\log p(\mathbf{x}|\vec{\mathbf{z}})\right] - L\,\mathbb{E}_{l\sim\mathrm{Unif}(1,L)}\left[\mathbb{E}_{\vec{\mathbf{z}}\sim q(\vec{\mathbf{z}}|\mathbf{x})}\log\frac{q(\mathbf{z}_l|\mathbf{z}_{>l}, \mathbf{x})}{p(\mathbf{z}_l|\mathbf{z}_{>l})}\right].$$

Proof.

$$\begin{aligned}
\log p(\mathbf{x}) &\geq \mathbb{E}_{\vec{\mathbf{z}}\sim q(\vec{\mathbf{z}}|\mathbf{x})}\left[\log p(\mathbf{x}|\vec{\mathbf{z}})\right] - \mathrm{KL}\left[q(\vec{\mathbf{z}}|\mathbf{x})||p(\vec{\mathbf{z}})\right] \\
&= \mathbb{E}_{\vec{\mathbf{z}}\sim q(\vec{\mathbf{z}}|\mathbf{x})}\left[\log p(\mathbf{x}|\vec{\mathbf{z}})\right] - \int \mathrm{d}\vec{\mathbf{z}} q(\vec{\mathbf{z}}|\mathbf{x})\log\left[\frac{\prod_{l=1}^{L} q(\mathbf{z}_l|\mathbf{z}_{>l}, \mathbf{x})}{\prod_{l=1}^{L} p(\mathbf{z}_l|\mathbf{z}_{>l})}\right] \\
&= \mathbb{E}_{\vec{\mathbf{z}}\sim q(\vec{\mathbf{z}}|\mathbf{x})}\left[\log p(\mathbf{x}|\vec{\mathbf{z}})\right] - \sum_{l=1}^{L}\int \mathrm{d}\vec{\mathbf{z}} q(\vec{\mathbf{z}}|\mathbf{x})\log\left[\frac{q(\mathbf{z}_l|\mathbf{z}_{>l}, \mathbf{x})}{p(\mathbf{z}_l|\mathbf{z}_{>l})}\right] \\
&= \mathbb{E}_{\vec{\mathbf{z}}\sim q(\vec{\mathbf{z}}|\mathbf{x})}\left[\log p(\mathbf{x}|\vec{\mathbf{z}})\right] - L\,\mathbb{E}_{l\sim\mathrm{Unif}(1,L)}\left[\mathbb{E}_{\vec{\mathbf{z}}\sim q(\vec{\mathbf{z}}|\mathbf{x})}\log\frac{q(\mathbf{z}_l|\mathbf{z}_{>l}, \mathbf{x})}{p(\mathbf{z}_l|\mathbf{z}_{>l})}\right].
\end{aligned}$$

□

This allows reducing the computational and memory costs of the KL-terms in the loss and depends on how many Monte Carlo samples are drawn. However, in contrast to diffusion models, all intermediate stochastic layers (up to the top-most and bottom-most layer chosen when sampling time steps in the recognition and generative model, respectively) still need to be computed as each latent variable's distribution depends on all previous ones.

C Code, computational resources, existing assets used

Code. We provide our PyTorch code base at https://github.com/FabianFalck/unet-vdvae. Our implementation is based on, modifies and extends the official implementation of VDVAE [9]. Below, we highlight key contributions:

- We implemented weight-sharing of individual ResNet blocks for a certain number of repetitions.
- We implemented the datasets and the preprocessing of MNIST and CelebA, which were previously not used with VDVAE.
- We implemented the option of synchronous and asynchronous processing in time (see Appendix G.4.5).
- We implemented Fourier features with hyperparameters choosing their frequencies following VDM [19]. One can concatenate them at three different locations as options.
- We simplified the multi-GPU implementation.
- We implemented an option to convert the VDVAE cell into a non-residual cell (see Appendix G.4.4).
- We implemented logging of various metrics and plots with weight&biases.
- We implemented gradient checkpointing [55] as an option in the decoder of VDVAE where the bulk of the computation occurs. We provide two implementations of gradient checkpointing, one based on the official PyTorch implementation which is unfortunately slow when using multiple GPUs, and a prototype for a custom implementation based on https://github.com/csrhddlam/pytorch-checkpoint.

The `README.md` contains instructions on installation, downloading the required datasets, the setup of weights&biases, and how to reproduce our main results.

Computational resources. For the majority of time during this project, we used two compute clusters: The first cluster is a Microsoft Azure server with two Nvidia Tesla K80 graphic cards with 11GB of GPU memory each, which we had exclusive access to. The second cluster is an internal cluster with 12 Nvidia GeForce GTX 1080 graphic cards and 10GB of GPU memory each, shared with a large number of users. In the late stages of the project, in particular to perform runs on ImageNet32, ImageNet64 and CelebA, we used a large-scale compute cluster with A100 graphic cards with 40GB of GPU memory each. We refer to the acknowledgements section for further details.

In the following, we provide a rough estimate of the total compute required to reproduce our main experiments. Compute time until convergence scales with the depth of the HVAEs. For the shallower HVAEs in our small-scale experiments in §G.1, training times range from several days to a week. For our larger-scale experiments on MNIST and CIFAR10, training times range between 1 to 3 weeks. For our deepest runs on ImageNet32 and CelebA, training times range between 2.5 to 4 weeks.

For orientation, in Table C.1, we provide an estimate of the training times of our large-scale runs in Table 1. We note that these runs have been computed on different hardware, i.e. the training times are only to some degree comparable, yet give an indication.

Existing assets used. In the experiments, our work directly builds on top of the official implementation of VDVAE [9] (MIT License). We use the datasets reported in Appendix D. In our implementation, we make use of the following existing assets and list them together with their licenses: PyTorch [56], highlighting the torchvision package for image benchmark datasets, and the gradient checkpointing implementation (custom license), Numpy [57] (BSD 3-Clause License) Weights&Biases [58] (MIT License), Apex [59] (BSD 3-Clause "New" or "Revised" License), Pickle [60] (license not available), Matplotlib [61] (PSF License), ImageIO [62] (BSD 2-Clause "Simplified" License), MPI4Py [63] (BSD 2-Clause "Simplified" License), Scikit-learn [64] (BSD 3-Clause License), and Pillow [65] (custom license).

D Datasets

In our experiments, we make use of the following datasets: MNIST [42], CIFAR10 [43], ImageNet32 [44, 45], ImageNet64 [44, 45], and CelebA [46]. We briefly discuss these datasets, focussing on

Table C.1: A large-scale study of parameter efficiency in HVAEs. For all our runs in Table 1, we report their stochastic depth and estimated training time.

	Method	Depth	Training time
MNIST (28×28)			
	`WS-VDVAE` (ours)	57	≈ 5 days
	`VDVAE`* (ours)	43	≈ 5 days
CIFAR10 (32×32)			
	`WS-VDVAE` (ours)	268	≈ 18 days
	`WS-VDVAE` (ours)	105	≈ 13 days
	`VDVAE`* (ours)	43	≈ 9 days
ImageNet (32×32)			
	`WS-VDVAE` (ours)	169	≈ 20 days
	`WS-VDVAE` (ours)	235	≈ 24 days
	`VDVAE`* (ours)	78	≈ 16 days
CelebA (64×64)			
	`WS-VDVAE` (ours)	125	≈ 27 days
	`VDVAE`* (ours)	75	≈ 21 days

their preprocessing, data splits, data consent and commenting on potential personally identifiable information or offensive content in the data. We refer to the training set as images used during optimisation, the validation set as images used to guide training (e.g. to compute evaluation metrics during training) but not used for optimisation directly, and the test set as images not looked at during training and only to compute performance of completed runs. For all datasets, we fix the training-validation-test split over different runs, and we scale images to be approximately centred and having a standard deviation of one based on statistics computed on the respective training set. If not stated otherwise, we use a modified version of the implementation of these datasets in [9].

MNIST. The MNIST dataset [42] contains gray-scale handwritten images of 10 digit classes ('0' to '9') with resolution 28×28. It contains $60,000$ training and $10,000$ test images, respectively. From the training images, we use $55,000$ images as the training set and 5000 images as the validation set. We use all $10,000$ test images as the testing set. We build on top of the implementation provided in NVAE [10] (`https://github.com/NVlabs/NVAE/blob/master/datasets.py`), which itself uses torchvision [56], and dynamically binarize the images, meaning that pixel values are binary, as drawn from a Bernoulli distribution with the probabilities given by the scaled gray-scale values in $[0, 1]$. Furthermore, we pad each image with zeros so to obtain the resolution 32×32.

The dataset is highly standardised and cropped to individual digits so that offensive content or personally identifiable information can be excluded. As the original NIST database from which MNIST was curated is no longer available, we cannot comment on whether consent was obtained from the subjects writing and providing these digits [27].

CIFAR10. The CIFAR10 dataset [43] contains coloured images from 10 classes ('airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck') with resolution 32×32. It contains $50,000$ training and $10,000$ test images, respectively. We split the training images into $45,000$ images in the training set and 5000 images in the validation set, and use all $10,000$ test images as the test set.

CIFAR10 was constructed from the so-called 80 million tiny images dataset by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton [66]. On the official website of the 80 million tiny images dataset, the authors state that this larger dataset was officially withdrawn by the authors on June 29th, 2020 due to offensive images being identified in it [67]. The authors of the 80 million tiny images dataset do not comment on whether CIFAR10, which is a subset of this dataset, likewise contains these offensive images or is unaffected. [43] states that the images in the 80 million tiny images dataset were retrieved by searching the web for specific nouns. The authors provide no information to which degree consent was obtained from the people who own these images.

ImageNet32. The ImageNet32 dataset, a downsampled version of the ImageNet database [44, 45], contains $1,281,167$ training and $50,000$ test images from $10,000$ classes with resolution 32×32.

From the training images, $5,000$ images as the validation set and the remaining $1,276,167$ as the training set, and further use all $50,000$ test images as the test set.

ImageNet is a human curated collection of images downloaded from the web via search engines. While ImageNet used Amazon Mechanical Turk to lable the images, we were unable to find information on processes which ensured no personally identifiable or offensive content was contained in the images, which is somewhat likely given the "in-the-wild" nature of the dataset. The ImageNet website states that the copyright of the images does not belong to authors of ImageNet.

ImageNet64. The ImageNet64 dataset, a second downsampled version of the ImageNet database [44, 45], likewise contains $1,281,167$ training and $50,000$ validation images with resolution 64×64. We use the same data splits as for ImageNet32. Refer to the above paragraph on ImageNet32 for discussion of personally identifiable information, offensive content and consent.

CelebA. The CelebA dataset [46] contains $162,770$ training, $19,867$ validation and $19,962$ test images with resolution 64×64 which we directly use as our training, validation and test set, respectively. Our implementation is a modified version of the one provided in NVAE [10] (`https://github.com/NVlabs/NVAE/blob/master/datasets.py`).

CelebA images are "obtained from the Internet". The authors state that these images are not the property of the authors of associated institutions [68]. As this dataset shows the faces of humans, these images are personally identifiable. We were unable to identify a process by which consent for using these images was obtained, or how potential offensive content was prevented.

E Potential negative societal impacts

Our work provides mainly theoretical and methodological contributions to U-Nets and HVAEs, and we hence see no direct negative societal impacts. Since U-Nets are widely used in applications, our theoretical results and any future work derived from them may downstream improve such applications, and thus also enhance their performance in malicious uses. In particular, U-Nets are widely used in generative modelling, and here, our work may have an effect on the quality of 'deep fakes', fake datasets or other unethical uses of generative modelling. For HVAEs, our work may inspire novel models which may lead to improved performance and stability of these models, also when used in applications with negative societal impact.

F Model and training details

On the stability of training runs. VDVAE uses several techniques to improve the training stability of HVAEs: First, gradient clipping [69] is used, which reduces the effective learning rate of a mini-batch if the gradient norm surpasses a specific threshold. Second, gradient updates are skipped entirely when gradient norms surpass a second threshold, typically chosen higher than the one for gradient clipping. Third, gradient updates are also skipped if the gradient update would cause an overflow, resulting in NaN values.

In spite of the above techniques to avoid deterioration of training in very deep HVAEs, particularly when using a lot of weight-sharing, we experienced stability problems during late stages of training. These were particularly an issue on CIFAR10 in the late stages of training (on average roughly after 2 weeks of computation time), and often resulted in NaN values being introduced or posterior collapse. We did not extensively explore ways to prevent these in order to do minimal changes compared to vanilla VDVAE. We believe that an appropriate choice of the learning rate (e.g. with a decreasing schedule in later iterations) in combination with other changes to the hyperparameters may greatly help with these issues, but principled fixes of, for instance, the instabilities identified in Theorem 5 are likewise important.

Gradient checkpointing, and other alternatives to reduce GPU memory cost. A practical limitation of training deep HVAEs (with or without weight-shared layers) is their GPU memory cost: Training deeper HVAEs means storing more intermediate activations in GPU memory during the forward pass, when memory consumption reaches its peak at the start of the backward pass. This limits the depth of the networks that can be trained on given GPU resources. To address this issue, particularly when training models which may not even fit on used hardware, we provide a prototype of a custom[5] *gradient checkpointing* implementation. Checkpointing occurs every few ResNet blocks which trades off compute for memory and can be used as an option. In gradient checkpointing, activations are stored only at specific nodes (checkpoints) in the computation graph, saving GPU memory, and are otherwise recomputed on-demand, requiring one additional forward pass per mini-batch [55]. Training dynamics remain unaltered. We note that other techniques exist specifically targeted at residual networks: For example, [70] propose to stochastically drop out entire residual blocks at training time [6]. This technique has two disadvantages: It changes training dynamics, and peak memory consumption varies between mini-batches, where particularly the latter is an inconvenient property for the practitioner as it may cause out-of-memory errors.

G Additional experimental details and results

In this section, we provide additional experimental details and results.

Hyperparameters and hyperparameter tuning. In the following, we describe the hyperparameters chosen in our experiments. As highlighted in the main text, we use the state-of-the-art hyperparameters of VDVAE [9] wherever possible. This was possible for CIFAR10, ImageNet32 and ImageNet64. On MNIST and CelebA, VDVAE [9] did not provide experimental results. For MNIST, we took the hyperparameters of CIFAR10 as the basis and performed minimal hyperparameter tuning, mostly increasing the batch size and tuning the number and repetitions of residual blocks. For CelebA, we used the hyperparameters of ImageNet64 with minimal hyperparameter tuning, focussing on the number and repetitions of the residual blocks. For all datasets, the main hyperparameter we tuned was the number and repetitions (through weight-sharing) of residual blocks ceteris paribus, i.e. without searching over the space of other important hyperparameters. As a consequence, it is likely that further hyperparameter tuning would improve performance as changing the number of repetitions changes (the architecture of) the model.

We provide three disjunct sets of hyperparameters: *global* hyperparameters (Table G.2), which are applicable to all runs, *data-specific* hyperparameters (Table G.3), which are applicable to specific datasets, and *run-specific* hyperparameters, which vary by run. The run-specific hyperparameters will be provided in the respective subsections of §G, where applicable.

In the below tables, 'factor of # channels in conv. blocks' refers to the multiplicative factor of the number of channels in the bottleneck of a (residual) block used throughout VDVAE. '# channels of z_l' refers to C in the shape [C, H, W] of the latent conditional distributions in the approximate posterior and prior, where height H and width W are determined by the resolution of latent $\boldsymbol{z}_l$. Likewise, '# channels in residual state' refers to C in the shape [C, H, W] of the residual state flowing through the decoder of VDVAE. 'Decay rate γ of evaluation model' refers to the multiplicative factor by which the latest model parameters are weighted during training to update the evaluation model.

Table G.2: Global hyperparameters.

factor of # channels in conv. blocks	0.25
Gradient skipping threshold	3000
Adam optimizer: Weight decay	0.01
Adam optimizer: β_1	0.9
Adam optimizer: β_2	0.9

G.1 "More from less": Parameter efficiency in HVAEs

In this experiment, we investigate the effect of repeating ResNet blocks in the bottom-up and top-down pass via weight-sharing. `rN` indicates that a ResNet block is repeated `N` times through weight-sharing where `r` is to be treated like an operator and `N` is a positive integer. In contrast, `xN`, already used in the official implementation of VDVAE, indicates `N` number of ResNet blocks without weight-sharing.

---

[5] Our implementation deviates from the official PyTorch implementation of gradient checkpointing which is slow when using multiple GPUs, and is based on https://github.com/csrhddlam/pytorch-checkpoint.

[6] This technique is called "stochastic depth" as the active depth of the network varies at random. In this work, however, we go with our earlier definition of this term which refers to the number of stochastic layers in our network, and thus avoid using its name to prevent ambiguities.

Table G.3: Data-specific hyperparameters.

Dataset	MNIST	CIFAR10	ImageNet32	ImageNet64	CelebA
Learning rate	0.0001	0.0002	0.00015	0.00015	0.00015
# iterations for learning rate warm-up	100	100	100	100	100
Batch size	200	16	8	4	4
Gradient clipping threshold	200	200	200	220	220
# channels of $\boldsymbol{z}_l$	8	16	16	16	16
# channels in residual state	32	384	512	512	512
Decay rate γ of evaluation model	0.9999	0.9999	0.999	0.999	0.999

In Tables G.4 and G.5, we provide the NLLs on the test set at convergence corresponding to the NLLs on the validation set during training which we reported in Fig. 5. In general, weight-sharing tends to improve NLL, and models with significantly less parameters reach or even surpass other models with more parameters. We refer to the main text for the intuition of this behavior. In Table G.5 (CIFAR10), we find that the not weight-sharing runs have test NLLs noticeably deviating from the results on the validation set, yet the overall trend of more weight-sharing improving NLL tends to be observed. This is in line with our general observation that our HVAE models are particularly unstable on CIFAR10.

Table G.4: A small-scale study on parameter efficiency of HVAEs on *MNIST*. We compare models with one, two, three and four parameterised blocks per resolution ($\{x1, x2, x3, x4\}$) against models with a single parameterised block per resolution weight-shared $\{2, 3, 5, 10, 20\}$ times ($\{r2, r3, r5, r10, r20\}$). We report NLL (↓) measured on the test set, corresponding to the results on the validation set in Fig. 5. NLL performance increases with more weight-sharing repetitions and surpasses models without weight-sharing but with more parameters.

Neural architecture	# Params	NLL (↓)
`r1/x1`	107k	≤ 86.87
`r2`	107k	≤ 85.25
`r3`	107k	≤ 84.92
`r5`	107k	≤ 83.92
`r10`	107k	≤ 82.67
`r20`	107k	≤ 81.84
`x2`	140k	≤ 84.44
`x3`	173k	≤ 82.64
`x4`	206k	≤ 82.46

In Table G.6, we provide key run-specific hyperparameters for the large-scale runs corresponding to Table 1 in the main text. Two points on the architecture of the encoder and the decoder are worth noting: First, note that the decoder typically features more parameters and a larger stochastic depth than the encoder. We here follow VDVAE which observed this distribution of the parameters to be beneficial. Second, note that while we experienced a benefit of weight-sharing, there is a diminishing return of the number of times a specific cell is repeated. Hence, we typically repeat a single block for no more than 10-20 times, beyond which performance does not improve while computational cost increases linearly with the number of repetitions. Exploring how to optimally exploit the benefit of weight-sharing in HVAEs would be an interesting aspect for future work.

Table G.5: A small-scale study on parameter efficiency of HVAEs on *CIFAR10*. We compare models with with one, two, three and four parameterised blocks per resolution ($\{x1, x2, x3, x4\}$) against models with a single parameterised block per resolution weight-shared $\{2, 3, 5, 10, 20\}$ times ($\{r2, r3, r5, r10, r20\}$). We report NLL (↓) measured on the test set, corresponding to the results on the validation set in Fig. 5. NLL performance tends to increase with more weight-sharing repetitions. However, in contrast to the validation set (see Fig. 5) where this trend is evident, it is less so on the test set.

Neural architecture	# Params	NLL (↓)
`r1/x1`	8.7m	≤ 4.17
`r2`	8.7m	≤ 4.93
`r3`	8.7m	≤ 4.78
`r5`	8.7m	–
`r10`	8.7m	≤ 4.32
`r20`	8.7m	≤ 3.54
`x2`	13.0m	≤ 5.77
`x3`	17.3m	≤ 3.07
`x4`	21.6m	≤ 3.01

Table G.6: A large-scale study of parameter efficiency in HVAEs. We here provide key run-specific hyperparameters corresponding to the results reported in Table 1 in the main text. Note that the row order of our runs directly corresponds with Table 1. δ refers to gradient clipping threshold. γ refers to the gradient skipping threshold. We use the same nomenclature for number of cells (`x`) and number of repetitions for one block (`r`) as before. In addition, as in VDVAE's official code base, we use `d` to indicate average pooling, where the integer before `d` indicates the resolution on which we pool, and the integer after indicates the down-scaling factor. Further, `m` indicates interpolating, where we up-scale from a source (integer after `m`) to a target resolution (integer before `m`).

Dataset	Method	Batch size	δ	γ	Encoder Architecture	Decoder Architecture
MNIST 28×28	`WS-VDVAE` (ours)	70	-	200	`32r3,32r3,32r3,32r3,32r3,32d2,16r3,16r3,16r3,16d2,8x6,8d2,4x3,4d4,1x3`	`1x1,4m1,4x2,8m4,8x5,16m8,16r3,16r3,16r3,16r3,16r3,32m16,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3`
	`VDVAE`* (ours)	70	-	200	`32x11,32d2,16x6,16d2,8x6,8d2,4x3,4d4,1x3`	`1x1,4m1,4x2,8m4,8x5,16m8,16x10,32m16,32x21`
CIFAR10 32×32	`WS-VDVAE` (ours)	16	400	4000	`32r12,32r12,32r12,32r12,32r12,32r12,32d2,16r12,16r12,16r12,16r12,16d2,8r12,8r12,8r12,8r12,8r12,8r12,8d2,4r12,4r12,4r12,4d4,1r12,1r12,1r12`	`1r12,4m1,4r12,4r12,8m4,8r12,8r12,8r12,8r12,8r12,16m8,16r12,16r12,16r12,16r12,16r12,32m16,32r12,32r12,32r12,32r12,32r12,32r12,32r12,32r12,32r12`
	`WS-VDVAE` (ours)	16	200	2500	`32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32d2,16r3,16r3,16r3,16r3,16r3,16r3,16d2,8x6,8d2,4x3,4d4,1x3`	`1x1,4m1,4x2,8m4,8x5,16m8,16r3,16r3,16r3,16r3,16r3,16r3,16r3,16r3,16r3,16r3,32m16,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3`
	`VDVAE`* (ours)	16	200	400	`32x11,32d2,16x6,16d2,8x6,8d2,4x3,4d4,1x3`	`1x1,4m1,4x2,8m4,8x5,16m8,16x10,32m16,32x21`
ImageNet 32×32	`WS-VDVAE` (ours)	8	200	5000	`32r10,32r10,32r10,32r10,32d2,16r10,16r10,16r10,16d2,8x8,8d2,4x6,4d4,1x6`	`1x2,4m1,4x4,8m4,8x9,16m8,16r10,16r10,16r10,16r10,16r10,32m16,32r10,32r10,32r10,32r10,32r10,32r10,32r10,32r10,32r10,32r10`
	`WS-VDVAE` (ours)	8	200	5000	`32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32d2,16r6,16r6,16r6,16r6,16r6,16d2,8x8,8d2,4x6,4d4,1x6`	`1x2,4m1,4x4,8m4,8x9,16m8,16r6,16r6,16r6,16r6,16r6,16r6,16r6,16r6,16r6,16r6,16r6,32m16,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6,32r6`
	`VDVAE`* (ours)	8	200	300	`32x15,32d2,16x9,16d2,8x8,8d2,4x6,4d4,1x6`	`1x2,4m1,4x4,8m4,8x9,16m8,16x19,32m16,32x40`
CelebA 64×64	`WS-VDVAE` (ours)	4	220	3000	`64r3,64r3,64r3,64r3,64r3,64r3,64r3,64d2,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32d2,16r3,16r3,16r3,16r3,16r3,16r3,16d2,8r3,8r3,8r3,8d2,4r3,4r3,4r3,4d4,1r3,1r3,1r3`	`1r3,1r3,4m1,4r3,4r3,4r3,8m4,8r3,8r3,8r3,8r3,16m8,16r3,16r3,16r3,16r3,16r3,16r3,16r3,32m16,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,32r3,64m32,64r3,64r3,64r3,64r3,64r3,64r3,64r3,64r3`
	`VDVAE`* (ours)	4	220	3000	`64x11,64d2,32x20,32d2,16x9,16d2,8x8,8d2,4x7,4d4,1x5`	`1x2,4m1,4x3,8m4,8x7,16m8,16x15,32m16,32x31,64m32,64x12`

G.2 HVAEs secretly represent time and make use of it

In this experiment, we measure the L_2 norm of the residual state at every ResNet block in both the forward (bottom-up/encoder) and backward (top-down/decoder) model. Let x_i be the output of ResNet block i in the bottom-up model, and y_i be the input of ResNet block i in the top-down model for one batch. In the following, augmenting Fig. 6 on MNIST in the main text, we measure $\|x_i\|_2$ or $\|y_i\|_2$, respectively, over 10 batches. We also use this data to compute appropriate statistics (mean and standard deviation) which we plot. We measure the state norm in the forward and backward pass for models trained on CIFAR10 and ImageNet32 in Figs. G.5 and G.6, respectively. We note that the forward pass of the ImageNet32 has a slightly unorthodox, yet striking pattern in terms of state norm magnitude, presumably caused by an overparameterisation of the model. In summary, these findings provide further evidence that the residual state norm of VDVAEs represents time.

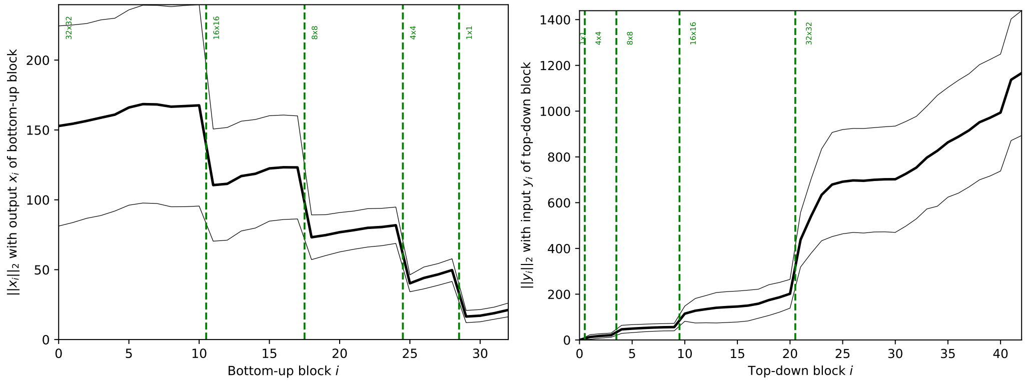


Figure G.5: HVAEs are secretly representing time *on CIFAR10*: We measure the L_2-norm of the residual state at every residual block i for the [Left] forward (bottom-up) pass, and [Right] the backward (top-down) pass, respectively, over 10 batches with 100 data points each. The thick line refers to the average and the thin, outer lines refer to ±2 standard deviations.

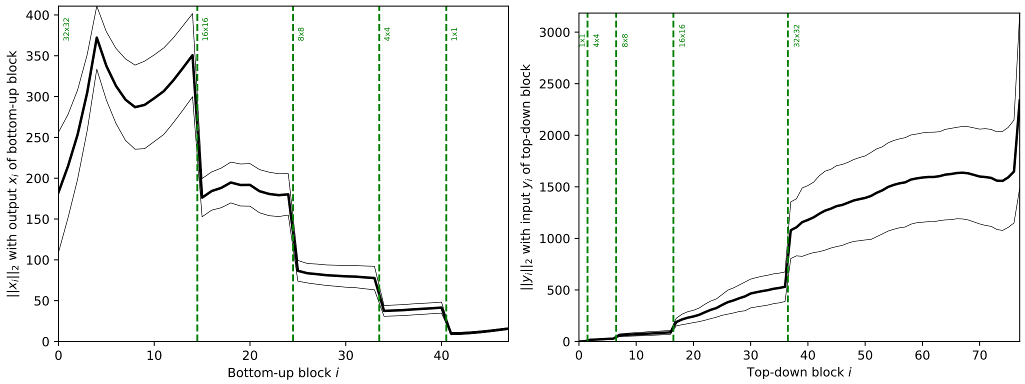


Figure G.6: HVAEs are secretly representing time *on ImageNet32*: We measure the L_2-norm of the residual state at every residual block i for the [Left] forward (bottom-up) pass, and [Right] the backward (top-down) pass, respectively, over 10 batches with 100 data points each. The thick line refers to the average and the thin, outer lines refer to ± 2 standard deviations.

When normalising the residual state in our experiments in Table 2 (case "normalised"), we do so at the same positions where we measure the state norm above. At the output of every forward ResNet block x_i and the input of every backward ResNet block y_i, we assign

$$x_i \leftarrow \frac{x_i}{\|x_i\|_2} \qquad\qquad y_i \leftarrow \frac{y_i}{\|y_i\|_2}$$

for every mini-batch during training. This results in a straight line in these plots for the "normalised" case. As the natural behavior of VDVAEs is—as we measured—to learn a non-constant norm, normalising the state norm has a deteriorating consequence, as we observe in Table 2. In contrast, the regular, unnormalised runs (case "non-normalised") show well-performing results.

We further analysed the normalised state norm experiments in Table 2. The normalised MNIST and CIFAR10 runs terminated early (indicated by ✗), more precisely after 18 hours and 4.5 days of training, respectively. From the very start of the optimisation, the normalised models have poor training behavior. To show this, in Fig. G.7, we illustrate the NLL on the validation set during training for the three normalised runs as compared to regular, non-normalised training. Validation ELBO only improves for a short time, after which the normalised runs deteriorate, showing no further improvement or even a worse NLL.

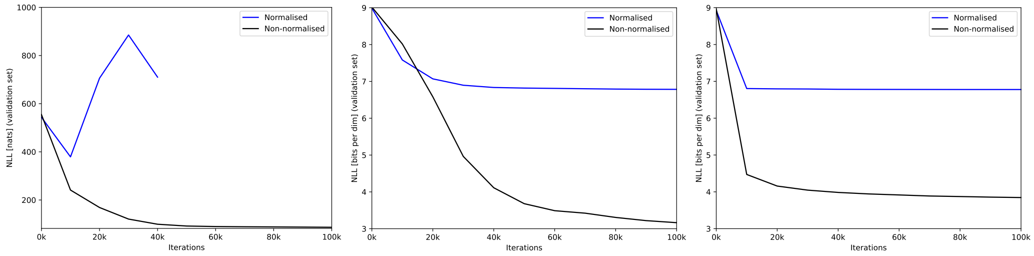


Figure G.7: On the training dynamics of VDVAE with and without a normalised residual state norm. NLL (↓) measured on the validation set of MNIST [left], CIFAR10 [middle] and ImageNet32 [right]. The normalised runs suffer from poor training dynamics from the very start of the optimisation and even terminate early on MNIST and CIFAR10, indicating that VDVAE makes use of the time representing state norm during training.

G.3 Sampling instabilities in HVAEs

When retrieving unconditional samples from our models, we scale the variances in the unconditional distributions with a temperature factor τ, as is common practice. We tune τ "by eye" to improve the fidelity of the retrieved images, yet do not cherry pick these samples. In Figs. G.8 to G.12, we provide additional, not cherry-picked unconditional samples for models trained on CIFAR10, ImageNet32, ImageNet64, MNIST and CelebA, extending those presented in Fig. 7. As shown earlier, the instabilities in VDVAE result in poor unconditional samples for CIFAR10, ImageNet32 and ImageNet64, but relatively good samples for MNIST and CelebA.

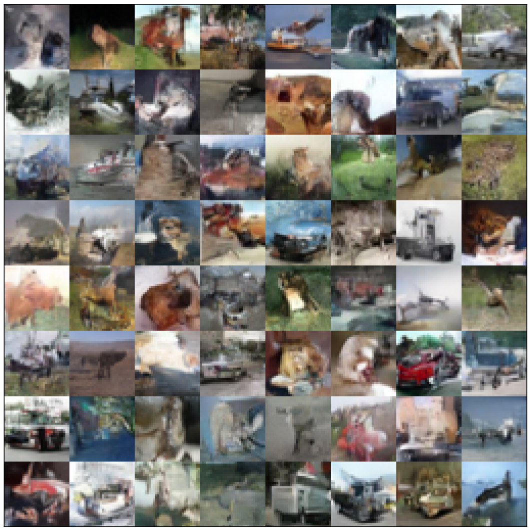


Figure G.8: Further unconditional samples (not cherry-picked) of `VDVAE`* on *CIFAR10*, augmenting those presented in Fig. 7. While samples on MNIST and CelebA demonstrate high fidelity and diversity, samples on CIFAR10, ImageNet32 and ImageNet64 are diverse, but unrecognisable, demonstrating the instabilities identified by Theorem 5. We chose the temperature as $\tau = 0.9$.

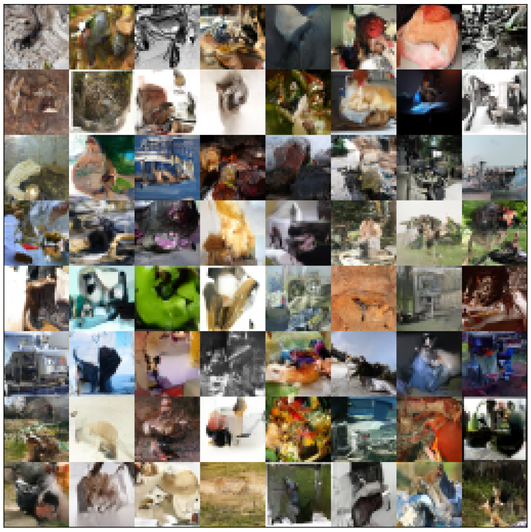


Figure G.9: Further unconditional samples (not cherry-picked) of `VDVAE`* on *ImageNet32*, augmenting those presented in Fig. 7. While samples on MNIST and CelebA demonstrate high fidelity and diversity, samples on CIFAR10, ImageNet32 and ImageNet64 are diverse, but unrecognisable, demonstrating the instabilities identified by Theorem 5. We chose the temperature as $\tau = 1.0$.

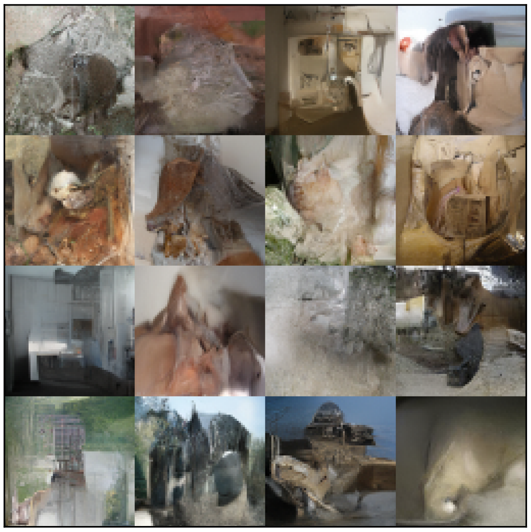


Figure G.10: Further unconditional samples (not cherry-picked) of `VDVAE`* on *ImageNet64*, augmenting those presented in Fig. 7. While samples on MNIST and CelebA demonstrate high fidelity and diversity, samples on CIFAR10, ImageNet32 and ImageNet64 are diverse, but unrecognisable, demonstrating the instabilities identified by Theorem 5. Temperatures τ are tuned for maximum fidelity. We chose the temperature as $\tau = 0.9$.

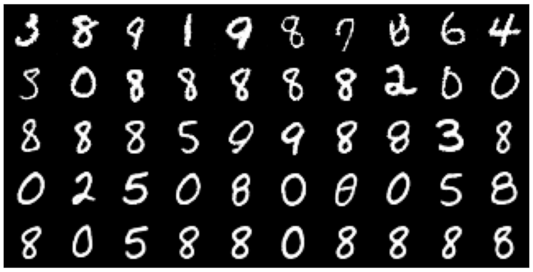


Figure G.11: Further unconditional samples (not cherry-picked) of `VDVAE`* on *MNIST*, augmenting those presented in Fig. 7. While samples on MNIST and CelebA demonstrate high fidelity and diversity, samples on CIFAR10, ImageNet32 and ImageNet64 are diverse, but unrecognisable, demonstrating the instabilities identified by Theorem 5. We chose the temperatures as $\tau \in \{1.0, 0.9, 0.8, 0.7, 0.5\}$ (corresponding to the rows).

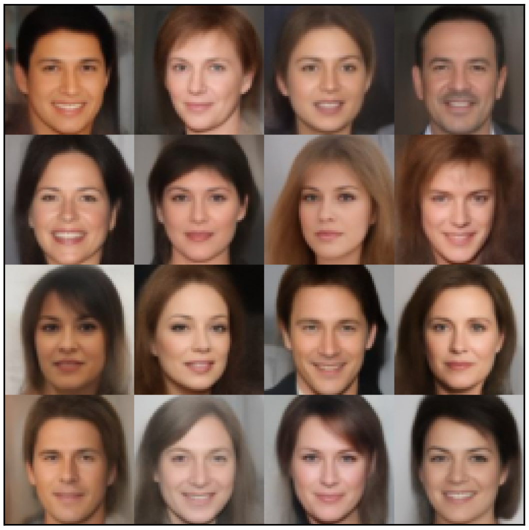


Figure G.12: Further unconditional samples (not cherry-picked) of `VDVAE`* on *CelebA*, augmenting those presented in Fig. 7. While samples on MNIST and CelebA demonstrate high fidelity and diversity, samples on CIFAR10, ImageNet32 and ImageNet64 are diverse, but unrecognisable, demonstrating the instabilities identified by Theorem 5. We chose the temperature as $\tau = 0.5$.

In addition, we here also visualise the representational advantage of HVAEs. Fig. G.13 shows samples where we gradually increase the number of samples from the posterior vs. the prior distributions in each resolution across the columns. This means that in column 1, we sample the first latent $\boldsymbol{z}_l$ *in each resolution* from the (on encoder activations conditional) posterior q, and all other latents from the prior p. A similar figure, but gradually increasing the contribution of the posterior across the blocks of all resolutions (i.e. column 1 samples $\boldsymbol{z}_l$ from the posterior in the very first resolution only) is shown in VDVAE [9, Fig. 4]. Fig. G.13

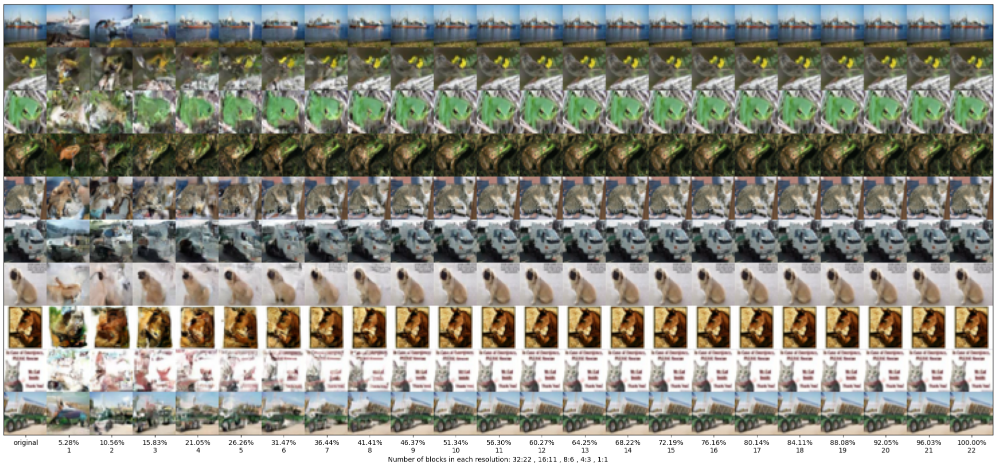


Figure G.13: Samples drawn from our model when gradually increasing the contribution of the approximate posterior. In each column with integer s, we sample the first s latent variables from the approximate posterior in each resolution, i.e. $\mathbf{z}_i \sim q(\mathbf{z}_i|\mathbf{z}_{>i})$ (up to the maximum number of latent variables in each resolution), and $\mathbf{z}_j \sim p(\mathbf{z}_j|\mathbf{z}_{>j})$ for all other latent variables. The percentage number indicates the fraction of the number of latent variables among all latent variables sampled from the approximate posterior. In the left-most column, we visualise corresponding input images.

G.4 Ablation studies

G.4.1 Number of latent variables

The number of latent variables increase when increasing the stochastic depth through weight-sharing. Thus, an important ablation study is the question whether simply increasing the number of latent variables improves HVAE performance, which may explain the weight-sharing effect. On CIFAR10, we find that this is not the case: In Table G.7, we analyse the effect of increasing the number of latent variables ceteris paribus. Furthermore, in Fig. G.14, we report validation NLL during training for the same runs. In this experiment, we realise the increase in number of latent variables by increasing the number of channels in each latent variable $\mathbf{z}_l$ exponentially while slightly decreasing the number of blocks so to keep the number of parameters roughly constant. Both results indicate that the number of latent variables, at least for this configuration on CIFAR10, do not add performance and hence cannot explain the weight-sharing performance.

Table G.7: On the effect of the number of latent variables on CIFAR10. We report the NLL on the test set at convergence.

# of latent variables	# Params	NLL ($\downarrow$)
396k	39m	2.88
792k	39m	2.88
1.584m	39m	2.87
3.168m	39m	2.88

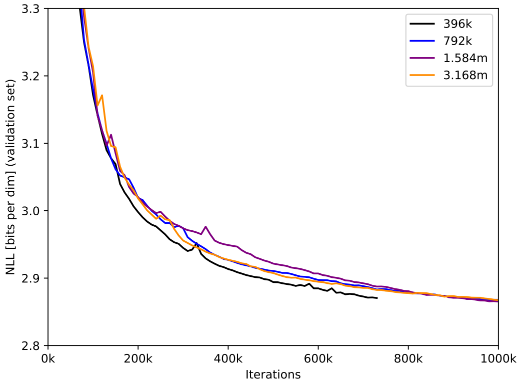


Figure G.14: On the effect of the number of latent variables. We report NLL on the validation set of CIFAR10 during training.

G.4.2 Fourier features

In this experiment, we are interested in the effect of Fourier features imposed onto a Haar wavelet basis due to the inductive bias of the U-Net. Intuitively, we would expect that Fourier features do not add performance as the U-Net already imposes a good basis for images. We now validate this hypothesis experimentally: We compute Fourier features in every ResNet block at three different locations as additional channels and varying frequencies. We implement Fourier features closely following VDM [19]: Let $h_{i,j,k}$ be an element of a hidden activation of the network, for example of a sampled latent $h = \mathbf{z}_l$, in spatial position (i, j) and channel k. Then, for each scalar $h_{i,j,k}$, we add two transformed scalars for each frequency governed by β as follows:

$$f^{\beta}_{i,j,k} = \sin\left(z_{i,j,k} 2^{\beta} \pi\right), \text{ and } g^{\beta}_{i,j,k} = \cos\left(z_{i,j,k} 2^{\beta} \pi\right).$$

In our experiments, we experiment with different choices for β, but typically select two values at a time (as in VDM), increasing the number of channels in the resulting activation by a factor of five. Fourier features are computed on and concatenated to activations at three different locations (in three separate experiments): At the input of the ResNet block, after sampling, and for the input of the two branches parameterising the posterior and prior distributions.

In Tables G.8 and G.9, we report performance when concatenating Fourier features at every ResNet block in these three locations. In all cases, Fourier features deteriorate performance in this multi-resolution wavelet basis, particularly for high-frequencies which often lead to early termination due to numeric overflows. However, if training only a single-resolution model where no basis is enforced, training does not deteriorate, not even for high-frequency Fourier features, yet performance can neither be improved. Furthermore, we experimented with computing and concatenating the Fourier features only to the input image of the model, hypothesising numerical instabilities caused by computing Fourier transforms at every ResNet block, and report results in Table G.10. Here, performance is significantly better as runs no longer deteriorate, but Fourier features still do not improve performance compared to not using Fourier features at all.

Table G.8: Fourier features introduced and concatenated in every ResNet block at three different locations on *MNIST*. VDVAE typically deteriorates or has poor performance.

Exponent β	NLL
Loc. 1	
$[1, 2]$	≤ 78.4
$[3, 4]$	≤ 80.55
$[5, 6]$ (✗)	–
Loc. 2	
$[1, 2]$	≤ 554.50
$[3, 4]$ (✗)	–
$[5, 6]$ (✗)	–
Loc. 3	
$[1, 2]$ (✗)	–
$[2, 3]$	≤ 306.67
$[3, 4]$	≤ 345.67
Loc. 1 & single-res.	
$[3, 4]$	≤ 87.55
$[5, 6]$	≤ 86.96
$[7, 8]$	≤ 91.67
No Fourier Features	≤ 79.81

Table G.9: Fourier features introduced and concatenated in every ResNet block at three different locations on *CIFAR10*. VDVAE typically deteriorates or achieves a poor performance.

Exponent β	NLL
Loc. 1	
$[3, 4]$ (✗)	–
$[5, 6]$ (✗)	–
$[7, 8]$	≤ 8.94
Loc. 2	
$[3, 4]$ (✗)	–
$[5, 6]$ (✗)	–
$[7, 8]$ (✗)	–
Loc. 3	
$[3, 4]$ (✗)	–
$[5, 6]$	≤ 8.94
$[7, 8]$	≤ 8.99
No Fourier Features	≤ 2.87

Table G.10: Fourier features introduced on the input image of the model only, with results on *CIFAR10*. While performing better than if introduced at every ResNet block, still Fourier features do not improve performance compared to using no Fourier features at all.

Exponent β	NLL
Fourier Features on input only	
$[3, 4]$	≤ 2.95
$[5, 6]$	≤ 2.96
$[7, 8]$	≤ 2.89
No Fourier Features	≤ 2.87

G.4.3 On the effect of a multi-resolution bridge.

State-of-the-art HVAEs have a U-Net architecture with pooling and, hence, are multi-resolution bridges (see Theorem 4). We investigate the effect of multiple resolutions in HVAEs (here with spatial dimensions $\{32^2, 16^2, 8^2, 4^2, 1^2\}$) against a single resolution (here with spatial dimension 32^2). We choose the number of blocks for the single resolution model such that they are distributed in the encoder and decoder proportionally to the multi-resolution model and the total number of parameters are equal in both, ensuring a fair comparison. As we show in Table G.11, the multi-resolution models perform slightly better than their single-resolution counterparts, yet we would have expected this difference to be more pronounced. We also note that it may be worth measuring other metrics for instance on fidelity, such as the FID score [71]. Additionally, multi-resolution models have a representational advantage due to their Haar wavelet basis representation (illustrated in Appendix G.4, Fig. G.13).

Table G.11: Single- vs. multi-resolution HVAEs.

# Resolutions	# Params	NLL
MNIST		
Single	328k	≤ 81.40
Multiple	339k	≤ 80.14
CIFAR10		
Single	39m	≤ 2.89
Multiple	39m	≤ 2.87
ImageNet32		
Single	119m	≤ 3.68
Multiple	119m	≤ 3.67

G.4.4 On the importance of a stochastic differential equation structure in HVAEs

A key component of recent HVAEs is a residual cell, as outlined in §4. The residual connection makes HVAEs discretise an underlying SDE, as we outlined in this work. Experimentally, it was previously noted as being crucial for stability of very deep HVAEs. Here, we are interested in ablating the importance of imposing an SDE structure into HVAEs: We compare models with a residual HVAE cell (as in VDVAE) with a non-residual HVAE cell which is as close to VDVAE as possible to ensure a fair comparison. The non-residual VDVAE cell does not possess a residual state which flows through the backbone architecture. We achieve this by removing the connection between the first and second element-wise addition in VDVAE's cell (see [9, Fig. 3]), which is equivalent to setting $Z_{i,+} = 0$. Hence, in the non-residual cell, during training and evaluation, the reparameterised sample is directly taken forward. Note that this is distinct from the Euler-Maruyama cell which features a residual connection. Our experiments confirm that a *residual* cell is key for training stability, as illustrated in Table G.12 and Fig. G.15: Without a residual state flowing through the decoder, models quickly experience posterior collapse of the majority of layers during training.

G.4.5 Synchronous vs. asynchronous processing in time

During the bottom-up pass, VDVAE takes forward the activation of the last time step in each resolution which is passed to every time step in the top-down pass on the same resolution (see Fig. 3 in VDVAE [9]). In this ablation study, we were interested in this slightly peculiar choice of an *asynchronous* forward and backward process and to what degree it is important for performance. We thus compare an asynchronous model, with skip connections as in VDVAE [9], with a *synchronous* model, where activations from the bottom-up pass are taken forward to the corresponding time step in the top-down pass. In other words, in the synchronous case, the skip connection mapping between time steps in the encoder and decoder is 'bijective', and it is not 'injective', but 'surjective' in the asynchronuous case. We realise the synchronous case by choosing the same number of blocks in the encoder as `VDVAE`* has in the decoder, i.e. constructing a 'symmetric' model. To ensure a fair comparison, both models (synchronous and asynchronous) are further constructed to have the same number of parameters. In Table G.13, we find that synchronous and asynchronous processing achieve comparable NLL, indicating that the asynchronous design is not an important contributor to performance in VDVAE. We

Table G.12: Residual vs. non-residual VDVAE cell. The residual HVAE strongly outperforms a non-residual VDVAE cell, where the latter's training deteriorates. This is also analysed in Fig. G.13. We report NLL on the test set at convergence, or at the last model checkpoint before deterioration of training.

Cell type	NLL
MNIST	
Residual VDVAE cell	≤ 80.05
Non-residual VDVAE cell	≤ 112.58
CIFAR10	
Residual VDVAE cell	≤ 2.87
Non-residual VDVAE cell	≤ 3.66
ImageNet	
Residual VDVAE cell	≤ 3.667
Non-residual VDVAE cell (✗)	≤ 4.608

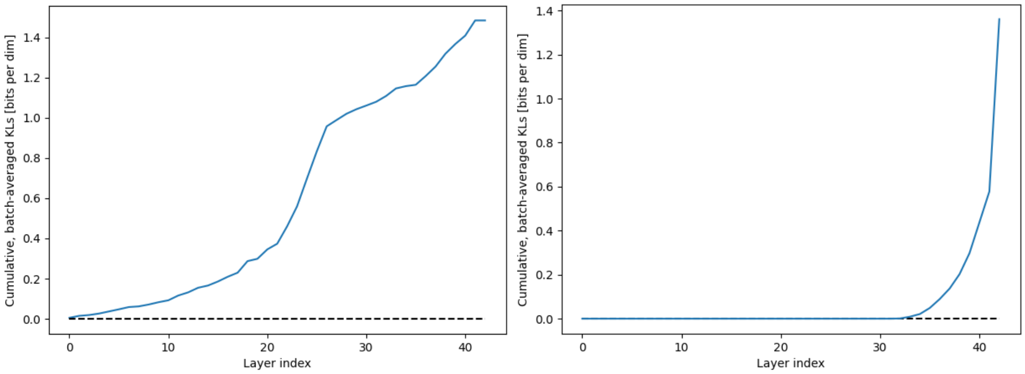


Figure G.15: Cumulative sum of KL-terms in the ELBO of a residual and non-residual VDVAE, averaged over a batch at convergence. We report the two CIFAR10 runs in Table G.12. The posterior collapses for the majority of the latent variables in the non-residual VDVAE cell case [right], but carries information for all latent variables in the regular, residual cell case [left].

note, however, that an advantage of the asynchronous design, which is exploited by VDVAE, is that the bottom-up and top-down architectures can have different capacities, i.e. have a different number of ResNet blocks. VDVAE found that a more powerful decoder was beneficial for performance [9].

Table G.13: Synchronous vs. asynchronous processing in time. We report NLL on the test set on CIFAR10 and ImageNet32, respectively.

Processing	NLL
CIFAR10	
Synchronous	≤ 2.85
Asynchronous	≤ 2.86
ImageNet32	
Synchronous	≤ 3.69
Asynchronous	≤ 3.69