GEOMETRY-AWARE ROBUST DENOISING VIA HIERARCHICAL MANIFOLD TRAVERSAL

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing manifold traversal algorithms denoise by navigating a graph via gradient-guided steps and zero-order shortcuts. Because they operate at a single resolution, single-layer traversal is susceptible to local minima traps and requires fixed initialization to succeed, making it impractical. We introduce *Hierarchical Manifold Traversal Networks* (HMTNs), which approximate a d-dimensional manifold with a stack of traversal graphs at geometrically increasing resolutions. A cross-layer map provides a strong initialization for each finer layer from the converged coarse layer. Importantly, in HMTN the hierarchy depth grows only logarithmically in the number of training samples N and depends on the intrinsic dimension d rather than the ambient dimension D. Our empirical results on time-series gravitational-wave manifolds show that, in the random-initialization regime, HMTN achieves a $134\times$ improvement in MSE relative to single-layer traversal at only $3.9\times$ the multiplication cost. On patch denoising, gains range from $1.6\times$ on MNIST, which has the lowest scale separation, to 6–$9\times$ on CIFAR-10 and CelebA, whose scale separation is substantially larger. HMTN also offers a favorable accuracy–compute trade-off against classical search-based and learning-based alternatives: it requires a median of $5.7\times$ fewer multiplications than exhaustive nearest-neighbor search, and nonlinear autoencoders need $25\times$ more multiplications to match its accuracy. Training HMTN is a one-time CPU cost of 9-32 minutes on our gravitational-wave benchmarks.

1 INTRODUCTION

A key structural assumption in machine learning is that data, including natural images, medical scans, and scientific waveforms, lie on or near a low-dimensional submanifold of their high-dimensional ambient space. In practice, observed data are contaminated with noise, making denoising a fundamental problem in signal and image processing. Manifold denoising aims to recover a clean signal that lies on a d-dimensional submanifold $\mathcal{M} \subset \mathbb{R}^D$ from a noisy observation.

Denoising approaches such as naive nearest-neighbor search achieve good accuracy but scale exponentially with the dimensionality d of the manifold (Vershynin, 2018; Wainwright, 2019). Diffusion models synthesize high-fidelity samples by iteratively applying a learned denoiser (Ho et al., 2020). Compressed sensing algorithms build upon similar iterative procedures to reconstruct medical and scientific signals from incomplete measurements (Candès et al., 2006; Donoho, 2006). These achieve impressive empirical performance; however, they are computationally demanding, requiring costly denoiser evaluations. Reducing test-time cost without sacrificing accuracy is a central challenge in deploying these methods at scale.

Recent work on manifold traversal (Wang et al., 2025) bridges this gap by formulating manifold traversal as a Riemannian optimization problem over a learned graph approximation of a priori unknown manifold $\mathcal{M}$. Specifically, in an online learning setting, each new training point either seeds a new node, updates an existing node's local tangent model to create a first-order edge, or adds a zero-order edge (tunnel) to escape

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

local minima, all without having to revisit prior data. In theory, this test-time efficient family of denoisers achieves near-optimal denoising error at substantially reduced cost. This analysis, however, assumes a high-quality zero-order edge structure affording global coverage of the manifold, an assumption that is expensive to certify and difficult to satisfy without sufficiently dense, well-distributed data. As a result, *single-layer traversal requires fixed initialization to succeed*, a very limiting requirement in practical settings. The single-resolution graph is not suitable for navigating manifolds that span many scales, where there exists a high density of fine-grained edges for global traversal.

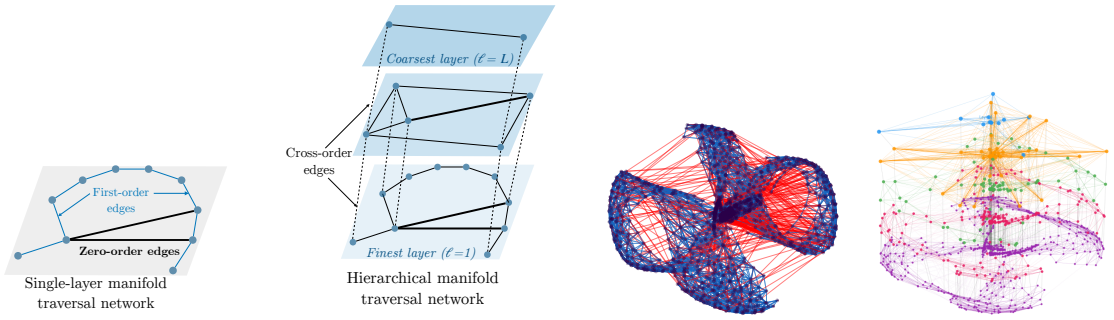


Figure 1: **Hierarchical Manifold Traversal Network (HMTN). (a)** Traditional single-layer traversal network. **(b)** Proposed hierarchical manifold traversal network (HMTN). **(c)** PCA visualization of a single-layer network. Blue points represent landmarks, blue lines denote first-order edges, and red lines denote zero-order edges. **(d)** HMTN on a gravitational waves dataset (Nitz et al., 2024).

In this work, we introduce *Hierarchical Manifold Traversal Networks (HMTNs)*, a multi-layer framework that addresses these limitations through a coarse-to-fine graph structure. HMTN maintains a data-adaptive hierarchy of networks with strictly decreasing coverage radii to facilitate efficient navigation. At test-time, a noisy data point is processed starting from the coarsest layer. Optimization then descends to subsequent finer layers via a *cross-layer map* which seeds traversal in the next layer at the nearest node. Each layer thus inherits a warm start from the layer above, confining expensive fine-grained computation to a small, correctly identified region of $\mathcal{M}$. The depth L of the hierarchy grows logarithmically in the number of training samples N, akin to the scaling of standard hierarchical search structures like Hierarchical Navigable Small World (HNSW) networks (Malkov and Yashunin, 2018). To summarize, we make the following contributions:

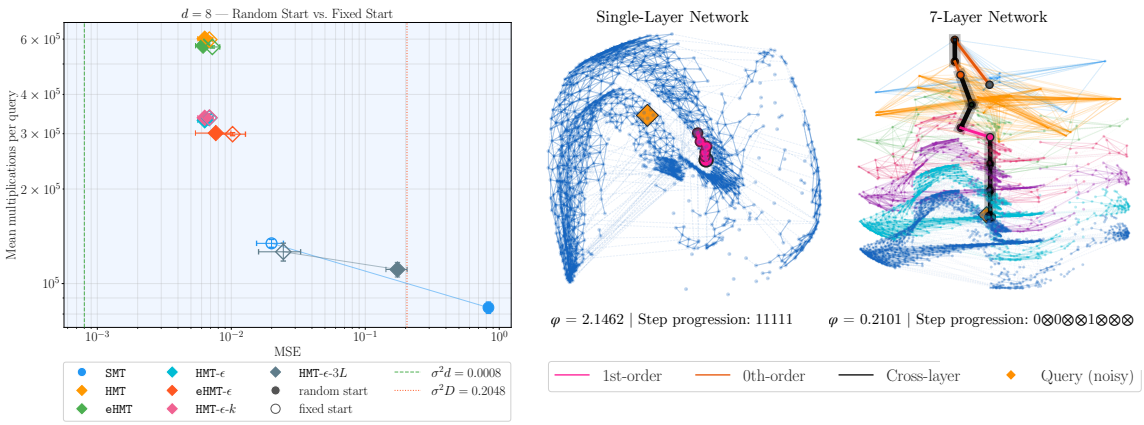
(i) **Hierarchical manifold traversal.** We introduce a novel coarse-to-fine multi-layer graph traversal algorithm for manifold denoising, with hierarchy depth L chosen to scale logarithmically with number of training samples N based on intrinsic manifold geometry.

(ii) **Robustness and accuracy from coarse-to-fine localization.** Geometry-aware coarse-to-fine navigation allows HMTN to bypass suboptimal local minima traps where single-layer methods get stuck. On 2048-dimensional gravitational wave data with intrinsic dimensions $d = 2$ and $d = 8$, HMTN demonstrates $\mathbf{134}\times$ gains in accuracy at $3.9\times$ the cost of single-layer baselines.

(iii) **Revealing multiscale structure.** HMTN reveals the multiscale structure of data manifolds through the ratio R_L/R_1 between its coarsest and finest radii, computed directly from the training data before the graph is built. This scale separation measures the range of scales the data spans, sets the depth of the hierarchy, and anticipates how much the hierarchy can gain over single-layer traversal. We demonstrate this on gravitational waves, MNIST, CIFAR-10, and CelebA patch datasets.

2 Related Work

Denoising has been established as a core task in image and signal processing (Buades et al., 2005; Tomasi and Manduchi, 1998; Tukey et al., 1977). Early work relied on smoothness or sparsity priors to recover



(a) **(b)** **(c)**

Figure 2: **(a) Complexity vs. MSE of Hierarchical Manifold Traversal (HMT).** `HMT` and all its variants are robust to initialization. `SMT` diverges when initialized randomly (to the right of the red dashed line). **(b) - (c) Traversal Paths Comparison.** Traversal paths of `SMT` and `HMT` on the same noisy query point. `SMT` converges quickly but terminates at a local minimum far from the query, while `HMT` escapes local minima. Gravitational wave data with parameters from a Gaussian distribution, $D = 2048$, $d = 8$, $p = 1\%$, $\rho = 0.1$.

clean signals. Sparse models represent data as a composition of simple bases, such as wavelets (Donoho and Johnstone, 1994; Chang et al., 2000), learned dictionaries (Aharon et al., 2006; Elad and Aharon, 2006), or overcomplete representations that enable robust recovery from corruption (Wright et al., 2008). Iterative shrinkage and thresholding algorithms operationalize these priors through repeated denoising steps (Beck and Teboulle, 2009). A complementary line of work (Buades et al., 2005; Dabov et al., 2007) exploits non-local self-similarity across image patches. Modern approaches interpret denoising as a proximal operation within larger optimization procedures (Xiang et al., 2021), giving rise to plug-and-play methods (Venkatakrishnan et al., 2013) and regularization-by-denoising frameworks (Romano et al., 2017).

Across different application domains in machine learning, denoising is implicitly utilized for signal construction and generation. Diffusion models (Ho et al., 2020; Song and Ermon, 2019) with functional forms realized with neural networks like ResNet (Wang et al., 2023), U-Net (Rombach et al., 2022), and transformers (Peebles and Xie, 2023) encode complex structural information about the data. Denoising methods serve as priors for solving inverse problems such as image superresolution (Zhang et al., 2021), deblurring (Kindermann et al., 2005), and even semantic matching (Hedlin et al., 2023). Besides diffusion models, generative techniques such as normalizing flows-based denoising approaches learn a low-dimensional data manifold for image generation (Horvat and Pfister, 2021; Bhattacharyya et al., 2020).

Early works on manifold denoising leverage the graph Laplacian with k-nearest neighbors (k-NN) (Hein and Maier, 2006), or use a locally linear denoising framework (Gong et al., 2010; Shi et al., 2005). All these methods incur high computational overhead due to an exhaustive sweep over the dataset. These costs motivate the use of approximate nearest neighbor methods, which trade exact search for tractable runtime. The concept of k-approximate nearest neighbors (ANN) allows for a bounded number of errors in exchange for sublinear search time (Indyk and Motwani, 1998). A prominent ANN method is Locality-Sensitive Hashing (Indyk and Motwani, 1998; Datar et al., 2004), which improves search efficiency by partitioning the space so that similar items share hash values, thereby narrowing the search to a small set of candidate pairs. Navigable Small World (NSW) models (Kleinberg, 2000; Watts and Strogatz, 1998) optimize search by exploiting the "small-world" property, using greedy routing to navigate toward targets via short path lengths and local connectivity. Building on this, Hierarchical Navigable Small Worlds (HNSW) (Malkov and

Yashunin, 2018) organize nearest-neighbor graphs into layers based on distance scales, enabling a coarse-to-fine search that significantly accelerates neighbor discovery in high-dimensional spaces.

Rooted in Riemannian optimization, recent work (Wang et al., 2025) approximates the manifold locally using affine fits during traversal, enabling efficient test-time denoising. However, the approach assumes global coverage of the manifold, i.e., that the graph, once built, reaches all relevant regions. Since the graph is constructed incrementally, this assumption is rarely satisfied, leaving underrepresented and particularly challenging regions of the manifold poorly served. To address this, we build a coarse-to-fine hierarchical network with graphs at multiple resolutions. Our hierarchical structure enables broad manifold coverage while maintaining robustness to accuracy and efficient test-time cost.

3 Problem Formulation

A *traversal graph* $\mathcal{G}$ is a geometric graph defined on a set of vertices $V = \{1, \ldots, M\}$, where each vertex i represents a landmark $\boldsymbol{q}_i \in \mathbb{R}^D$ approximating a point on an d-dimensional submanifold $\mathcal{M} \subset \mathbb{R}^D$. Formally, $\mathcal{G}$ consists of landmarks, tangent spaces, edge embeddings, first-order edges, and zero-order edges (*cf.* Section 4.1). Rather than scanning an entire dataset exhaustively with worst-case complexity of at least $\mathcal{O}(D \cdot e^{cd})$, as nearest-neighbor methods do, the traversal graph encodes the manifold's geometry discretely through landmarks, tangent spaces, and edge embeddings, enabling efficient test-time optimization. The graph is constructed online, growing one sample at a time via an incremental learning algorithm, which ensures test-time efficiency and adapts to high-dimensional datasets.

Our goal is to denoise data sampled from a d-dimensional submanifold $\mathcal{M} \subset \mathbb{R}^D$ by solving,

$$\min_{\boldsymbol{q} \in \mathcal{M}} \varphi_{\boldsymbol{y}}(\boldsymbol{q}) \equiv \tfrac{1}{2}\|\boldsymbol{q} - \boldsymbol{y}\|_2^2. \tag{1}$$

We learn a hierarchical traversal graph by observing clean training samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^D$, with signal $\boldsymbol{x}_i \sim \boldsymbol{\mu}$, a distribution supported on $\mathcal{M}$. At test time, a noisy input $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{z}$, $\boldsymbol{z} \sim_{\text{iid}} \mathcal{N}(0, \sigma^2)$, navigates the graph via a mixed-order traversal: *first-order* steps move efficiently along the manifold's local geometry by selecting the neighbor whose edge embedding is best aligned with the negative Riemannian gradient, while *zero-order* steps jump over local minimizers by selecting the zero-order neighbor with the smallest objective value. Finally, *cross-layer* edges facilitate cross-layer hops.

4 Methodology

In this section, we describe a learned hierarchical denoiser we call Hierarchical Manifold Traversal Networks (HMTNs), which yield *gains in accuracy while staying robust to initialization*. We first introduce background (Section 4.1), then define the HMTN method (Section 4.2) and its graph construction (Section 4.3), and finally, elaborate on the test-time denoising process of our proposed framework (Section 4.4).

4.1 Background and Single-Layer Traversal Networks

Building upon the learned mixed-order optimizer, akin to traversal networks in (Wang et al., 2025), we propose an online learning framework for HMTNs. Our approach constructs a hierarchical graph, with each layer of the hierarchy serving as an approximation to the manifold $\mathcal{M}$ at different resolutions. The graph construction process is shown in Algorithm 1, with detailed description of each line provided in Section I.

We denote HMTN as $\{\mathcal{G}_l\}_{l=1}^L = \{\mathcal{Q}_l, \mathcal{E}_l^1, \mathcal{E}_l^0, \mathcal{E}_l^\otimes\}_{l=1}^L$, a directed hierarchical graph that consists of L layers, with each layer l containing sets of landmarks $\mathcal{Q}_l$, first-order edges $\mathcal{E}_l^1$, zero-order edges $\mathcal{E}_l^0$, and cross-layer edges $\mathcal{E}_l^\otimes$. Here, layers $l = 1$ and $l = L$ correspond to the finest and the coarsest layers, respectively. Next, we define each component and provide intuition.

Landmarks. Landmarks are a collection of clean training points in $\mathbb{R}^D$ that serve as a discrete approximation of the manifold $\mathcal{M}$ at a given layer l. In a given layer l, we find a set of k_l landmarks

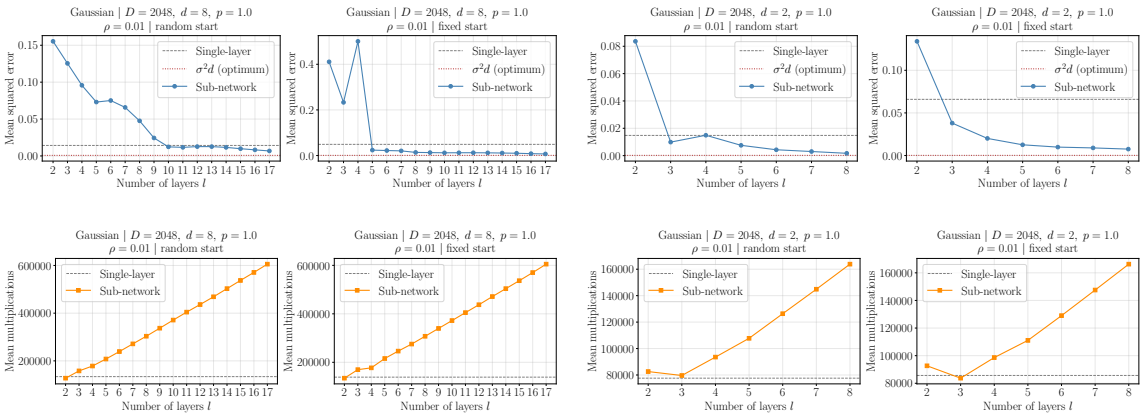


Figure 3: **Progressive Layer Ablation.** Denoising error and mean multiplications per query as a function of layers l, evaluated on gravitational-wave signals with $d = \{2, 8\}$, $p = 1$, $\rho = 0.01$, under random and fixed initialization. Sub-networks use layers from a trained L-layer model, adding intermediate layers progressively between the coarsest and finest. Error decreases monotonically with l while cost grows modestly, demonstrating a favorable accuracy-complexity trade-off.

$\mathcal{Q}_l = \{\boldsymbol{q}_i^{(l)},\ i = 1, \ldots, k_l \mid \boldsymbol{q}_i^{(l)} \in \mathbb{R}^D\}$, corresponding to a set of vertices $V_l = \{v_i^{(l)},\ i = 1, \ldots, k_l\}$ in the constructed hierarchical graph. Each landmark is a local representative of the manifold, equipped with a local tangent space $T_i^{(l)} = \text{span}(\boldsymbol{U}_i^{(l)})$ with $\boldsymbol{U}_i^{(l)} \in \mathbb{R}^{D \times d}$, $d = \dim\mathcal{M}$. Additionally, each landmark $\boldsymbol{q}_i^{(l)}$ consists of first-order edges $\mathcal{E}_{l,i}^1$, zero-order edges $\mathcal{E}_{l,i}^0$, and cross-layer edges $\mathcal{E}_{l,i}^{\otimes}$.

First-order edges. While landmarks represent a discrete approximation, first-order edges $\mathcal{E}^1$ provide information on the *local* geometry of the manifold $\mathcal{M}$. These are bidirectional edges that connect landmarks within a radius R_{nbrs} of each other. A first-order edge is denoted by " $\to_1$ ", and $\boldsymbol{q}_i \to_1 \boldsymbol{q}_j$ indicates that landmarks $\boldsymbol{q}_i$ and $\boldsymbol{q}_j$ are connected via a first-order edge. As this is a direct connection in ambient space $\mathbb{R}^D$, we also create *embeddings* of it as the projection of the edge onto tangent spaces T_i and T_j at $\boldsymbol{q}_i$ and $\boldsymbol{q}_j$, and denote it by $\xi_{i\to j} = \boldsymbol{U}_i^T(\boldsymbol{q}_j - \boldsymbol{q}_i)$, and $\xi_{j\to i} = \boldsymbol{U}_j^T(\boldsymbol{q}_i - \boldsymbol{q}_j)$, respectively[1]. Edge embeddings ξ are d-dimensional and lie in tangent spaces, and therefore are an efficient way to traverse across the manifold.

Zero-order edges. While the first-order edges described above provide an efficient way to traverse the proposed graph, they can be susceptible to getting stuck at local minima, as traversal is guided via Riemannian gradient optimization. To address this, (Wang et al., 2025) construct a set of zero-order edges $\mathcal{E}^0$ that are based on a lower objective value. We use the notation " $\to_0$ " to denote that two landmarks are connected via a zero-order edge. Compared to first-order edges, taking a zero-order step during traversal is more expensive, as it requires D multiplications to compute the objective value. Specifically, a zero-order step at landmark $\boldsymbol{q}_i$ costs $O(D \deg_i^0)$, where $\deg_i^0$ is the number of zero-order edges at landmark $\boldsymbol{q}_i$.

4.2 Hierarchical Manifold Traversal Networks (HMTNs)

Cross-layer edges. The proposed HMTN is comprised of L approximations of manifold $\mathcal{M}$, each at a different resolution. Each approximation is a result of forming a traditional traversal network with a different covering radius parameter R_l, from coarse to fine: $R_L > R_{L-1} > \cdots > R_1$, with L being data-dependent (*c.f.* Section 4.3), unlike prior work where the number of layers in the hierarchy is based on an arbitrary heuristic. We construct these individual traversal networks to form a tower of graphs – the HMTN itself,

[1]We use notation T_i instead of $T_{\boldsymbol{q}_i}$ for the tangent space at landmark $\boldsymbol{q}_i$, and ξ_i instead of ξ_{v_i} for an edge embedding at vertex v_i for simplicity.

Algorithm 1 `GraphConstruction`$(\mathcal{G}^{(l)}, \boldsymbol{x}, R_{\text{nbrs}})$

Input: Networks $\{\mathcal{G}^{(l)}\}_{l=1}^{L}$ with radii $R_1 < \cdots < R_L$, clean sample $\boldsymbol{x} \in \mathbb{R}^D$, radius R_{nbrs}
for $l = 1$ **to** L **do**
  $i \leftarrow$ `HierarchicalTraversal`$(\mathcal{G}_l, \boldsymbol{x})$
  if $\|\boldsymbol{q}_i - \boldsymbol{x}\| \le R_l$ **then** ▷ If clean point $\boldsymbol{x}$ is within R_l of landmark $\boldsymbol{q}_i$, update the local model
    `UpdateLocalModel` of $\boldsymbol{U}_i^{(l)}, \Xi_i^{(l)}$ at $\boldsymbol{q}_i$ with $\boldsymbol{x}$
  else
    $i_\star \leftarrow \arg\min_j \|\boldsymbol{q}_j - \boldsymbol{x}\|$ ▷ If clean point $\boldsymbol{x}$ is not within R_l distance of landmark $\boldsymbol{q}_i$, resort to a linear scan
    if $\|\boldsymbol{q}_{i_\star} - \boldsymbol{x}\| \le R_l$ **then** ▷ If the linear scan found a landmark such that the point $\boldsymbol{x}$ is within R_l of that landmark, then
      $\boldsymbol{q}_i^{(l)}$ is a local min, add $i \xrightarrow{0} i_\star$ ▷ Add a zero-order edge
      `UpdateLocalModel` (i.e. $\boldsymbol{U}_{i_\star}^{(l)}, \Xi_{i_\star}^{(l)}$) at $\boldsymbol{q}_{i_\star}^{(l)}$ with $\boldsymbol{x}$
    else
      `InitLandmark` Create a new landmark $\boldsymbol{q}_M \leftarrow \boldsymbol{x}$
      Connect $\boldsymbol{q}_M$ to neighboring landmarks within R_{nbrs}, initialize $\boldsymbol{U}_M^{(l)}, \Xi_M^{(l)}$
    end if
  end if
end for
Output: $\{\mathcal{G}_l\}_{l=1}^{L} = (V^{(l)}, \mathcal{E}^{(l)})$ ▷ The graph is a collection of vertices V and edges $\mathcal{E}$.

Algorithm 2 `HierarchicalTraversal`$(\mathcal{G}^{(l)}, \boldsymbol{y})$

Input: Noisy test sample $\boldsymbol{y} \in \mathbb{R}^D$, $\{\mathcal{G}^{(l)}\}_{l=1}^{L}$, $l \leftarrow L$, initialize $i \leftarrow 1$ (fixed initialization) or uniformly at random.
$\varphi \leftarrow \|\boldsymbol{q}_i^{(L)} - \boldsymbol{y}\|^2$.
while not globally converged **do**
  while not layer-l converged **do**
    $\mathbf{g} \leftarrow \mathbf{U}_i^{(l)*}(\boldsymbol{q}_i^{(l)} - \boldsymbol{y})$
    $i^\sharp \leftarrow \arg\max_{j:\, i \xrightarrow{1} j} \langle -\mathbf{g}, \boldsymbol{\xi}_{ij}^{(l)} \rangle$
    if $\|\boldsymbol{q}_{i^\sharp}^{(l)} - \boldsymbol{y}\|^2 < \varphi$ **then**
      $i \leftarrow i^\sharp, \quad \varphi \leftarrow \|\boldsymbol{q}_i^{(l)} - \boldsymbol{y}\|^2$
    else
      $i^\flat \leftarrow \arg\min_{j:\, i \xrightarrow{0} j} \|\boldsymbol{q}_j^{(l)} - \boldsymbol{y}\|^2$
      if $\|\boldsymbol{q}_{i^\flat}^{(l)} - \boldsymbol{y}\|^2 < \varphi$ **then**
        $i \leftarrow i^\flat, \quad \varphi \leftarrow \|\boldsymbol{q}_i^{(l)} - \boldsymbol{y}\|^2$
      else
        layer-l converged
      end if
    end if
  end while
  if pointer $\pi^{(l)}(i)$ exists **then**
    $i \leftarrow \pi^{(l)}(i), \quad l \leftarrow l - 1, \quad \varphi \leftarrow \|\boldsymbol{q}_i^{(l)} - \boldsymbol{y}\|^2$
  else
    globally converged
  end if
end while
Output: i

illustrated in Figure 1b and described in Algorithm 1 and Section 4.3. To traverse from top-to-bottom, we connect consecutive layers to one another via cross-layer edges $\mathcal{E}^{\otimes}$, described in Algorithm 3 of the Appendix.

Given the trained networks $\{\mathcal{G}_l\}_{l=1}^{L}$, we construct a set of cross-layer edge maps $\pi^{(l)} : V_l \to V_{l-1}$ for each consecutive pair of layers $l = L, \ldots, 2$. Formally, the cross-layer pointer map $\pi^{(l)} : V_l \to V_{l-1}$ is defined as

$$\pi^{(l)}(i) = \arg\min_{j \in \mathcal{C}_i^{(l)}} \left\| \boldsymbol{q}_i^{(l)} - \boldsymbol{q}_j^{(l-1)} \right\|_2^2, \tag{2}$$

where $\mathcal{C}_i^{(l)} \subseteq V_{l-1}$ denotes the set of candidates within some radius $R_\otimes$ of $\boldsymbol{q}_i^{(l)}$, defaulting to the singleton nearest neighbor when no candidate exists. At query time, upon converging to landmark i in layer l, traversal jumps unconditionally to the unique landmark in layer $l-1$ indexed by $\pi^{(l)}(i)$.

For each landmark $\boldsymbol{q}_i^{(l)} \in \mathcal{Q}_l$, we identify the set of candidate landmarks $\mathcal{C}_i^{(l)} = \{j \in V_{l-1} : \|\boldsymbol{q}_i^{(l)} - \boldsymbol{q}_j^{(l-1)}\|_2 \le R_\otimes\}$ in the adjacent finer layer that lie within a fixed radius $R_\otimes$.[2]. In practice, if no such landmark exists, we fall back to the single nearest neighbor in $\mathcal{Q}_{l-1}$ to guarantee connectivity. In the cases where there is more than one candidate landmark, we prune the set $\mathcal{C}_i^{(l)}$ down to a single nearest target and add the directed cross-layer edge $i \xrightarrow{\otimes} \pi(i)$. The result is a collection of deterministic maps $\{\pi^{(l)}\}_{l=2}^{L}$, each routing every coarse-layer node to its nearest counterpart in the following finer layer.

4.3 Hierarchical Graph Construction

The HMTN is constructed in an online fashion described in Algorithm 1. For each clean sample $\boldsymbol{x}$ and each layer $l = 1, \ldots, L$, the graph is grown incrementally as follows. We first run `HierarchicalTraversal`

[2]In ambient space, all layers are centered around the origin, and we measure distances between landmarks in two consecutive layers via $R_\otimes$. For illustrative purposes only, the layers are vertically separated in Figures 1b and 1d.

on $\mathcal{G}^{(l)}$ to locate the nearest landmark $\boldsymbol{q}_i^{(l)}$. If $\boldsymbol{q}_i^{(l)}$ lies within the layer's covering radius R_l, the sample is assigned to that landmark and its local tangent basis $\boldsymbol{U}_i^{(l)}$ and embedding $\Xi_i^{(l)}$ are updated via `UpdateLocalModel`, which uses incremental SVD for efficient and scalable updates. Otherwise, the traversal has failed to reach a sufficiently close landmark, and we inspect the global nearest neighbor $i_\star = \arg\min_j \|\boldsymbol{q}_j^{(l)} - \boldsymbol{x}\|$, exhaustively. If $\boldsymbol{q}_{i_\star}^{(l)}$ is within R_l, then $\boldsymbol{q}_i^{(l)}$ is a local minimum of the traversal graph; we resolve this by adding a zero-order escape edge $i \xrightarrow{0} i_\star$ and assigning the sample to $i_\star$. If no existing landmark is within R_l, the sample lies in an uncovered region of parameter space, and a new landmark is created at $\boldsymbol{q}_M \leftarrow \boldsymbol{x}$ via `InitLandmark`, which connects $\boldsymbol{q}_M$ to all existing landmarks within R_{nbrs}, and initializes with its own local parameters $\boldsymbol{U}_M^{(l)}, \Xi_M^{(l)}$, establishes $\mathcal{E}^l$. The procedure is repeated for all layers, yielding a collection of layered graphs $\{\mathcal{G}_l\}_{l=1}^L = \{\mathcal{Q}_l, \mathcal{E}_l^1, \mathcal{E}_l^0, \mathcal{E}_l^\otimes\}_{l=1}^L$ with landmarks having their associated local parameters $\boldsymbol{U}_M^{(l)}$ that span the local tangent space and $\Xi_M^{(l)}$ first-order edge embeddings.

Layer count. Algorithm 1, called `GraphConstruction`, with each line detailed in Appendix Section I, builds a tower of L graphs $\{\mathcal{G}^{(l)}\}_{l=1}^L$ with geometrically spaced landmark nets of radii $R_1 < \cdots < R_L$ and cross-layer maps $\pi^{(l)} : V_l \to V_{l-1}$, formally defined in Definition 2. We set the finest spacing $R_1 = (\text{vol}(\mathcal{M})/N)^{1/d}$ to match the average inter-sample distance, and the coarsest spacing $R_L = \min\{\text{diam}(\mathcal{M}), \tau_\mathcal{M}\}$, with $\tau_\mathcal{M}$ as the manifold reach, to remain within the regime where the manifold can be faithfully approximated locally (Boumal, 2023). We set the inter-layer ratio to be $\lambda = e^{-1/d}$, inspired by (Kleinberg, 2000; Malkov and Yashunin, 2018; Watts and Strogatz, 1998), so that adjacent layers differ in landmark density by exactly a factor of e, yielding a geometrically uniform spacing in log-density. Under these choices, the maximum number of layers required for the hierarchy satisfies:

$$L \;\le\; 1 + \ln\left(\frac{N\left(\min\{\text{diam}(\mathcal{M}),\, \tau_\mathcal{M}\}\right)^d}{\text{vol}_d(\mathcal{M})}\right). \tag{3}$$

The data-adaptive bound in Equation (3) exhibits the correct dependence on all relevant geometric and statistical quantities. First, L grows only logarithmically in the number of training samples N, so that adding more data increases the required number of layers at a vanishing rate. Second, the geometry of $\mathcal{M}$ enters through $\min\{\text{diam}(\mathcal{M}), \tau_\mathcal{M}\}$: the coarsest useful scale in the hierarchy is capped by the reach $\tau_\mathcal{M}$ — since no layer should have radius exceeding $\tau_\mathcal{M}$, beyond which the tangent plane approximation breaks down — and by diam$\mathcal{M}$, beyond which a single landmark suffices to cover the whole manifold. Lastly, L depends on intrinsic dimension d, instead of ambient dimension D. See Section B.1 for the derivation.

4.4 Traversing Over Hierarchical Networks

Given a noisy test sample $\boldsymbol{y} \in \mathbb{R}^D$, traversal begins at the coarsest layer $l = L$ from an initial landmark i (fixed or chosen uniformly at random from the set of landmarks) and proceeds by greedily minimizing the squared distance $\varphi = \|\boldsymbol{q}_i^{(l)} - \boldsymbol{y}\|^2$ to the current landmark. As described in Algorithm 2, we take first-order steps in the negative gradient direction, identifying the first-order neighbor most aligned with the desired direction of descent; otherwise, we fall back to a zero-order neighbor – the escape edge instantiated during training – and traverse it if it reduces the objective. When neither move improves φ, the traversal has converged on layer l. Line-by-line description of this algorithm is provided in Section I of the Appendix. If a cross-layer pointer $\pi^{(l)}(i)$ exists, we hop to the finer layer $l-1$ via $\pi^{(l)}(i)$ and resume traversal at the finer resolution; if no such pointer exists, the traversal is globally converged and the current landmark index i is returned as the estimate of the nearest neighbor to $\boldsymbol{y}$. An example traversal path is shown in Figure 2 (c). We provide a detailed complexity analysis in Appendix G

5 EXPERIMENTAL RESULTS

Choice of depth L. In our experiments, the number of layers L is set by theory: we evaluate Equation (3) after replacing its theoretical quantities with standard computational counterparts (West et al., 2001). This reduces the choice of L to the choice of two radii, the finest R_1 and the coarsest R_L. Given these, we set the intermediate radii to form a geometric sequence $R_l = \lambda^{L-l} R_L$ with decay factor $\lambda \in (0, 1)$. Since layer l has roughly $n_l \approx \mathrm{vol}(\mathcal{M})/R_l^d$ landmarks, consecutive layers differ in size by a factor of λ^{-d}. Choosing $\lambda = e^{-1/d}$, so that each coarser layer has e times fewer landmarks, gives

$$L \quad = \quad \left\lceil d \ln \frac{R_L}{R_1} \right\rceil + 1, \tag{4}$$

derived in Section B. It remains to estimate R_1 and R_L from the training data. We set R_1 to the p-th percentile of pairwise distances, so that the finest layer has many landmarks. The coarsest radius R_L should not exceed the manifold diameter, which we estimate by the graph diameter, i.e., the largest distance between any two training points; this is a natural proxy because the clean training points lie on the manifold. Within this bound, we set $R_L = (1 - \rho)\, r_{\min}$, where $r_{\min} \leq \mathrm{diam}\mathcal{M}$ is the minimum eccentricity of the training set (the smallest radius of a ball, centered at a training point, that encloses all the data), and the margin ρ controls how far R_L sits below this threshold. We fix $p = 1$ and vary $\rho \in \{0.01, 0.02, 0.05, 0.1\}$. Figure 2(a) reports the mean over this range with ± 1 standard deviation error bars, and Section C gives a larger sweep.

Datasets. We use gravitational-wave (GW) signals simulated with PyCBC (Nitz et al., 2024) (90k train / 10k test, $D = 2048$), with intrinsic dimension $d = 2$ (masses only) or $d = 8$ (masses and spins), as well as 4×4 and 14×14 patches from MNIST (LeCun et al., 1998b), CIFAR-10 (Krizhevsky et al., 2009), and CelebA (Liu et al., 2015). Details are described in Section C.

Methods and baselines. `SMT` traverses only the finest layer. `HMT` (ours) starts at the coarsest layer and descends to the finest via learned cross-layer maps $\pi^{(l)}$. `HMT-`ϵ leaves layer l early once the residual falls below $\epsilon_l = R_l^2$, and `HMT-`ϵ`-3L` uses only the finest and two coarsest layers. Further variants are defined in Section D. We also compare against kNN (Cover and Hart, 1967), HNSW (Malkov and Yashunin, 2018), nonlinear autoencoders (Hinton and Salakhutdinov, 2006), and `SMT` with multiple random restarts.

Evaluation on Time-Series Data. We perform denoising experiments on time-series data as well as images. For time-series, we use GW datasets described above, with black hole masses drawn from a Gaussian distribution. We evaluate mean-squared error (MSE) and the number of multiplications required at test time, and sweep hyperparameters ρ and p, using $\rho \in \{0.01, 0.02, 0.05, 0.1\}$ and fixing $p = 1\%$; error bars denote ± 1 standard deviation. Noisy inputs are generated as $\mathbf{y} = \mathbf{x} + \mathbf{z}$ with $\mathbf{z} \sim \mathcal{N}(0, \sigma^2)$ and $\sigma = 0.01$, yielding a target denoising MSE of $\sigma^2 d = 8 \cdot 10^{-4}$ (input noise level $\sigma^2 D = 0.205$), in dashed vertical lines.

Figure 2 shows that the `SMT` baseline is highly sensitive to initialization: under random start, it diverges, producing MSE exceeding the input noise level, while with a fixed start it remains above target despite low computational cost. In contrast, `HMT` variants achieve consistently low MSE across both initialization regimes. Notably, `HMT-`ϵ matches the accuracy of full `HMT` while reducing multiplications by approximately 45%. The three-layer variant (`HMT-`ϵ`-3`L) performs substantially worse, indicating that shallow hierarchies are insufficient at $d = 8$. These results support the interpretation that coarse-to-fine traversal acts as a *basin-finding mechanism*, guiding queries into the correct basin of attraction prior to local refinement.

Table 2 reports the MSE ratio and computational cost for all methods under both initialization regimes. At $d = 8$, `HMT-`ϵ reduces MSE by a median $134\times$ (116–$145\times$) over `SMT` under random initialization at only $3.9\times$ the cost. In Tables 3, 4, and 5 of the Appendix, we report results for $d = 2$ and mixed Gaussian+Uniform parameter settings for gravitational waves. These represent more challenging regimes in which the baseline degrades further, while hierarchical methods retain competitive accuracy.

Evaluation on Image Benchmarks. We apply the same protocol to unit-normalized patches from MNIST, CIFAR-10, and CelebA, which increase in visual complexity. The scale separation ratio R_L/R_1, computable from the training data before any experiment is run, predicts HMTN's benefit: it is low for MNIST (1.9), larger for CIFAR-10 (up to 39), and largest for CelebA (up to 75.7). Gains follow the same trend. Under random start, HMTN improves MSE by $1.62\times$ on MNIST, 7.9–$8.9\times$ on CIFAR-10, and 6.0–$9.4\times$ on CelebA. HMTN thus becomes more useful as data exhibit stronger multiscale structure. Dataset sizes, sweeps over patch size, stride, intrinsic dimension, and noise level, and full results are in Section C.3.

Comparison to Alternative Baselines. We further compare hierarchical traversal against additional baselines on GW data. Against brute-force *k-nearest-neighbor* search ($k = 1$), when denoising more complicated data with higher intrinsic dimension $d = 8$, HMTN outperforms kNN across all configurations. Against *HNSW*, HMTN is more accurate at $d = 8$ across all beam widths: HNSW's MSE is $\sim 8\times$ worse at its cheapest setting ($\mathrm{ef} = 1$) and still $\sim 4\times$ worse at its most expensive ($\mathrm{ef} = 128$), a gap we attribute to HMTN's geometry-aware tangent-plane projection. Against *nonlinear autoencoders* (12 architectures), HMTN is more accurate at comparable budgets ($\mathrm{MSE} \approx 1.1\times10^{-2}$ vs. 1.5×10^{-2} at $\sim10^5$ multiplications), and the cheapest autoencoder that matches HMTN's best accuracy requires $\sim 25\times$ more multiplications (Section D.3). Finally, to show that HMTN's gains are not an artifact of poor initialization, we compare against *SMT with k random restarts* ($k \in \{1, 2, 4, \ldots, 32\}$). With a single restart, SMT is 80–$85\times$ worse than HMTN, and matches HMTN's accuracy only after $k \approx 16$ ($d = 2$) to $k \approx 32$ ($d = 8$) restarts, at which point it uses 4.7–$5.4\times$ HMTN's compute. This confirms that HMTN's advantage stems from searching the space structurally. Full results are in Section D.

Compute resources. Experiments were run on a single CPU-only workstation. On the GW data, HMTN answers a query in under 3 ms using 2–$4\times$ the memory of the single-layer method, and one-time training takes under 1 hour. On the CIFAR-10 and CelebA patch sets, training takes about 3 hours (*c.f.* Section H).

Ablation on number of layers. Starting from a fully trained L-layer network, we construct sub-networks of depth $\ell = \{2, \ldots, L\}$ by retaining the coarsest and finest layers and progressively adding intermediate resolutions. For each sub-network, we measure test MSE and multiplications per query (Figure 3). Increasing the number of layers consistently reduces MSE, while computational cost increases only linearly.

6 Discussion

We propose Hierarchical Manifold Traversal Networks (HMTNs), a novel, geometry-aware coarse-to-fine traversal algorithm for manifold denoising. HMTNs approximate the manifold at different resolutions through a hierarchical graph, supported by theoretical analysis that justifies the hierarchy depth. HMTN provides data-adaptive measure of multiscale structure of manifolds, which can be computed prior to training, making this an effective way to measure expected gains. The hierarchy of graphs provides efficient pathways to optimal regions of the manifold while remaining robust to initialization, a critical property in many real-world settings. Compared to prior hierarchical search methods, HMTN provides better accuracy. We validated our method on challenging datasets of time-series gravitational waves, as well as image datasets MNIST, CIFAR-10, and CelebA patches. Our method achieved considerable gains in accuracy with minimal computational overhead as opposed to competitors.

Limitations and future work. In future work, we will explore integration of HMTN into generative denoising algorithms to enhance their efficiency as HMTN can improve inference-time efficiency of such iterative denoising algorithms. Additionally, for images with static, monotonous backgrounds, such as MNIST digits, the choice of patch size can create datapoints without any or massively repetitive information. We will explore how the choice of patch size affects multiscale structure in such dataset, and compare that phenomenon to more complex images.

AI USE STATEMENT

In this work, we used generative AI tools to polish writing and for experiment execution (building on top of human-generated codebase). LLM-generated code was verified and tested for correctness by the corresponding author.

REPRODUCIBILITY STATEMENT

All code and materials will be shared upon acceptance of the paper.

REFERENCES

Michal Aharon, Michael Elad, and Alfred Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

Apratim Bhattacharyya, Shweta Mahajan, Mario Fritz, Bernt Schiele, and Stefan Roth. Normalizing flows with multi-scale autoregressive priors. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8412–8421. IEEE, 2020.

Nicolas Boumal. *An introduction to optimization on smooth manifolds*. Cambridge University Press, 2023.

Antoni Buades, Bartomeu Coll, and J-M Morel. A non-local algorithm for image denoising. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 60–65. Ieee, 2005.

Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52 (2):489–509, 2006.

S Grace Chang, Bin Yu, and Martin Vetterli. Adaptive wavelet thresholding for image denoising and compression. *IEEE Transactions on Image Processing*, 9(9):1532–1546, 2000.

Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, 2007.

Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry*, pages 253–262, 2004.

David L Donoho. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006.

David L Donoho and Iain M Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81(3): 425–455, 1994.

Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing*, 15(12):3736–3745, 2006.

Dian Gong, Fei Sha, and Gérard Medioni. Locally linear denoising on image manifolds. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 265–272. JMLR Workshop and Conference Proceedings, 2010.

Eric Hedlin, Gopal Sharma, Shweta Mahajan, Hossam Isack, Abhishek Kar, Andrea Tagliasacchi, and Kwang Moo Yi. Unsupervised semantic correspondence using stable diffusion. *Advances in Neural Information Processing Systems*, 36:8266–8279, 2023.

Matthias Hein and Markus Maier. Manifold denoising. *Advances in Neural Information Processing Systems*, 19, 2006.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

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

Christian Horvat and Jean-Pascal Pfister. Denoising normalizing flow. *Advances in Neural Information Processing Systems*, 34:9099–9111, 2021.

Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, pages 604–613, 1998.

Stefan Kindermann, Stanley Osher, and Peter W Jones. Deblurring and denoising of images by nonlocal functionals. *Multiscale Modeling & Simulation*, 4(4):1091–1115, 2005.

Jon M. Kleinberg. Navigation in a small world. *Nature*, 406(6798):845–845, 2000.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

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

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998a.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998b.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 3730–3738, 2015.

Yu A. Malkov and Dmitry A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836, 2018.

Alex Nitz, Ian Harry, Duncan Brown, Christopher M Biwer, Josh Willis, Tito Dal Canton, Collin Capano, Thomas Dent, Larne Pekowsky, Gareth S Cabourn Davies, et al. gwastro/pycbc: v2. 3.2 release of pycbc. *Zenodo*, 2024.

Partha Niyogi, Stephen Smale, and Shmuel Weinberger. Finding the homology of submanifolds with high confidence from random samples. *Discrete & Computational Geometry*, 39(1):419–441, 2008.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

Yaniv Romano, Michael Elad, and Peyman Milanfar. The little engine that could: Regularization by denoising (red). *SIAM Journal on Imaging Sciences*, 10(4):1804–1844, 2017.

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.

Rongjie Shi, I-Fan Shen, Wenbin Chen, and Su Yang. Manifold learning for image denoising. In *The Fifth International Conference on Computer and Information Technology (CIT'05)*, pages 596–602. IEEE, 2005.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, pages 839–846. IEEE, 1998.

John Wilder Tukey et al. *Exploratory data analysis*, volume 2. Springer, 1977.

Singanallur V. Venkatakrishnan, Charles A. Bouman, and Brendt Wohlberg. Plug-and-play priors for model based reconstruction. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 945–948. IEEE, 2013.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge University Press, 2018.

Martin J Wainwright. *High-dimensional Statistics: A Non-Asymptotic Viewpoint*, volume 48. Cambridge University Press, 2019.

Shiyu Wang, Mariam Avagyan, Yihan Shen, Arnaud Lamy, Tingran Wang, Szabolcs Marka, Zsuzanna Marka, and John Wright. Fast, accurate manifold denoising by tunneling Riemannian optimization. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025.

Tangjun Wang, Zehao Dou, Chenglong Bao, and Zuoqiang Shi. Diffusion mechanism in residual neural network: theory and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46 (2):667–680, 2023.

Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684): 440–442, 1998.

Douglas Brent West et al. *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River, 2001.

John Wright, Allen Y Yang, Arvind Ganesh, S Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE transactions on pattern analysis and machine intelligence*, 31(2):210–227, 2008.

Jinxi Xiang, Yonggui Dong, and Yunjie Yang. Fista-net: Learning a fast iterative shrinkage thresholding network for inverse problems in imaging. *IEEE Transactions on Medical Imaging*, 40(5):1329–1339, 2021.

Kai Zhang, Yawei Li, Wangmeng Zuo, Lei Zhang, Luc Van Gool, and Radu Timofte. Plug-and-play image restoration with deep denoiser prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):6360–6376, 2021.

A NOTATION

In Table 1, we provide a comprehensive list of all elements used in this work.

Table 1: Notation of all parameters used in this work.

Symbol	Name	Description	Size / shape
d	Intrinsic dimension	Dimensionality of manifold $\mathcal{M}$	Scalar
D	Ambient dimension	Dimensionality of ambient space	Scalar
$\text{diam}(\mathcal{M})$	Manifold diameter	Max distance between points on $\mathcal{M}$	Scalar
$\mathcal{E}^0$	Zero-order edge	Helps escape local mins	$\mathcal{O}(1)$ lookup pointer
$\mathcal{E}^1$	First-order edge	Connects neighbors within radius R_{nbrs}	$\mathcal{O}(1)$ lookup pointer
$\mathcal{E}^{\otimes}$	Cross-layer edge	Connects consecutive layers l and $l-1$	$\mathcal{O}(1)$ lookup pointer
κ	Curvature	Geometric property of $\mathcal{M}$	Scalar
L	Hierarchy depth	Number of layers in the graph hierarchy	$\mathcal{O}(\log N)$
λ	Inter-layer ratio	Geometric series common ratio ($e^{-1/d}$)	Scalar
$\mathcal{M}$	Submanifold	Compact, smooth d-dimensional manifold	$\subset \mathbb{R}^D$
N	Training samples	Number of clean training points	Scalar
$\pi^{(l)}$	Cross-layer map	Routes coarse node to nearest in $l-1$	$V_l \to V_{l-1}$
$\boldsymbol{q}$	Landmark	Discrete approximation of $\mathcal{M}$	$\mathbb{R}^D$
R_l	Covering radius	Landmark net resolution at layer l	Scalar
τ	Reach	Radius of largest tube around $\mathcal{M}$ with unique projection	Scalar
T_i	Tangent space	Local linear approximation at vertex v_i	$\text{span}(\boldsymbol{U}_i)$
$\boldsymbol{U}_i$	Tangent basis	Orthogonal basis for T_i	$\mathbb{R}^{D\times d}$
$\text{vol}(\mathcal{M})$	Manifold volume	d-dimensional volume of $\mathcal{M}$	Scalar
$\varphi_{\boldsymbol{y}}(\boldsymbol{q})$	Objective function	Squared distance $\frac{1}{2}\|\boldsymbol{q}-\boldsymbol{y}\|_2^2$	Scalar
$\boldsymbol{x}$	Clean point	Ground truth point lying on $\mathcal{M}$	$\mathbb{R}^D$
Ξ	Collection of embeddings	Set of all edge embeddings for a landmark	$\mathbb{R}^{d\times \deg}$
ξ	Edge embedding	Projection of a chord onto the tangent space	$\mathbb{R}^d$
$\boldsymbol{y}$	Noisy point	Observed data point ($\boldsymbol{y}=\boldsymbol{x}+\boldsymbol{z}$)	$\mathbb{R}^D$
$\boldsymbol{z}$	Gaussian noise	Additive noise component	$\mathbb{R}^D$

B MODEL PARAMETERS

B.1 NUMBER OF LAYERS IN HIERARCHY (HIERARCHY DEPTH)

In the proposed Hierarchical Manifold Traversal (HMT) algorithm, we need to decide the number of layers L in order to build the graph. Given a compact, smooth manifold $\mathcal{M} \subseteq \mathbb{R}^D$, and a collection of clean points $\mathbf{x_1}, \ldots, \boldsymbol{x}_N \in \mathcal{M}$, below we derive an expression for the number of total layers L in the hierarchy based on the properties of the manifold $\mathcal{M}$, such as curvature κ and reach τ. For each layer, where the bottom layer is the 1-st and the top layer is the L-th, we define a covering radius $R_1 < R_2 < \cdots < R_L \in \mathbb{R}$. These covering radii create approximations of $\mathcal{M}$ at different resolutions – from coarse to fine. Inspired by Malkov and Yashunin (2018); Watts and Strogatz (1998); Kleinberg (2000), we formulate the covering radii to be a geometric series:

$$R_{l-1} = \lambda R_l, \quad \forall l \in \{L, \ldots, 1\} \tag{5}$$

where l is the layer index and λ is the common ratio in this geometric series. Because this is an exponentially strictly decaying sequence, it is clear that $0 < \lambda < 1$.

To have a meaningful approximation of $\mathcal{M}$, we must ensure that the largest radius R_L is within appropriate bounds, i.e., $R_L < \tau$. Then,

$$
\begin{aligned}
R_1 &= \lambda^{L-1} R_L && (6)\\
\ln R_1/R_L &= (L-1)\ln\lambda && (7)\\
\frac{\ln R_1/R_L}{\ln\lambda} &= L-1 && (8)\\
L &= 1 + \frac{\ln R_1/R_L}{\ln\lambda} && (9)
\end{aligned}
$$

It is meaningful to get an upper-bound on L, i.e., given the properties of $\mathcal{M}$, what is the maximum useful depth of the hierarchy, beyond which there is no point in having more layers? To this end, we provide detailed investigation on covering radii $R_1, \ldots, R_L$ in the next subsections. Meanwhile, by plugging equation 30, equation 16, and equation 22 into equation 9, we get

$$
\begin{aligned}
L &\leq 1 + \frac{\ln\left(\frac{\left(\frac{\text{vol}(\mathcal{M})}{N}\right)^{1/d}}{\min\{\text{diam}\mathcal{M},\tau\}}\right)}{\ln e^{-1/d}} && (10)\\
&\leq 1 + d\ln\left(\frac{\min\{\text{diam}\mathcal{M},\tau\}}{\left(\frac{\text{vol}(\mathcal{M})}{N}\right)^{1/d}}\right) && (11)\\
&\leq 1 + \ln\left(\frac{(\min\{\text{diam}\mathcal{M},\tau\})^d}{\left(\frac{\text{vol}(\mathcal{M})}{N}\right)}\right) && (12)\\
L &\leq 1 + \ln\left(\frac{N(\min\{\text{diam}\mathcal{M},\tau\})^d}{\text{vol}(\mathcal{M})}\right), && (13)
\end{aligned}
$$

which is the final expression for L. To make the dependence on the key quantities explicit, we rewrite the right-hand side as

$$
L \leq 1 + \ln N + d\ln\min\{\text{diam}(\mathcal{M}), \tau_{\mathcal{M}}\} - \ln\text{vol}(\mathcal{M}). \tag{14}
$$

Thus, the depth of the hierarchy grows *logarithmically* in the number of training samples N, at most *linearly* in the intrinsic dimension d, and is *independent* of the ambient dimension D.

Interpretation. We see that the number of layers L does not depend on ambient dimension D. It only depends on the properties of the manifold, such as intrinsic dimension d, the volume vol$\mathcal{M}$, diameter diam$\mathcal{M}$, reach τ, and the number of training points N. Thus, the number of layers is determined purely by the intrinsic geometry of the manifold and the dataset size, not by ambient dimension D.

We note that L depends on the number of training points N logarithmically, therefore more training points will not blow up the value of L. It depends on d at most linearly, as higher intrinsic dimension will require more covering.

For an intuitive interpretation, we further simplify the expression of L to be

$$
L \leq 1 + \ln\left(\frac{\min\{\text{diam}\mathcal{M},\tau\}}{R_1}\right)^d \tag{15}
$$

The ratio in the logarithm represents how many ε-balls of radius R_1 fit inside a single ball of volume vol$\mathcal{M}$.

Dependence of L on Curvature. We also notice the relationship between the number of layers L and manifold curvature. When curvature is high, the reach τ must be small, and $\tau <$ diam$\mathcal{M}$, and L depends on curvature. However, if the manifold is flat, then $\tau \to \infty$, and L depends on the diameter instead.

B.2 Lower Bound on R_1

To develop an expression for the number of layers needed to traverse efficiently and accurately, we must develop a lower bound on L that depends on the properties of the manifold. To this accord, we investigate covering radii as they control landmark placement, and develop a lower-bound for the finest radius R_1.

Proposition 1. Let $\mathcal{M} \subset \mathbb{R}^D$ be a compact d-dimensional Riemannian submanifold with reach $\tau > 0$ and intrinsic volume $\mathrm{vol}(\mathcal{M})$. Let $\mathcal{X} = \{x_1, \ldots, x_N\} \subset \mathcal{M}$ be a set of N training points, and suppose landmarks are selected via the online greedy cover of Algorithm 1: a new landmark is created at $\mathbf{x} \in \mathcal{X}$ only when no existing landmark lies within Euclidean distance R_1 of $\mathbf{x}$ (the `InitLandmark` condition). This yields a landmark set $\mathcal{Q} \subseteq \mathcal{X}$ whose centers satisfy $\|q_i - q_j\| > R_1$ for all $i \neq j$. Then

$$R_1 \;\gtrsim\; \left(\frac{\mathrm{vol}(\mathcal{M})}{N}\right)^{1/d}. \tag{16}$$

Proof. Since $\mathcal{Q} \subseteq \mathcal{X}$ and $\mathcal{Q}$ forms an R_1-net of $\mathcal{M}$ by construction, the covering number satisfies

$$\mathcal{N}(\mathcal{M}, R_1) \;\leq\; |\mathcal{Q}| \;\leq\; N. \tag{17}$$

It therefore suffices to establish a lower bound on $\mathcal{N}(\mathcal{M}, R_1)$ in terms of R_1 and $\mathrm{vol}(\mathcal{M})$.

Write $\mathcal{Q} = \{q_1, \ldots, q_m\}$ with $m = |\mathcal{Q}|$. By the `InitLandmark` condition, a point is added to $\mathcal{Q}$ only if it lies outside the Euclidean R_1-ball of every existing landmark; hence $\|q_i - q_j\| > R_1$ for all $i \neq j$. Consequently, the Euclidean balls $B(q_i, R_1/2)$ of radius $R_1/2$ are pairwise disjoint: if $B(q_i, R_1/2) \cap B(q_j, R_1/2) \neq \varnothing$ for some $i \neq j$, then $\|q_i - q_j\| < R_1$, a contradiction.

It remains to lower-bound the d-dimensional volume of each $B(q_i, R_1/2) \cap \mathcal{M}$. Since $R_1 < \tau$, the exponential map $\exp_{q_i} : T_{q_i}\mathcal{M} \to \mathcal{M}$ is a diffeomorphism on the ball of radius τ in the tangent space Niyogi et al. (2008). For $R_1 \ll \tau$, the manifold is locally flat at scale R_1, and the Euclidean ball $B(q_i, R_1/2)$ intersects $\mathcal{M}$ in a region whose d-dimensional volume satisfies

$$\mathrm{vol}_d(B(q_i,\, R_1/2) \cap \mathcal{M}) \;\geq\; c_d \left(\frac{R_1}{2}\right)^d, \tag{18}$$

where $c_d = \mathrm{vol}(B^d(0,1))$ is the volume of the unit d-ball. This follows because the exponential map pulls the intersection back to a full Euclidean ball in $T_{q_i}\mathcal{M} \cong \mathbb{R}^d$, with metric distortion of order $(R_1/\tau)^2$ Niyogi et al. (2008), which is negligible for $R_1 \ll \tau$ and absorbed into the $\gtrsim$ of equation 16. Since the intersections $\{B(q_i, R_1/2) \cap \mathcal{M}\}_{i=1}^m$ are pairwise disjoint and all contained in $\mathcal{M}$, summing their volumes gives

$$m \cdot c_d \left(\frac{R_1}{2}\right)^d \;\leq\; \mathrm{vol}(\mathcal{M}). \tag{19}$$

Since $m \geq \mathcal{N}(\mathcal{M}, R_1)$, rearranging gives

$$\mathcal{N}(\mathcal{M}, R_1) \;\geq\; \frac{c_d}{2^d}\,\frac{\mathrm{vol}(\mathcal{M})}{R_1^d}. \tag{20}$$

Combining equation 17 and equation 20 and solving for R_1,

$$N \geq \frac{c_d}{2^d} \frac{\text{vol}(\mathcal{M})}{R_1^d} \implies R_1 \geq \frac{c_d^{1/d}}{2} \left(\frac{\text{vol}(\mathcal{M})}{N} \right)^{1/d}, \tag{21}$$

establishing equation 16 with explicit constant $c_d^{1/d}/2$. □

Remark B.1. The bound equation 16 is purely intrinsic: it depends on d and $\text{vol}(\mathcal{M})$ but not on the ambient dimension D. The argument requires only $R_1 < \tau$ and the pairwise separation of landmarks, which is guaranteed by the `InitLandmark` condition. No distributional assumption on $\mathcal{X}$ is required, and the bound holds for any fixed training set.

B.3 Upper Bound on R_L

R_L is the coarsest covering radius and its choice should depend on the properties of the manifold. When the manifold is flat, its reach $\tau \to \infty$, in which case the manifold can be easily and well approximated by one point, and R_L can be as big as the diam$\mathcal{M}$ – the largest geodeic distance between any two points on the manifold. On the other hand, if the manifold is curvy, we must keep $R_L < \tau$ in order for this approximation to be meaningful and not lead us astray. Therefore, the upper-bound on R_L is

$$R_L = \min\{\text{diam}\mathcal{M}, \tau\}. \tag{22}$$

B.4 Choice of λ

Since λ is the ratio between radii of consecutive layers, and radii of consecutive layers decide number of landmarks in each layer, we take a closer look at the ratio of number of landmarks in layers l and $l+1$, i.e.,

$$\begin{aligned}
\frac{N(R_l)}{N(R_{l+1})} &= \frac{\text{vol}(\mathcal{M})/R_l^d}{\text{vol}(\mathcal{M})/R_{l+1}^d} && (23)\\
&= \frac{R_{l+1}^d}{R_l^d} && (24)\\
&= \left(\frac{R_{l+1}}{R_l} \right)^d && (25)\\
&= \left(\frac{R_{l+1}}{\lambda R_{l+1}} \right)^d && (26)\\
&= \lambda^{-d}. && (27)
\end{aligned}$$

In other words, to go from one layer to another, the number of landmarks will increase by a factor of λ^{-d}. Given our choice of geometric series of covering radii, we set $\lambda^{-d} = e$. Then, we derive an expression for λ to be

$$\begin{aligned}
-d \ln \lambda &= 1 && (28)\\
\ln \lambda &= -\frac{1}{d} && (29)\\
\lambda &= e^{-1/d}. && (30)
\end{aligned}$$

C Additional Experiments and Details

C.1 Main Results and Dataset Details of Gravitatial Waves (GW)

Our experiments use high-dimensional scientific time-series data derived from gravitational wave (GW) simulations. The GW datasets are generated using the PyCBC package (Nitz et al., 2024), with $90,000$ training samples and $10,000$ test samples. Each sample is a $D = 2048$-dimensional signal with intrinsic dimension $d \in \{2, 8\}$, corresponding to configurations without and with spin, respectively[3]. Figure 13 of the Appendix visualizes the low-dimensional manifold structure. Black hole mass parameters are sampled from a Gaussian distribution with mean 35 and standard deviation 15, and then clipped to the interval $[20, 50]$. For the $d = 8$ setting, in addition to the two mass parameters, the parameter space includes six spin components, representing rotation of each black hole about the x, y, and z axes. To investigate denser manifold coverage, we also construct a hybrid dataset of $200,000$ samples split evenly between Gaussian and uniform $[20, 50]$ mass parameters. Main results are shown in Table 2. Notably, under random start, toelrance-based HMTN denoises a median of 134 $\times$ better compared to single-layer traversal at the mere $3.9\times$ the cost.

Table 2: Results on gravitational waves dataset with black hole masses from Gaussian distribution ($d = 8$) for fixed and random starts.

Method	Fixed Start			Random Start		
	MSE improvement factor	MSE improvement factor median	Cost	MSE improvement factor	MSE improvement factor median	Cost
SMT	baseline (0.0201)		$1\times$	baseline (0.8349)		$1\times$
HMT	$2.3\times$–$3.5\times$	$2.9\times$	$4.5\times$	$116\times$–$145\times$	$134\times$	$7.2\times$
eHMT	$1.8\times$–$3.7\times$	$2.9\times$	$4.2\times$	$118\times$–$149\times$	$139\times$	$6.8\times$
HMT-ϵ	$2.3\times$–$3.5\times$	$2.9\times$	$2.5\times$	$\mathbf{116\times}$–$\mathbf{145\times}$	$\mathbf{134\times}$	$\mathbf{3.9\times}$
eHMT-ϵ	$1.1\times$–$3.7\times$	$1.9\times$	$2.2\times$	$73\times$–$148\times$	$122\times$	$3.6\times$
eHMT-ϵ-k	$2.3\times$–$3.5\times$	$2.9\times$	$2.5\times$	$116\times$–$145\times$	$134\times$	$4.0\times$
HMT-ϵ-3L	$0.6\times$–$1.0\times$	$0.9\times$	$0.9\times$	$3.4\times$–$5.8\times$	$5.3\times$	$1.3\times$

To investigate denser manifold coverage, we also construct a hybrid dataset of $200,000$ GW samples split evenly between Gaussian and uniform $[20, 50]$ mass parameters. Tables 3, 4, and 5 report results for gravitational-wave signals generated via PyCBC Nitz et al. (2024) with black hole parameters drawn from a Gaussian distribution ($d = 2$) and a Gaussian+Uniform distribution ($d = 2$ and $d = 8$). The Gaussian+Uniform setting mixes black hole parameters drawn from a Gaussian with those drawn uniformly from the parameter space, producing a more heterogeneous signal structure with better manifold coverage than the Gaussian-only case. We include HMT-ϵ-3L as the worst-case hierarchical baseline (a deliberately shallow hierarchy) to illustrate the cost of a poorly chosen L and highlight the importsmce of the theoretical layer-count prescription of Section B.

Across both parameter distributions, two consistent patterns emerge. Under **random initialization**, the hierarchy provides the largest gains, but the magnitude depends on both dimension and parameter distribution. At $d = 8$, HMT and its variants are robust across both parameter distributions: tolerance-based HMT achieves a median MSE improvement factor of $134\times$ (Gaussian parameters) (shown in Table 2) and $92\times$ (Gaus-

[3]Gravitational waves are minute ripples in spacetime generated by accelerating massive objects such as black holes. Denoising is a central task in gravitational wave astronomy, as it improves detection sensitivity.

Table 3: Results for gravitational-wave signals generated via PyCBC with black hole masses drawn from a Gaussian distribution ($d = 2$), for fixed and random starts. HMT and its variants demonstrate significantly higher MSE improvement gains in the random start regime, reaching the best median improvement of $496\times$.

Method	Fixed Start			Random Start		
	MSE improvement factor	MSE improvement factor median	Cost	MSE improvement factor	MSE improvement factor median	Cost
SMT	baseline (0.0103)		$1\times$	baseline (1.2033)		$1\times$
HMT	$0.1\times$–$2.0\times$	$1.3\times$	$2.0\times$	$60\times$–$518\times$	$\mathbf{496}\times$	$\mathbf{3.2}\times$
eHMT	$0.1\times$–$1.9\times$	$1.2\times$	$1.9\times$	$31\times$–$510\times$	$464\times$	$3.0\times$
HMT-ϵ	$0.9\times$–$2.0\times$	$1.4\times$	$1.5\times$	$75\times$–$599\times$	$\mathbf{398}\times$	$\mathbf{2.5}\times$
eHMT-ϵ	$0.9\times$–$1.9\times$	$1.4\times$	$1.4\times$	$39\times$–$588\times$	$377\times$	$2.2\times$
HMT-ϵ-k	$0.1\times$–$2.0\times$	$1.3\times$	$1.6\times$	$69\times$–$786\times$	$482\times$	$2.5\times$
HMT-ϵ-3L	$0.3\times$–$0.8\times$	$0.5\times$	$0.8\times$	$9.6\times$–$114\times$	$45\times$	$1.2\times$

Table 4: Results for gravitational-wave signals generated with black hole masses drawn from a Gaussian + Uniform distributions ($d = 2$), for fixed and random starts. HMT-ϵ provides the best accuracy-effeciency balance across both initialization regimes.

Method	Fixed Start			Random Start		
	MSE improvement factor	MSE improvement factor median	Cost	MSE improvement factor	MSE improvement factor median	Cost
SMT	baseline (0.0461)		$1\times$	baseline (1.2419)		$1\times$
HMT	$1.3\times$–$2.5\times$	$1.5\times$	$1.9\times$	$5.3\times$–$82\times$	$15\times$	$2.9\times$
eHMT	$1.2\times$–$1.9\times$	$1.5\times$	$1.8\times$	$5.8\times$–$84\times$	$12\times$	$2.6\times$
HMT-ϵ	$1.3\times$–$4.3\times$	$1.9\times$	$1.4\times$	$7.9\times$–$63\times$	$\mathbf{17}\times$	$\mathbf{2.1}\times$
eHMT-ϵ	$1.3\times$–$4.3\times$	$1.7\times$	$1.3\times$	$8.0\times$–$77\times$	$13\times$	$1.9\times$
HMT-ϵ-k	$0.6\times$–$2.4\times$	$1.3\times$	$1.5\times$	$5.3\times$–$118\times$	$15\times$	$2.1\times$
HMT-ϵ-3L	$0.6\times$–$1.2\times$	$0.8\times$	$0.9\times$	$4.6\times$–$27\times$	$7.7\times$	$1.2\times$

sian+Uniform parameters) (shown in Table 5) at the modest cost of $3.9\times$ and $4.1\times$ times, respectively, with tight ranges, indicating low sensitivity to ρ. HMT-ϵ consistently matches full HMT accuracy at roughly half the cost. At $d = 2$, with Gaussian parameters (Table 3), tolerance-based HMT achieves a median of $398\times$ at $2.5\times$ cost. With Gaussian and Uniform parameter distributions under random initialization (Table 4), HMT-ϵ improves MSE $17\times$, at the extra cost of only $2.1\times$. At $d = 8$ with Gaussian and Uniform distributed parametres (Table 5), we see that *added heterogeneity makes SMT less reliable even in the fixed initialization regime.* In all settings, HMT-ϵ-3L performs worst among hierarchical methods, confirming that a shallow hierarchy fails to provide adequate basin-finding and that a well-chosen L according to our theoretical recipe is essential.

C.2 Hyperparameter Sweeps

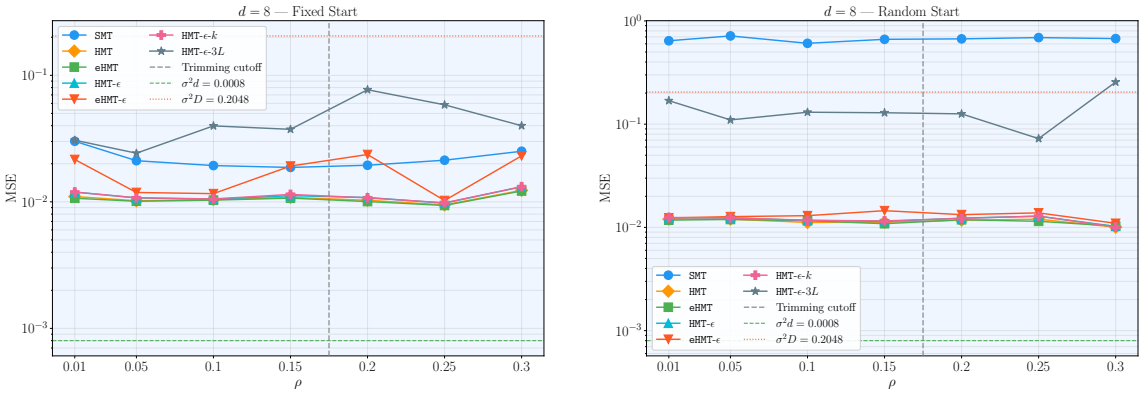
The hierarchical traversal algorithm is governed by two geometric hyperparameters. The finest-layer radius R_1 is set as the p-th percentile of pairwise distances, controlling the resolution at the bottom of the hierarchy; we use $p = 1\%$ in our main experiments. The coarsest-layer margin ρ sets the top-layer radius as $R_L = (1-\rho)\cdot \mathrm{ecc}_{\min}$, where $\mathrm{ecc}_{\min}$ is the minimum eccentricity of the training set, i.e., the smallest ball radius that still covers the entire dataset from some center point. Setting ρ close to zero ensures the coarsest layer spans the full input space, enabling the hierarchy to escape local minima under random initialization. The number of layers L is derived from both as $L = \lceil d \ln(R_L/R_1)\rceil + 1$ and is not a free parameter. In practice, ρ is the more

Table 5: Results for gravitational-wave signals generated via PyCBC with black hole parameters drawn from a Gaussian + Uniform distribution ($d = 8$), for fixed and random starts. In higher dimensions, the hierarchical methods continue to show substantial MSE ratio improvements, with eHMT reaching a 102× improvement in the random start scenario.

Method	Fixed Start			Random Start		
	MSE improvement factor	MSE improvement factor median	Cost	MSE improvement factor	MSE improvement factor median	Cost
SMT	baseline (0.0456)		1×	baseline (0.9667)		1×
HMT	3.8×–7.1×	4.3×	4.5×	87×–111×	90×	7.6×
eHMT	3.0×–7.1×	4.4×	4.3×	89×–118×	102×	7.2×
HMT-ϵ	4.2×–7.2×	4.8×	2.4×	87×–117×	**92**×	**4.1**×
eHMT-ϵ	2.4×–5.2×	3.8×	2.2×	84×–115×	88×	3.7×
HMT-ϵ-k	3.8×–7.1×	4.3×	2.5×	88×–113×	90×	4.1×
HMT-ϵ-3L	0.6×–1.0×	0.8×	1.0×	4.5×–6.2×	5.3×	1.5×

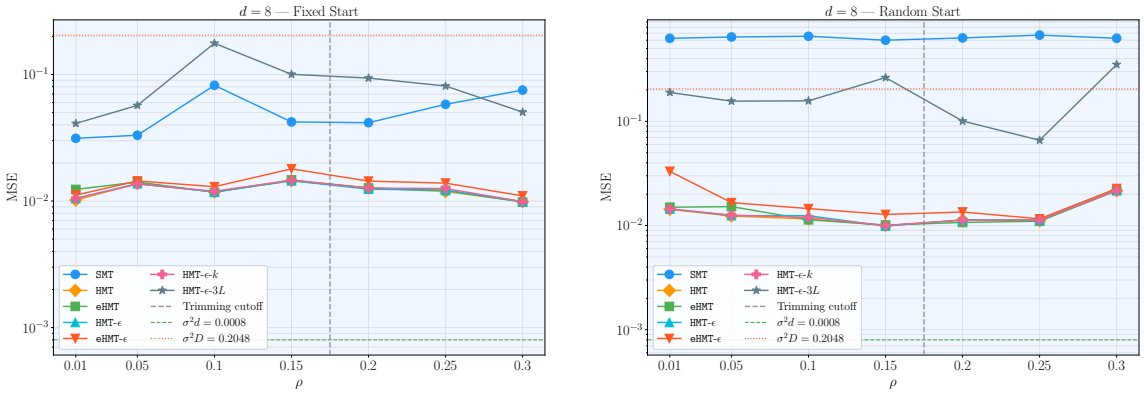
sensitive of the two: values of $\rho \geq 0.2$ cause the coarsest layer to lose global coverage, leading to degradation especially in low dimension and under random start. We therefore restrict our main evaluation to $\rho \in \{0.01, 0.02, 0.05, 0.1\}$ and provide a full sensitivity analysis over $\rho \in \{0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$ and $p \in \{1\%, 2\%, 5\%\}$ percentile for R_1 below.

Sensitivity at $d = 8$. Figures 4 and 5 show the effect of ρ on denoising MSE for gravitational-wave signals with black hole parameters drawn from Gaussian and Gaussian+Uniform distributions, respectively, at $d = 8$. The results are consistent across both parameter distributions: under fixed start, all variants of the hierarchical method cluster tightly around 10^{-2} across the full range $\rho \in [0.01, 0.3]$, exhibiting no meaningful sensitivity to the hyperparameter. Under random start, SMT is pinned near 0.5–0.6 regardless of ρ, while HMT, eHMT, HMT-ϵ, eHMT-ϵ, and HMT-ϵ-k remain clustered near 10^{-2}, maintaining a roughly 50–60× MSE advantage across the entire tested range. In both settings, HMT-ϵ-3L is the sole exception. These results confirm that at $d = 8$ the performance advantage of the full multilayer methods is a structural property of the hierarchy and not a consequence of careful hyperparameter tuning.



(a) Fixed start (b) Random start

Figure 4: **Hyperparameter Tuning.** Sensitivity of denoising MSE to ρ for gravitational-wave signals with parameters drawn from a Gaussian distribution, $d = 8$, under fixed start (left) and random start (right). Each point is the mean MSE averaged over $R_1 \in \{1\%, 2\%, 5\%\}$; the dashed vertical line marks the trimming cutoff $\rho = 0.175$.

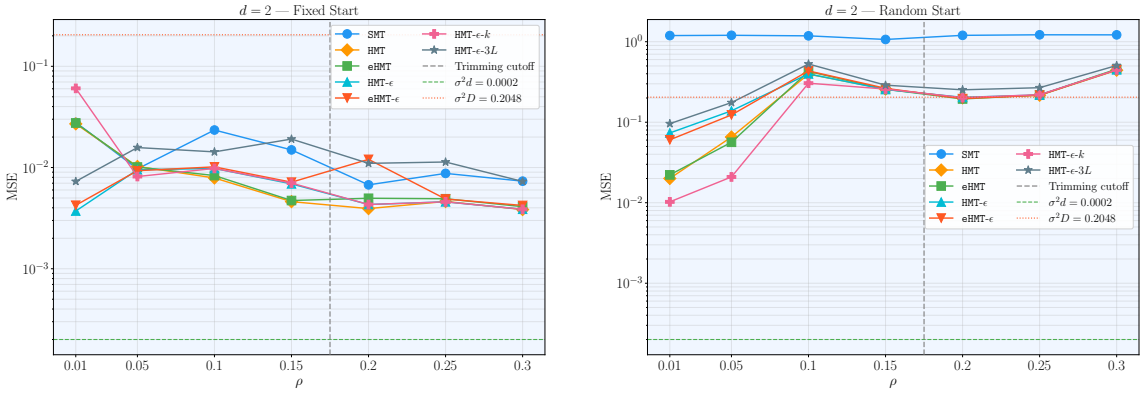


(a) Fixed start (b) Random start

Figure 5: **Hyperparameter Tuning.** Sensitivity of denoising MSE to ρ for gravitational-wave signals with parameters drawn from Gaussian + Uniform distributions, $d = 8$, under fixed start (left) and random start (right). Each point is the mean MSE averaged over $R_1 \in \{1\%, 2\%, 5\%\}$; the dashed vertical line marks the trimming cutoff $\rho = 0.175$.

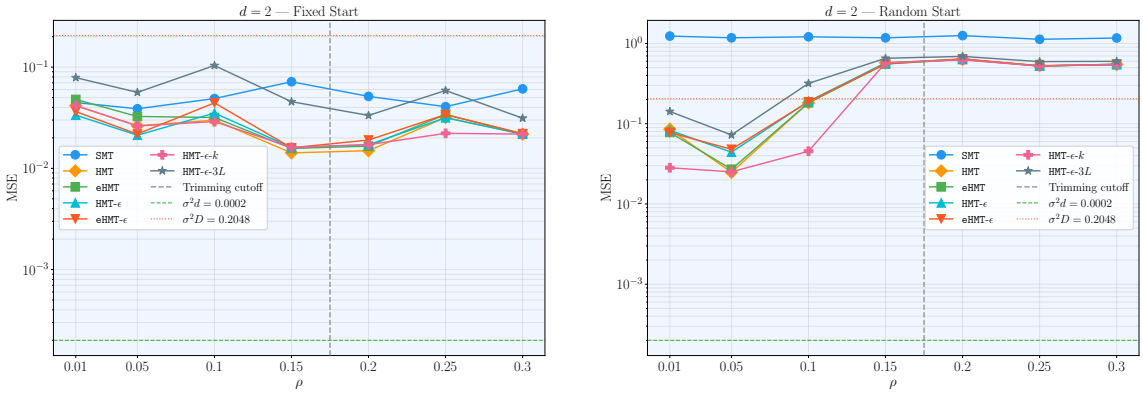
Sensitivity at $d = 2$. Figures 6 and 7 show the corresponding results at $d = 2$, where the sensitivity to ρ is considerably more severe and the two parameter distributions diverge. Under fixed start, no method maintains a stable advantage across the full ρ range for either parameter distribution, reflecting that fixed start in low dimension is an easier problem where the hierarchy provides only marginal benefit, if at all. Under random start, the sensitivity is stark: at $\rho \in \{0.01, 0.05\}$, HMT and its variants achieve MSE in the range 10^{-2}–10^{-1}, orders of magnitude below SMT. However, at $\rho = 0.1$ all methods degrade sharply, with MSE jumping above $\sigma^2 D = 0.2048$ — meaning the hierarchy is actively worsening noise and traversal is diverging, though still producing better results than SMT (recall that SMT consistently diverges in the random initialization regime). This collapse is consistent under Gaussian+Uniform parameters. These results confirm that $\rho \leq 0.05$ is the reliable operating regime for $d = 2$ random start, and that the trimming cutoff is not conservative but necessary. We therefore restrict our main evaluation to $\rho \in \{0.01, 0.02, 0.05, 0.1\}$, a range that densely covers the reliable operating regime ($\rho \leq 0.05$ for $d = 2$ random start) while including $\rho = 0.1$ to capture the onset of degradation at low dimension (these results are shown in the main paper), while the full sensitivity analysis over $\rho \in \{0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$ and $R_1 \in \{1\%, 2\%, 5\%\}$ in presented in this section of the Appendix.

MSE vs. multiplications across all hyperparameters. Figures 8 and 9 show MSE vs. multiplications with error bars absorbing variation over all $\rho \in \{0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$ and $p \in \{1\%, 2\%, 5\%\}$ for R_1, for gravitational-wave signals with parameters drawn from Gaussian+Uniform and Gaussian distributions, respectively. The patterns are consistent across parameter distributions. At $d = 8$, the hierarchy completely eliminates the random-start penalty: filled (random) and hollow (fixed) markers for HMT, eHMT, HMT-ϵ, eHMT-ϵ, and HMT-ϵ-k overlap at MSE $\approx 10^{-2}$, with near-horizontal connecting lines and narrow error bars in both axes, confirming robustness to both ρ and R_1. SMT, by contrast, has its two markers separated by roughly two orders of magnitude in MSE. HMT-ϵ and eHMT-ϵ achieve similar MSE to HMT and eHMT at roughly half the multiplication cost. At $d = 2$, the wide horizontal error bars under random start are driven primarily by ρ rather than R_1: HMT and eHMT have random-start means pulled to ≈ 0.3–0.4



(a) Fixed start (b) Random start

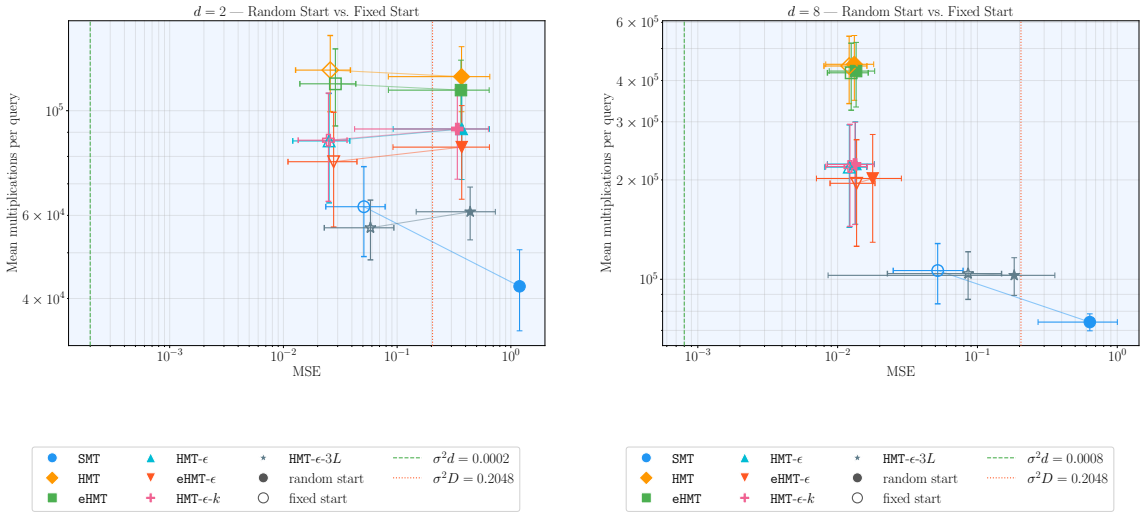
Figure 6: **Hyperparameter Tuning.** Sensitivity of denoising MSE to ρ for gravitational-wave signals with parameters drawn from a Gaussian distribution, $d = 2$, under fixed start (left) and random start (right). Under random start, all multilayer methods collapse above $\sigma^2 D$ for $\rho \geq 0.1$, confirming that small ρ is essential in a low dimensional setting.



(a) Fixed start (b) Random start

Figure 7: **Hyperparameter Tuning.** Sensitivity of denoising MSE to ρ for gravitational-wave signals with parameters drawn from Gaussian + Uniform distributions, $d = 2$, under fixed start (left) and random start (right). The Gaussian + Uniform parameter distribution amplifies the collapse at $\rho \geq 0.1$ under random start, with all multilayer methods rising above $\sigma^2 D$ by $\rho = 0.15$.

by catastrophically bad ρ values. Under fixed start, in the Gaussian+Uniform case, all hierarchical variants cluster near 10^{-2} with narrow bars in both dimensions.



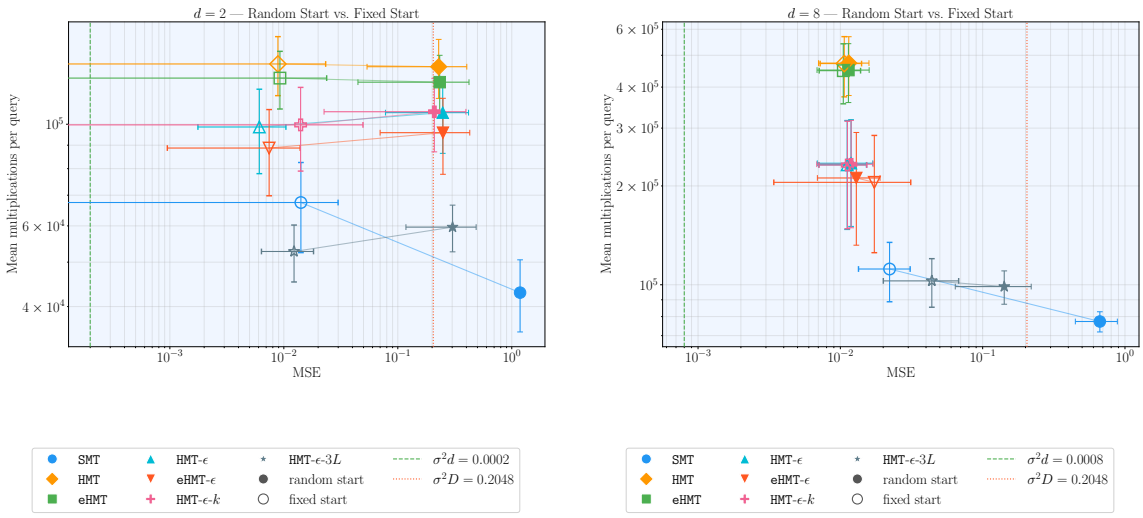
(a) Gaussian + Uniform parameters ($d = 2$) (b) Gaussian + Uniform parameters ($d = 8$)

Figure 8: **Hyperparameter Tuning.** MSE vs. multiplications for gravitational-wave signals generated via PyCBC with Gaussian + Uniform black hole parameters, at $d = 2$ (left) and $d = 8$ (right), with error bars over all $\rho \in \{0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$ and $R_1 \in \{1\%, 2\%, 5\%\}$. Filled markers denote random start, hollow markers fixed start. At $d = 8$ the multilayer methods completely close the random-start gap; at $d = 2$ the wide horizontal error bars are driven by ρ-sensitivity.

As ρ increases, $R_L = (1 - \rho) \cdot \mathrm{ecc}_{\min}$ decreases, reducing the ratio R_L/R_1 and consequently the number of layers $L = \lceil d \ln(R_L/R_1) \rceil + 1$. When ρ is large, the hierarchy becomes too shallow: the coarsest layer operates at a resolution close to that of the finest layer, and the geometric separation between levels that enables coarse-to-fine basin-finding is lost. In the limit, a single-level hierarchy is equivalent to `SMT`. Under random initialization this collapse is particularly damaging, as the traversal can no longer use coarse-layer structure to escape the basin of the initial position, and performance degrades toward that of `SMT`. This geometric argument explains why large ρ values are harmful independent of parameter distribution or dimension, and provides the theoretical basis for restricting ρ to the range where the hierarchy retains sufficient depth to be effective.

C.3 Evaluation on Standard Image Benchmarks

To show our reader that hierarchical traversal generalizes beyond gravitational-wave signals, we evaluate it on patch-denoising tasks drawn from three standard image dataset patches — MNIST, CIFAR-10, and CelebA — chosen to span a wide range of multiscale structure. Across all three, we use the same experimental protocol: patches are unit-normalized, Gaussian test-time noise with standard deviation $\sigma = 0.01$ is added, and the layer schedule is built with R_1-percentile p and coarseness margin ρ as described in Section 5. We compare two initializations — *fixed start* (deterministic start at landmark 0) and *random start*



(a) Gaussian parameters ($d = 2$) (b) Gaussian parameters ($d = 8$)

Figure 9: **Hyperparameter Tuning.** MSE vs. multiplications for gravitational-wave signals generated via PyCBC with Gaussian-distributed black hole parameters, at $d = 2$ (left) and $d = 8$ (right), with error bars over all $\rho \in \{0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$ and $R_1 \in \{1\%, 2\%, 5\%\}$. The extremely wide horizontal bars for `HMT` and `eHMT` at $d = 2$ are driven by ρ-sensitivity, while `HMT`$-\epsilon$ and `eHMT`$-\epsilon$ remain more robust.

(uniform random start over coarse landmarks) — and report, over $10,000$ held-out test points, the *MSE improvement* (median of single-layer MSE divided by hierarchical-traversal MSE, so values above 1 favor hierarchical traversal) and the *compute overhead* (median of hierarchical-traversal multiplications divided by single-layer multiplications).

Our central finding is that the scale ratio R_L/R_1 — the number of zoom levels spanned by the hierarchy, computable from the training data before any experiment is run — is a practical predictor of benefit: larger ratios onsistently yield larger MSE improvements. The three datasets below realize increasing scale separation ($R_L/R_1 \approx 1.9\times$ for MNIST, 9–39$\times$ for CIFAR-10, and up to $75.7\times$ for CelebA) and trace out this predicted trend.

C.3.1 Results on MNIST Patches

We extract 14×14 non-overlapping patches (stride $= 14$) from the MNIST handwritten digit images (LeCun et al., 1998a), yielding $N \approx 277{,}000$ unit-normalized patches of ambient dimension $D = 196$, with layer schedule built at $p = 1\%$, $\rho = 0.10$.

Results. Table 6 reports results for the base configuration ($d = 5$, $\sigma = 0.01$, 14×14 stride-14 patches) under both initializations, together with variations in patch size, stride, intrinsic dimension, and noise level that confirm R_L/R_1 remains low (1.6–1.9$\times$) regardless of these choices. With random start, HMTN achieves an MSE improvement of factor 1.62, which is a 38% reduction in denoising error, at $1.55\times$ multiplications overhead. This benefit is consistent with MNIST's low scale separation.

Interpretation. The failure of fixed start on MNIST reflects the same basin-of-attraction mechanism seen throughout this evaluation: without a random start, traversal begins from a single fixed node and depends entirely on the coarse layer's guidance from that point. When scale separation is small the coarse layer's guidance is weak, so a deterministic start is unlikely to escape a suboptimal fine-layer basin. A random start

instead has a chance of landing in a better coarse basin, and even the modest scale separation present here ($R_L/R_1 \approx 1.9\times$) turns that into a consistent accuracy gain.

Table 6: Hierarchical traversal on MNIST patches ($p = 1\%$, $\rho = 0.10$). *MSE improvement factor* is `median`(single-layer traversal MSE / hierarchical traversal MSE) over 10,000 test points: values above 1 mean HMTN achieves lower denoising error than single-layer. *Compute overhead* is `median`(HMTN mults / single-layer traversal mults): values above 1 mean HT uses more multiplications. Panel (a) compares fixed start and random start for the base configuration; panels (b) and (c) vary patch size, stride, intrinsic dimension, and noise level under both initializations, confirming R_L/R_1 remains low and random start consistently outperforms fixed start regardless of configuration.

Configuration	Init	L	R_L/R_1	MSE improvement factor	Compute overhead
(a) Base config ($d = 5$, $\sigma = 0.01$, 14×14 stride 14, $D = 196$)					
	fixed start	5	$1.92\times$	0.94	$1.15\times$
	random start	5	$1.92\times$	**1.62**	$1.55\times$
(b) Varying patch size and stride ($d = 5$, $\sigma = 0.01$)					
14×14, stride 14 (non-overlapping), $D = 196$	fixed start	5	$1.92\times$	0.94	$1.15\times$
	random start	5	$1.92\times$	**1.62**	$1.55\times$
14×14, stride 7 (overlapping), $D = 196$	fixed start	4	$1.68\times$	0.96	$0.71\times$
	random start	4	$1.68\times$	**1.22**	$1.25\times$
16×16, stride 4 (overlapping), $D = 256$	fixed start	4	$1.55\times$	1.03	$0.73\times$
	random start	4	$1.55\times$	**1.13**	$1.46\times$
(c) Varying intrinsic dimension and noise (14×14 stride 14, $D = 196$)					
$d = 5$, $\sigma = 0.01$	fixed start	5	$1.92\times$	0.94	$1.15\times$
	random start	5	$1.92\times$	**1.62**	$1.55\times$
$d = 5$, $\sigma = 0.05$	fixed start	5	$1.88\times$	1.09	$1.27\times$
	random start	5	$1.88\times$	**1.50**	$1.54\times$
$d = 7$, $\sigma = 0.01$	fixed start	6	$1.89\times$	0.97	$1.39\times$
	random start	6	$1.91\times$	**1.40**	$1.79\times$
$d = 11$, $\sigma = 0.01$	fixed start	9	$1.90\times$	1.02	$2.21\times$
	random start	9	$1.93\times$	**1.34**	$2.47\times$
$d = 12$, $\sigma = 0.01$	fixed start	9	$1.88\times$	0.96	$2.04\times$
	random start	9	$1.91\times$	**1.30**	$2.45\times$
$d = 13$, $\sigma = 0.01$	fixed start	10	$1.88\times$	1.06	$2.22\times$
	random start	10	$1.88\times$	**1.28**	$2.63\times$

C.3.2 CIFAR-10 Patch Dataset Results

We extract 4×4 patches from CIFAR-10 (Krizhevsky, 2009), converting each 32×32 RGB image to grayscale luminance via the ITU-R BT.601 formula before patching. With stride 2 and a unit-norm filter (threshold 0.5), this yields $N \approx 13.1$ million patches of ambient dimension $D = 16$. Due to having patch size 4×4, we set intrinsic dimension $d = 2$. We hold out 10,000 patches as a test set and train on a random subset of $N_{\text{train}} = 50{,}000$ of the remaining patches. The hierarchy is built from clean patches, and Gaussian noise with $\sigma = 0.01$ is added to the test patches only. The finest-layer radius R_1 is set to the p-th percentile of pairwise distances and the coarsest radius R_L is set using the coarseness margin ρ; we sweep $p \in \{1\%, 5\%\}$ and $\rho \in \{0.05, 0.10, 0.20, 0.30\}$, which gives $L = 8$–9 layers for $p = 1\%$ and $L = 6$–7

layers for $p = 5\%$. Each configuration (p, ρ) is trained separately for each of the two initialization modes (fixed start and random start), and every trained network is evaluated on the 10,000 test patches.

In contrast to MNIST under any configuration, CIFAR-10 4×4 patches exhibit substantially larger scale separation: across our sweep, R_L/R_1 ranges from $9\times$ to $39\times$ (23–$39\times$ for $p = 1\%$ and 9–$15\times$ for $p = 5\%$; Table 8). Table 7 reports the median MSE improvement and median compute overhead for each configuration. In the case of **fine** R_1 **($p = 1\%$) under random start,** MSE improvement is 7.9–$8.9\times$ across all four values of ρ, reflecting the many layers traversed combined with a random initialization that forces a full coarse-to-fine descent from an arbitrary starting point. Once R_L/R_1 is large enough for the coarse layer to meaningfully disambiguate, hierarchical traversal converges to the correct landmark while single-layer search gets stuck in local basins.

Table 7: Hierarchical traversal on CIFAR-10 4×4 grayscale patches ($D = 16$, $d = 2$, $\sigma = 0.01$). Each cell reports median MSE improvement | median compute overhead over 10,000 test points, for four coarseness margins ρ. R_L/R_1 is the scale ratio for $\rho = 0.10$; it varies by $\pm 3\times$ across other ρ values.

p	Init	R_L/R_1	$\rho = 0.05$	$\rho = 0.10$	$\rho = 0.20$	$\rho = 0.30$
1%	Fixed	$\sim 34\times$	$1.14\times$ | $2.9\times$	$1.32\times$ | $4.7\times$	$1.27\times$ | $5.4\times$	$1.19\times$ | $3.8\times$
	Random	$\sim 30\times$	$\mathbf{8.93}\times$ | $121\times$	$\mathbf{8.26}\times$ | $105\times$	$\mathbf{7.94}\times$ | $101\times$	$\mathbf{8.47}\times$ | $114\times$
5%	Fixed	$\sim 13\times$	$1.00\times$ | $2.1\times$	$1.00\times$ | $2.0\times$	$1.00\times$ | $2.3\times$	$1.00\times$ | $2.1\times$
	Random	$\sim 14\times$	$1.47\times$ | $3.2\times$	$1.55\times$ | $3.6\times$	$1.34\times$ | $2.8\times$	$1.72\times$ | $3.8\times$

Table 8: Exact scale ratio R_L/R_1 for every configuration in the CIFAR-10 sweep (Table 7), replacing the single $\rho = 0.10$ estimate with the value used at each ρ.

p	Init	$\rho = 0.05$	$\rho = 0.10$	$\rho = 0.20$	$\rho = 0.30$
1%	Fixed	$28.2\times$	$33.7\times$	$27.9\times$	$23.2\times$
	Random	$38.6\times$	$30.3\times$	$25.8\times$	$24.0\times$
5%	Fixed	$14.5\times$	$12.8\times$	$12.2\times$	$9.4\times$
	Random	$12.9\times$	$14.1\times$	$10.8\times$	$11.7\times$

C.3.3 Results on CelebA

We extract 4×4 grayscale patches from the CelebA aligned face dataset (Liu et al., 2015) (202,599 celebrity portraits). Each 178×218 image is center-cropped to 178×178 (discarding only the extra vertical margin), resized to 64×64, and converted to grayscale luminance via the ITU-R BT.601 formula before patching. With stride 8 and a unit-norm filter (threshold 0.5), this yields $N \approx 11.2$ million patches of ambient dimension $D = 16$, and we set $d = 2$ as for CIFAR-10. The stride is larger than CIFAR-10's stride 2 because CelebA has roughly $3.4\times$ as many source images, which keeps the patch count of the same order as CIFAR-10 (11.2M vs. 13.1M).

We hold out 10,000 patches as a test set and train on a random subset of $N_{\text{train}} = 50{,}000$ of the remaining patches. The hierarchy is built from clean patches, and Gaussian noise with $\sigma = 0.01$ is added to the test patches only. We set the finest-layer radius R_1 to the $p = 5\%$ percentile of pairwise distances and sweep the coarseness margin $\rho \in \{0.05, 0.10, 0.20, 0.30\}$, which gives $L = 7$–8 layers with $R_L/R_1 \approx 17$–26. At the finest setting $p = 1\%$ we run a single calibration ($\rho = 0.10$, fixed start), giving $L = 10$ layers with $R_L/R_1 = 75.7$; the full $p = 1\%$ sweep is omitted because each run is prohibitively expensive at this scale. Each configuration is trained separately for each of the two initialization modes (fixed start and random start), and every trained network is evaluated on the 10,000 test patches.

CelebA patches exhibit substantially stronger scale separation than CIFAR-10. At $p = 1\%$, $\rho = 0.10$, the layer schedule has $L = 10$ layers with $R_L/R_1 = 75.7\times$ — more than double CIFAR-10's $33.7\times$ under the same hyperparameters. Table 9 shows this is driven almost entirely by R_1: because all patches are unit-normalized, the coarsest radius R_L is similar for both datasets (≈ 0.9 at $\rho = 0.10$), but R_1 (the p-th percentile of pairwise distances) is roughly $2.5\times$ smaller for CelebA at $p = 1\%$ ($1.8\times$ at $p = 5\%$). Face patches contain large near-uniform regions that unit-normalization maps to nearly identical vectors, producing many near-duplicate patches with very small pairwise distances: 28% of CelebA patches lie within 0.05 of the uniform patch, compared with 17% for CIFAR-10.

Table 9: Raw R_1 and R_L values underlying the scale ratios reported above, at $\rho = 0.10$, fixed start (one representative run per row, for CelebA patch dataset. The $p = 1\%$ rows are the calibration runs, as in Table 10/Table 7). R_L is nearly identical across datasets at matched R_1-percentile p, since all patches are unit-normalized; the difference in R_L/R_1 is driven almost entirely by R_1.

Dataset	p	R_1	R_L	R_L/R_1
CIFAR-10	5%	0.0681	0.874	$12.8\times$
CIFAR-10	1%	0.0268	0.902	$33.7\times$
CelebA	5%	0.0373	0.929	$24.9\times$
CelebA	1%	0.0118	0.894	$75.7\times$

Results across four coarseness margins $\rho \in \{0.05, 0.10, 0.20, 0.30\}$ at $p = 5\%$ under both initializations are shown in Table 10, together with the $p = 1\%$, $\rho = 0.10$ calibration run (the full $p = 1\%$ sweep is omitted: $R_1 = 0.0078$ produces $L = 11$ layers that are prohibitively slow to train). The pattern mirrors CIFAR-10 but is more pronounced. Under **random start** and any ρ, MSE improvement rises to 6–$10\times$ across all four values of ρ. With fine R_1 ($p = 1\%$), even with a fixed start, this run achieves MSE improvement $1.64\times$ at $R_L/R_1 = 75.7\times$: the much larger scale separation lets the coarse layer provide a useful warm start even from a fixed initial node.

The calibration result shows that with sufficient scale separation the coarse-to-fine descent provides meaningful guidance even without random initialization: the large scale ratio produces a deep layer schedule ($L = 10$), so traversal passes through enough intermediate scales for the descent itself to meaningfully refine the solution at each step, rather than the result depending solely on where a fixed start happens to land at the coarsest layer.

Table 10: Hierarchical traversal on CelebA 4×4 grayscale patches ($D = 16$, $d = 2$, $\sigma = 0.01$). For each configuration, the R_L/R_1 row reports the realized scale ratio at each coarseness margin ρ, and the row below it reports median MSE improvement | median compute overhead over 10,000 test points. The $p = 1\%$ row uses only the calibration run ($\rho = 0.10$); the full $p = 1\%$ sweep is omitted due to resulting in $R_1 = 0.0078$, which is too small to be meaningful.

p	Initialization regime	$\rho = 0.05$	$\rho = 0.10$	$\rho = 0.20$	$\rho = 0.30$
5%	Fixed: R_L/R_1	$25.9\times$	$24.9\times$	$21.9\times$	$16.9\times$
	Fixed: MSE impr. | overhead	$1.00\times$ | $2.5\times$	$1.00\times$ | $4.0\times$	$1.00\times$ | $2.9\times$	$1.00\times$ | $2.5\times$
	Random: R_L/R_1	$24.8\times$	$22.4\times$	$22.5\times$	$19.3\times$
	Random: MSE impr. | overhead	$\mathbf{7.58}\times$ | $26\times$	$\mathbf{5.99}\times$ | $18\times$	$\mathbf{9.43}\times$ | $42\times$	$\mathbf{8.00}\times$ | $30\times$

D Comparison to Alternative Search and Learning-Based Baselines

We compare our proposed method against other traversal-based methods, as well as classical search and learning-based methods.

Classical Search and Learning-based Baselines. To show the effectiveness and strength of HMTN with respect to tried-and-true methods in the field, we compare against kNN search (Cover and Hart, 1967), hierarchical navigable small world (HNSW) networks (Malkov and Yashunin, 2018), and nonlinear autoencoders (Hinton and Salakhutdinov, 2006).

Traversal-based baselines include the single-layer manifold traversal (`SMT`) which performs traversal exclusively on the finest (highest-resolution) layer of the hierarchy, with no coarse-to-fine guidance. `HMT` is our approach with complete hierarchical traversal: beginning at the coarsest layer, it descends level by level to the finest, using learned cross-layer mappings $\pi^{(l)}$ to propagate candidate landmarks across layers. Efficient HMT (`eHMT`) follows the same process as `HMT` but keeps the objective value after cross-layer hops fixed rather than recomputing it, reducing the per-transition computational overhead. `HMT-`ϵ augments `HMT` with a tolerance-based early-exit criterion: traversal halts at layer l whenever the current residual falls below the per-layer threshold $\epsilon_l = R_l^2$, delegating further traversal to finer layers. `eHMT-`ϵ combines the tolerance exit of `HMT-`ϵ with `eHMT`. `HMT-`ϵ`-`k modifies the tolerance traversal by performing an exhaustive scan over all landmarks at any layer containing k or fewer landmarks, creating a hybrid method comprised of k-nearest neighbor search and hierarchical manifold traversal. `HMT-`ϵ`-3L` applies the tolerance traversal to a three-layer sub-network, with the finest layer and the top two coarsest ones serving as a reduced-complexity variant. Lastly, we also compare our method against multiple-restart `SMT`.

We further compare hierarchical traversal (HT) against three additional baselines on the same gravitational-wave sweep used in Section 5: brute-force nearest-landmark search, Hierarchical Navigable Small World search (HNSW), and nonlinear autoencoders.

D.1 Comparison to Brute-Force k-Nearest-Neighbor Search

To situate hierarchical traversal (HT) relative to the simplest possible alternative, we compare it against a brute-force k-nearest-neighbor search with $k = 1$ over the finest-layer landmark set. For a noisy query $\boldsymbol{x}$, this baseline evaluates $\|\boldsymbol{q}_j - \boldsymbol{x}\|^2$ against every landmark $\boldsymbol{q}_j$ in the finest layer and selects the minimizer $\boldsymbol{q}_{j^*}$. Unlike HT and single-layer, which denoise with the tangent-plane projection $\hat{\boldsymbol{x}} = \boldsymbol{q}_{j^*} + \mathbf{U}_{j^*}\mathbf{U}_{j^*}^\top(\boldsymbol{x} - \boldsymbol{q}_{j^*})$, the kNN baseline returns the landmark position *verbatim*, $\hat{\boldsymbol{x}} = \boldsymbol{q}_{j^*}$, with no local linear correction. This is a deliberate asymmetry: kNN represents a naive nearest-neighbor lookup — the true nearest landmark, but none of the local-model refinement that HT and single-layer apply — so the comparison isolates how much of HT/single-layer's accuracy comes from the tangent-plane fit itself versus from which landmark is selected. The finest layer is the natural landmark set for this comparison: it is the full-resolution set that HT's coarse-to-fine hierarchy is designed to search more cheaply than exhaustively. Following the exhaustive-search cost convention used elsewhere in this work, the brute-force search over N_L finest-layer landmarks is charged $D \cdot N_L$ multiplications.

We summarize the trade-off with two ratios computed per configuration: a *compute overhead*, $M_{\mathrm{HT}}/M_{\mathrm{kNN}}$ (multiplications HT uses relative to brute force — expected below 1, since HT is designed to avoid the full exhaustive scan), and an *MSE ratio*, $\mathrm{MSE}_{\mathrm{kNN}}/\mathrm{MSE}_{\mathrm{HT}}$ (kNN's error relative to HT's). Because kNN no longer receives the tangent-plane correction, this MSE ratio is *not* guaranteed to fall below 1 the way it would if both methods used the same denoising step: whether kNN or HT is more accurate now depends on whether landmark selection or local correction dominates the error in a given regime.

Setup. We evaluate the kNN baseline post hoc against all 224 trained networks from the main sweep, spanning $d \in \{2, 8\}$, both initialization modes (rs0: fixed start; rs1: random start), R_1-percentile $p \in \{1\%, 2\%, 5\%\}$, and coarseness margin $\rho \in \{0.01, 0.02, 0.05, 0.10, 0.15, 0.20, 0.25, 0.30\}$, with $\sigma = 0.01$ and 10,000 test queries per configuration. No networks were retrained for this comparison; the baseline is computed directly from the saved trained networks.

Results. Figure 10 places single-layer, HT, and the kNN baseline onv the same MSE-vs-multiplications axes. The compute overhead is unaffected by the denoising-formula change and matches our earlier result: HT uses a median of $5.7\times$ fewer multiplications than brute force ($M_{\mathrm{HT}}/M_{\mathrm{kNN}}$ median $0.175\times$, range $0.044\times$–$1.192\times$). The MSE ratio, however, now splits sharply by intrinsic dimension. Table 11 reports the breakdown, including the fraction of runs in which kNN is outright worse than HT. At $d = 8$, HT is more accurate than kNN in **every one of** $112/112$ **configurations** (both start modes): the median MSE ratio is 4.1–$4.2\times$, meaning naive lookup at the true nearest landmark still errs roughly $4\times$ more than HT's traversal with its tangent-plane correction. At $d = 2$ with a fixed start, the two methods are closely matched (median ratio $0.76\times$, kNN worse in $13/56$ configurations). At $d = 2$ with a random start, kNN is more accurate in $55/56$ configurations, often substantially so (median ratio $0.034\times$, i.e. HT's error is typically $\sim 29\times$ larger) — the same rs1 fragility at $d = 2$ documented in the coarseness-margin sensitivity analysis in the appendix.

Table 11: Hierarchical traversal (HT) versus naive kNN ($k = 1$, no tangent correction) over finest-layer landmarks, aggregated over R_1-percentile $p \in \{1, 2, 5\}\%$ and coarseness margin $\rho \in \{0.01, \ldots, 0.30\}$ ($n = 56$ runs per group, 224 total). Compute overhead and MSE ratio are medians per group; ranges in brackets. "kNN worse" counts configurations where $\mathrm{MSE}_{\mathrm{kNN}} > \mathrm{MSE}_{\mathrm{HT}}$.

Group	n	Overhead $M_{\mathrm{HT}}/M_{\mathrm{kNN}}$	MSE ratio $\mathrm{MSE}_{\mathrm{kNN}}/\mathrm{MSE}_{\mathrm{HT}}$	kNN worse
$d = 8$, fixed start (rs0)	56	$0.288\times$ $[0.131, 1.155]$	$4.196\times$ $[2.980, 22.082]$	$56/56$
$d = 8$, random start (rs1)	56	$0.285\times$ $[0.135, 1.192]$	$4.128\times$ $[1.656, 27.382]$	$56/56$
$d = 2$, fixed start (rs0)	56	$0.075\times$ $[0.044, 0.302]$	$0.763\times$ $[0.013, 2.069]$	$13/56$
$d = 2$, random start (rs1)	56	$0.071\times$ $[0.044, 0.319]$	$0.034\times$ $[0.004, 1.309]$	$1/56$
Overall	224	$0.175\times$ $[0.044, 1.192]$	$2.067\times$ $[0.004, 27.382]$	$126/224$

Interpretation. Because kNN always finds the true nearest landmark but never applies the tangent-plane correction, its accuracy relative to HT isolates which ingredient — landmark selection or local linear correction — dominates the error in each regime. At $d = 8$, HT beats kNN in every configuration tested: even the best possible landmark choice, without the tangent-plane fit, is a worse denoiser than HT's traversal with correction. This says the correction step is doing most of the work at $d = 8$, and HT's traversal is finding landmarks good enough that the correction can act effectively. At $d = 2$ with a random start, the relationship inverts: kNN wins in $55/56$ configurations, often by more than an order of magnitude, meaning HT's traversal is frequently converging to a landmark poor enough that no local correction can compensate — the bottleneck there is landmark *selection*, not the correction step, consistent with the random coarsest-layer start leading the descent into the wrong region when the coarse layer is sparsely populated, as governed by the layer-count schedule $L = \lceil d \ln(R_L/R_1) \rceil + 1$. The $d = 2$ fixed-start group falls in between (kNN worse in $13/56$ configurations), consistent with a deterministic start avoiding the worst random-start failures while still occasionally landing HT on a suboptimal landmark.

Summary. HT uses a median of $5.7\times$ fewer multiplications than exhaustive nearest-landmark search across the full sweep, but whether it is also more *accurate* than naive nearest-neighbor lookup depends entirely on intrinsic dimension and initialization: at $d = 8$, HT's tangent-plane correction makes it strictly more accurate than kNN in every configuration tested; at $d = 2$ with a random start, HT's traversal itself becomes unreliable enough that plain lookup at the true nearest landmark wins in nearly every configuration. This decomposition — compute overhead from the search strategy, accuracy from the interaction of landmark selection and local correction — gives a more complete picture than compute or accuracy alone: HT is cheap everywhere, but its accuracy advantage over the simplest possible baseline is concentrated in the higher-intrinsic-dimension regime where the tangent-plane correction has the most work to do.

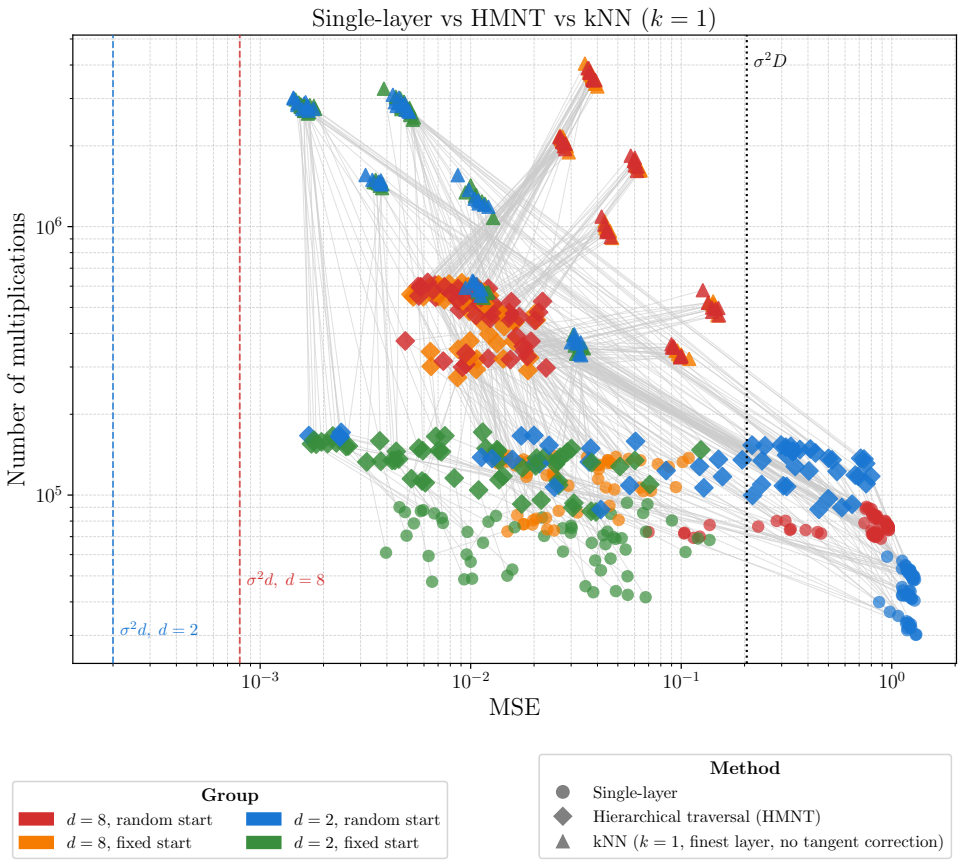


Figure 10: MSE versus multiplications for single-layer (circles), hierarchical traversal (diamonds), and naive kNN, $k = 1$, no tangent correction (triangles), across all 224 sweep configurations. Color encodes intrinsic dimension and start mode; grey lines connect the three methods within one configuration. Vertical lines mark $\sigma^2 D$ (input noise floor) and $\sigma^2 d$ (target lower bound) for $d \in \{2, 8\}$. At $d = 8$ (red/orange) kNN sits to the right of HT (higher MSE); at $d = 2$ (blue/green) the two methods interleave, with $d = 2$ random-start (blue) kNN points often to the left of HT.

D.2 Comparison to Hierarchical Navigable Small World (HNSW) Search

We also compare hierarchical traversal (HT) against HNSW (Malkov and Yashunin, 2018), the approximate-nearest-neighbor method most structurally similar to HT: both build a graph over a fixed point set and search it coarse-to-fine. Unlike the exact brute-force kNN baseline (Section D.1), HNSW is itself approximate, with a query-time beam width ef that trades accuracy for cost, analogous to HT's coarseness margin ρ.

Because standard HNSW libraries do not report per-query distance evaluations, we implement HNSW directly over the finest-layer landmarks ($M = 16$, $\text{ef}_{\text{construction}} = 200$) and charge D multiplications per distance evaluation, as elsewhere in this work. Index construction is a one-time cost reported separately, just as HT's training cost is excluded from its per-query counts. We evaluate all 224 trained networks from the main sweep, with $\text{ef} \in \{1, 2, \ldots, 128\}$.

When HNSW finds the true nearest landmark, this does not necessarily translate into lower error. At $d = 8$, HT is more accurate in every configuration, and widening HNSW's beam does not close the gap: even

when HNSW finds the correct landmark, its error remains several times HT's. Wall-clock time tracks the multiplication counts (Figure 11).

Takeaway. Without taking into account the geometry of data manifolds, HNSW cannot match HT's accuracy at $d = 8$ at any beam width (even at its most expensive setting). HNSW is a search method that is not geometry-aware, while HMTN uses local approximations of data manifolds (tangent space projection) to obtain more accurate results.

Table 12: HT versus HNSW at HNSW's cheapest ($\mathrm{ef} = 1$) and most expensive ($\mathrm{ef} = 128$) settings; $n = 56$ configurations per group. HT/HNSW MSE < 1 means HT is more accurate; HNSW/HT mults > 1 means HT is cheaper. All values are medians.

Group	ef	n	HNSW/HT mults	HT/HNSW MSE
$d = 2$, fixed start	1	56	$1.04\times$	$0.40\times$
$d = 2$, fixed start	128	56	$3.28\times$	$0.71\times$
$d = 2$, random start	1	56	$1.08\times$	$3.68\times$
$d = 2$, random start	128	56	$3.38\times$	$7.98\times$
$d = 8$, fixed start	1	56	$0.30\times$	$0.12\times$
$d = 8$, fixed start	128	56	$1.59\times$	$0.23\times$
$d = 8$, random start	1	56	$0.30\times$	$0.13\times$
$d = 8$, random start	128	56	$\mathbf{1.62}\times$	$\mathbf{0.23}\times$
Overall	1	224	$0.67\times$	$0.17\times$
Overall	128	224	$2.43\times$	$0.28\times$

D.3 AUTOENCODERS

We include nonlinear autoencoders as an additional baseline for denoising gravitational wave signals, as they are generic learning architectures specifically designed to leverage low-dimensional structure in data. To this end, we create 12 different networks (see Table 13) with various depths and widths and symmetric encoder/decoders. The widest layers match the input dimensionality, while the narrowest bottleneck layer is set to dimension 2. All autoencoders are trained using the Adam optimizer with a learning rate of 1×10^{-3}. As we can see in Figure 12, high-complexity autoencoders can reach high accuracy. We also observe that while autoencoders we tested can achieve higher accuracy, our method exhibits significantly better efficiency-accuracy trade-offs.

Table 13: Autoencoder Architectures. Decoder is a mirror image of the encoder.

NUMBER	# OF LAYERS IN ENCODER	DIMENSIONALITY OF LAYERS
1	1	$2048 \to 2$
2	1	$2048 \to 1024$
3	2	$2048 \to 512 \to 2$
4	2	$2048 \to 1024 \to 512$
5	4	$2048 \to 512 \to 128 \to 16 \to 2$
6	4	$2048 \to 1024 \to 512 \to 256 \to 128$
7	6	$2048 \to 1024 \to 512 \to 128 \to 64 \to 16 \to 2$
8	6	$2048 \to 1024 \to 512 \to 256 \to 128 \to 64 \to 32$
9	2	$2048 \to 32 \to 2$
10	3	$2048 \to 64 \to 32 \to 4$
11	3	$2048 \to 16 \to 8 \to 4$
12	3	$2048 \to 16 \to 4 \to 2$

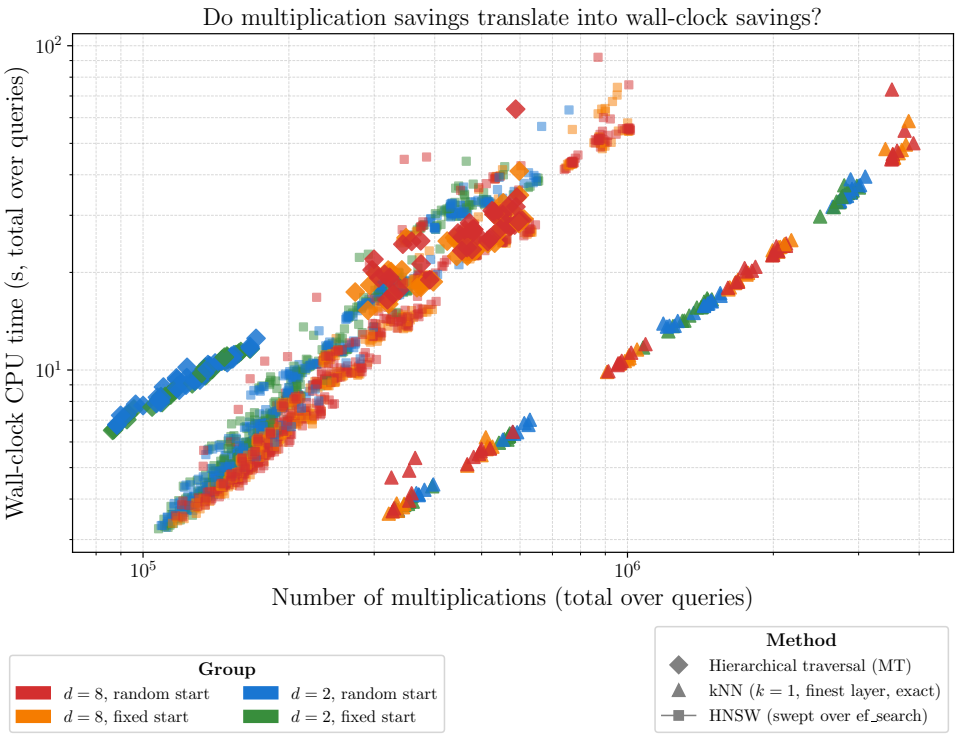


Figure 11: Wall-clock search time versus multiplications for HT, kNN, and HNSW. Index-construction time is excluded.

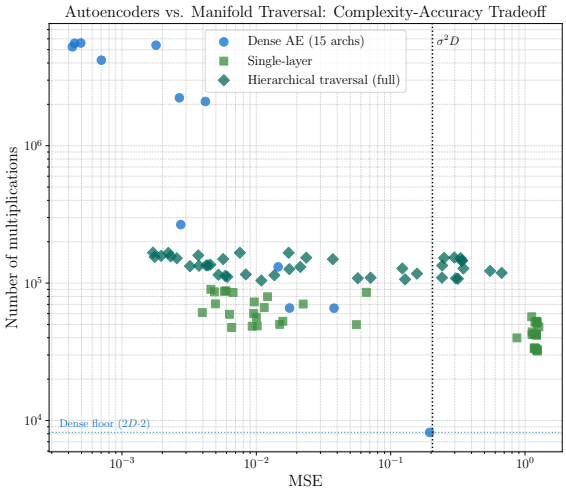


Figure 12: Test-Time Complexity-Accuracy Tradeoff of Manifold Traversal versus Autoencoders.

Across the 12 architectures, test MSE ranges from 4.3×10^{-4} (architecture 4, at 5.2×10^{6} multiplications) to 0.195 (architecture 1, the single layer directly to the intrinsic dimension, at only 8,192 multiplications),

against a noisy-input baseline MSE of 0.205. The cheapest autoencoder architecture reaching accuracy at least as good as our method's best operating point ($\mathrm{MSE} \approx 1.7 \times 10^{-3}$ at 1.7×10^5 multiplications) is architecture 2, which requires 4.2×10^6 multiplications (roughly $25\times$ more) to reach comparable MSE. This gap narrows at lower complexity budgets: at a comparable cost of $\sim 1.3 \times 10^5$ multiplications, architecture 9 reaches $\mathrm{MSE} \approx 1.5 \times 10^{-2}$, somewhat behind our method's $\mathrm{MSE} \approx 1.1 \times 10^{-2}$ at a similar (1.0×10^5) multiplication budget.

D.4 Comparison to Single-Layer Traversal with Random Restarts

A natural question raised by Section 5 is whether HT's advantage over single-layer traversal (SMT) under random initialization is simply a fix for a bad initial guess — one that repeatedly restarting SMT from new random points would achieve for free, without any hierarchical structure at all. We test this directly by running SMT with k independent random restarts per query, keeping the restart that reaches the lowest converged objective and summing multiplications cumulatively across all k restarts.

Setup. We sweep $k \in \{1, 2, 4, 8, 16, 32\}$ restarts against all 224 trained networks from the main sweep, spanning $d \in \{2, 8\}$, R_1-percentile $p \in \{1\%, 2\%, 5\%\}$, and coarseness margin $\rho \in \{0.01, \ldots, 0.30\}$, with $\sigma = 0.01$. No networks were retrained for this comparison. We report, per (d, k), the median MSE ratio $\mathrm{MSE}_{\text{SMT-restart}}/\mathrm{MSE}_{\text{HT}}$ and the median compute ratio (cumulative SMT-restart multiplications across all k restarts, divided by HT's multiplications), each with its winner (ratio > 1 favors HT for accuracy, and favors HT for compute when SMT-restart uses *more* multiplications than HT).

Results. Table 14 reports the full sweep over k. A single random start performs poorly — 80–85$\times$ worse MSE than HT at both intrinsic dimensions — though at only 0.15–0.34$\times$ HT's compute. As k grows, SMT-restart's accuracy improves monotonically, and it does eventually catch up to and surpass HT, but only after roughly 16 restarts at $d = 2$ and roughly 32 restarts at $d = 8$, suggesting that higher-dimensional problems need even more restarts to close the gap.

Table 14: Single-layer traversal with k independent random restarts (SMT-restart) versus HT, median ratios over all 224 configurations, aggregated over R_1-percentile $p \in \{1, 2, 5\}\%$ and coarseness margin $\rho \in \{0.01, \ldots, 0.30\}$. MSE ratio > 1 favors HT; compute ratio > 1 means SMT-restart uses more (cumulative) multiplications than HT.

d	k	MSE ratio (SMT-restart / HT)	Accuracy winner	Compute ratio	Compute winner
2	1	84.53×	HT	0.34×	SMT-restart
2	2	53.31×	HT	0.67×	SMT-restart
2	4	23.72×	HT	1.34×	HT
2	8	6.95×	HT	2.69×	HT
2	16	0.76×	SMT-restart	5.37×	HT
2	32	0.28×	SMT-restart	10.74×	HT
8	1	80.63×	HT	0.15×	SMT-restart
8	2	49.08×	HT	0.30×	SMT-restart
8	4	23.55×	HT	0.59×	SMT-restart
8	8	8.19×	HT	1.19×	HT
8	16	2.10×	HT	2.36×	HT
8	32	0.78×	SMT-restart	4.73×	HT

Interpretation. Critically, SMT-restart's catch-up is not free: by the point it first matches HT's accuracy ($k \approx 4$–8), it has already spent 1.2–2.7$\times$ HT's compute budget, and by $k = 32$ — where it clearly beats HT on accuracy — it costs 4.7–10.7$\times$ more. So HT is not simply patching a bad initialization that brute-force

restarting would fix for free; it reaches comparable-or-better accuracy at a fraction of the cost because it searches the space structurally, coarse-to-fine, rather than by repeated flat guessing.

Summary. Across both intrinsic dimensions, SMT-restart only becomes competitive with HT on accuracy after enough restarts to also exceed HT's compute cost by a wide margin, and the number of restarts required grows with intrinsic dimension. This confirms that HT's random-start robustness (Section 5) reflects a structural property of the coarse-to-fine hierarchy, not a cheaply-replicable side effect of any single initialization.

E Practical Parameter Choices in Experiments

Choosing Layer Count L in Experiments The theoretical lower bound equation 13, while valid, requires estimating $\mathrm{vol}(\mathcal{M})$, $\mathrm{diam}(M)$, and the reach τ, which are not directly accessible from data. To use this formula experimentally, we create a principled and data-driven formula for L and the intermediate R values such that we can compute these from training data and justify with theory.

The hierarchy uses L layers with covering radii $R_1 < R_2 < \cdots < R_L$, where R_1 is the finest (smallest) radius and R_L is the coarsest (largest). We require the radii to form a geometric sequence with decay factor $\lambda \in (0, 1)$, so that $R_i = \lambda^{L-i} R_L$ and in particular $R_1 = \lambda^{L-1} R_L$. The number of landmarks in layer i scales as $n_i \approx \mathrm{vol}(\mathcal{M})/R_i^d$, so the ratio of landmark counts between consecutive layers is $(R_{i+1}/R_i)^d = \lambda^{-d}$. We choose λ such that each coarser layer contains exactly e times fewer landmarks than the layer below it, i.e. $\lambda^{-d} = e$, giving $\lambda = e^{-1/d}$. Substituting this criterion in equation 9, we get the number of layers to be

$$L \;=\; \left\lceil d \ln \frac{R_L}{R_1} \right\rceil + 1. \tag{31}$$

Estimating Coarsest Radius R_L. The coarsest radius R_L must be small enough that the top-layer graph contains at least two landmarks. Because landmark construction is greedy and the first training point always becomes landmark $\boldsymbol{q}_0$, a second landmark forms if and only if R_L is strictly less than the eccentricity of $\boldsymbol{q}_0$, where the eccentricity of a point $\boldsymbol{x}$ is $e(\boldsymbol{x}) = \max_{\boldsymbol{x}' \in \mathbf{X}} |\boldsymbol{x}' - \boldsymbol{x}|$. To guarantee at least two landmarks regardless of which point becomes $\boldsymbol{q}_0$, we set R_L below the minimum eccentricity over the training set:

$$R_\star = \min_{\boldsymbol{x} \in \mathbf{X}} \max_{\boldsymbol{x}' \in \mathbf{X}} \|\boldsymbol{x}' - \boldsymbol{x}\|. \tag{32}$$

Geometrically, $R_\star$ is the radius of the smallest enclosing ball whose center is constrained to lie in $\mathbf{X}$; no single training point can cover the entire dataset within radius $R_\star$, so any threshold $R_L < R_\star$ forces at least two landmarks to form. We approximate $R_\star$ by sampling $K = 20$ anchor points uniformly at random from $\mathbf{X}$, computing the eccentricity of each against a subsample of $M = 10{,}000$ points, and taking the minimum:

$$\widehat{R}_\star = \min_{k=1,\ldots,K} \max_{j=1,\ldots,M} \|\boldsymbol{x}^{(j)} - \boldsymbol{x}^{(k)}\|. \tag{33}$$

We then set $R_L = (1 - \rho)\widehat{R}_\star$, where $\rho \in (0, 1)$ is a margin hyperparameter. Larger ρ yields a smaller R_L, a smaller ratio R_L/r_1, and hence fewer layers; beyond a threshold, the hierarchy becomes too shallow to provide meaningful basin-finding, and performance degrades.

In the main experiments, described in Section 5 we fix R_1 at the 1% percentile of pairwise distances and restrict to $\rho \in \{0.01, 0.02, 0.05, 0.1\}$, the regime in which the hierarchy is sufficiently deep for all settings. On the gravitational-wave dataset ($D = 2048$, unit-norm signals) this produces $R_L \in [1.80, 1.98]$

for $d = 8$ and $R_L \approx 1.46$–1.65 for $d = 2$, with $R_1 \approx 0.27$ and $R_1 \approx 0.07$ respectively. In additional experiments shown in Appendix Section C we additionally sweep $R_1 \in \{1\%, 2\%, 5\%\}$ and $\rho \in \{0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$ to characterize sensitivity across the full range, including the degraded regime.

Estimating Finest Radius R_1. The finest-layer radius R_1 is chosen in a data-driven manner as a low percentile of the empirical pairwise-distance distribution. Concretely, we draw a random subsample of $N' = 1{,}000$ training points, compute all $\binom{N'}{2}$ pairwise Euclidean distances, and set r_1 to the p-th percentile of that distribution. Placing R_1 in the lower tail of the pairwise distances ensures that the overwhelming majority of point pairs lie farther apart than R_1, so that each data point either forms its own landmark or falls within the reach of a nearby one — producing a dense, fine-grained graph at the lowest layer of the hierarchy. In all main experiments we use $p = 1\%$, and we provide a sensitivity analysis over $p \in \{1\%, 2\%, 5\%\}$ in Appendix Section C alongside the sensitivity analysis over ρ.

Algorithm 3 `BuildCrossLayerEdges`$(\{\mathcal{G}_l\}_{l=1}^{L}, R_\otimes, \{V_l\}_{l=1}^{L})$

Input: Trained networks $\{\mathcal{G}_l\}_{l=1}^{L}$, radius $R_\otimes$, sets of landmark indices $\{V_l\}_{l=1}^{L}$.
for $l = L$ **downto** 2 **do**
  for each $\boldsymbol{q}_i^{(l)} \in \mathcal{Q}_l$ **do**
    $\mathcal{C}_i^{(l)} \leftarrow \{j \in V_{l-1} : \|\boldsymbol{q}_i^{(l)} - \boldsymbol{q}_j^{(l-1)}\|_2 \le R_\otimes\}$ ▷ A set of indices in the next layer
    if $\mathcal{C}_i^{(l)} = \emptyset$ **then** ▷ If there are no nearby landmarks in the next layer
      $\mathcal{C}_i^{(l)} \leftarrow \{\arg\min_{j \in V_{l-1}} \|\boldsymbol{q}_i^{(l)} - \boldsymbol{q}_j^{(l-1)}\|_2\}$ ▷ guarantee connectivity via exhaustive search
    end if
    $\pi^{(l)}(i) \leftarrow \arg\min_{j \in \mathcal{C}_i^{(l)}} \|\boldsymbol{q}_i^{(l)} - \boldsymbol{q}_j^{(l-1)}\|_2$ ▷ prune to single nearest target
    Add cross-layer edge $i \xrightarrow{\otimes} \pi^{(l)}(i)$
  end for
end for
Output: Cross-layer edge maps $\{\pi^{(l)} : V_l \to V_{l-1}\}_{l=2}^{L}$.

Cross-Layer Edges and Radius $R_\otimes$. After graphs $\mathcal{G}_l$ in each layer $l = 1, \ldots, L$ are trained independently via the online graph construction procedure 1 of **??**, a post-training pass constructs a set of directed cross-layer edges that link adjacent layers and enable coarse-to-fine traversal at query time, as described in Algorithm 3. For each pair of adjacent layers $(l, l-1)$ and each landmark $\boldsymbol{q}_i^{(l)} \in \mathcal{Q}_l$, a cross-layer edge $i \xrightarrow{\otimes} j$ is added for every landmark $\boldsymbol{q}_j^{(l-1)} \in \mathcal{Q}_{l-1}$ satisfying

$$\|\boldsymbol{q}_i^{(l)} - \boldsymbol{q}_j^{(l-1)}\| \le R_\otimes. \tag{34}$$

The parameter $R_\otimes$ decides inter-layer connectivity: landmarks in consecutive layers that are geometrically proximate in $\mathbb{R}^D$ are wired together, while distant pairs are not. Cross-layer edges are directed from coarser to finer layers only; no reverse edges are introduced.

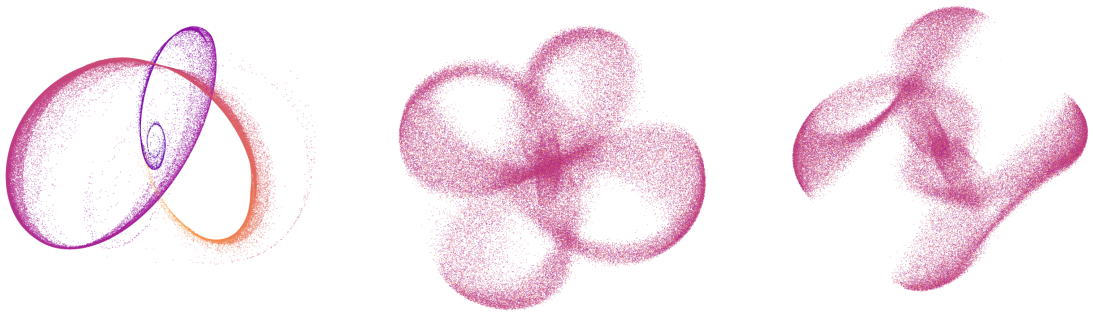
To guarantee that every non-finest-layer landmark has at least one outgoing cross-layer edge, any landmark with no candidate within $R_\otimes$ is connected by exhaustive search to its nearest landmark in the adjacent finer layer. This fallback prevents traversal from stalling before reaching the bottom layer. Each landmark's candidate list is then pruned to a single edge, retaining the nearest target in a Euclidean sense, so that every landmark $\boldsymbol{q}_i^{(l)}$ carries exactly one pointer to a landmark $\boldsymbol{q}_{\pi(i)}^{(l-1)}$ in the next layer.

Formally, the cross-layer pointer map $\pi^{(l)} : V_l \to V_{l-1}$ is defined as

$$\pi^{(l)}(i) = \arg\min_{j \in \mathcal{C}_i^{l-1}} \left\| \boldsymbol{q}_i^{(l)} - \boldsymbol{q}_j^{(l-1)} \right\|_2^2, \tag{35}$$

where $\mathcal{C}_i^{l-1} \subseteq V_{l-1}$ denotes the set of candidates within $R_\otimes$ of $\boldsymbol{q}_i^{(l)}$, defaulting to the singleton nearest neighbor when no candidate exists. At query time, $\pi^{(l)}(i)$ identifies the unique landmark in layer $l-1$ to which traversal jumps unconditionally upon converging at landmark i in layer l. The full construction is given in Algorithm 3.

At query time, traversal proceeds in alternating within-layer and cross-layer phases. Within each layer l, the standard mixed-order traversal (first-order, then zero-order steps) runs to local convergence. Upon convergence, the algorithm unconditionally follows the single cross-layer edge at the current landmark, landing at $\boldsymbol{q}_{\pi(i)}^{(l+1)}$ without evaluating any distance criterion. Within-layer traversal then restarts from this landing point in layer $l+1$. The process terminates upon convergence in the finest layer, which carries no outgoing cross-layer edges. The net effect is a hierarchical funnel: coarse layers rapidly localize the query to the correct region of the manifold, and the finest layer provides the precise denoising landmark.



(a) (b) (c)

Figure 13: PCA visualizations of gravitational wave data for intrinsic dimension **(a)** $d = 2$, and **(b)-(c)** $d = 8$.

F Background: Covering Numbers on Manifolds

In this section, we define covering numbers on manifolds and establish bounds on this quantity. We follow Definitions 4.2.1 and 4.2.2 of Vershynin (2018), adapted to submanifolds of $\mathbb{R}^D$ with Euclidean balls.

Definition F.1 (ε-net)**.** Let $\mathcal{M} \subset \mathbb{R}^D$ be a compact smooth d-dimensional submanifold and let $\varepsilon > 0$. A subset $\mathcal{N} \subset \mathcal{M}$ is called an *ε-net of $\mathcal{M}$* if every point of $\mathcal{M}$ lies within Euclidean distance ε of some point of $\mathcal{N}$, i.e.,

$$\forall\, \boldsymbol{x} \in \mathcal{M}, \quad \exists\, \boldsymbol{x}_0 \in \mathcal{N} \ \text{ such that } \ \|\boldsymbol{x} - \boldsymbol{x}_0\| \le \varepsilon.$$

Definition F.2 (ε-covering number)**.** The *covering number* of $\mathcal{M}$ at scale ε, denoted $\mathcal{N}(\mathcal{M}, \varepsilon)$, is the smallest cardinality of any ε-net of $\mathcal{M}$:

$$\mathcal{N}(\mathcal{M}, \varepsilon) \;=\; \inf\left\{ n \in \mathbb{N} \;:\; \exists \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subseteq \mathcal{M} \ \text{ s.t. } \ \mathcal{M} \subseteq \bigcup_{i=1}^{n} B(\boldsymbol{x}_i, \varepsilon) \right\}, \tag{36}$$

where $B(\boldsymbol{x}_i, \varepsilon) = \{\mathbf{y} \in \mathbb{R}^D : \|\boldsymbol{x}_i - \mathbf{y}\| \le \varepsilon\}$ is the closed Euclidean ball of radius ε centered at $\boldsymbol{x}_i$.

This Euclidean notion of covering is the one directly implemented by Algorithm 1, which evaluates $\|\boldsymbol{q}_i - \boldsymbol{x}\|$ to determine whether a training point is already covered.

These are used to derive a lower bound on the finest covering radius R_1 in section B.2.

G Computational Complexity Calculations

We measure the computational cost of single-layer traversal and the proposed Hierarchical Manifold Traversal Network (HMTN) in terms of the number of scalar multiplications per query. This hardware-independent characterization of arithmetic work allows a direct, apples-to-apples comparison between the two approaches.

Single-layer traversal: a complexity baseline. We first establish the complexity of the single-layer traversal network (SMT), which serves as the baseline for comparison. Theorem 7.1 of Wang et al. (2025) bounds the total number of arithmetic operations as

$$O\left(\left(D + e^{c' \log(a)\, d}\right)\left(\frac{\mathrm{diam}^2(\mathcal{M})}{\epsilon_1 \delta} + \frac{1}{\kappa\delta}\right) d + D \cdot \$^{\eta}_{R_1}(\mathcal{M})\right). \tag{37}$$

To interpret each term: the factor $(D + e^{c' \log(a) d})d$ is the per-step cost of a first-order traversal step, where Dd comes from projecting the D-dimensional gradient into the d-dimensional tangent space, and $de^{c' \log(a) d}$ accounts for comparing the gradient against all first-order edge embeddings. The factor $(\mathrm{diam}^2(\mathcal{M})/\epsilon_1\delta + 1/\kappa\delta)$, with δ as landmark spacing, counts the total number of first-order steps: the diam^2 term covers the initial descent phase from a potentially distant starting point, and the $1/\kappa\delta$ term covers the fine-grained convergence phase near the target. Finally, $\$^{\eta}_{R_1}(\mathcal{M})$ is the *vista number*, which bounds the number of zero-order edges evaluated per landmark, each costing D multiplications.

Here and throughout, $A \lesssim B$ denotes $A \leq CB$ for an absolute constant $C > 0$.

Specializing to the single-layer setting with landmark spacing $R_1 = \lambda^{L-1} R_L$ (where R_L is the coarsest-layer spacing and $\lambda \in (0, 1)$ is the inter-layer scale factor), we obtain

$$\begin{aligned} M_{\text{SMT}} &\lesssim d\left(D + e^{c' \log(a) \cdot d}\right)\left[\frac{c\lambda^{-2} R_1^2}{\epsilon_1 R_1} + \frac{1}{\kappa R_1}\right] + D\$^{\eta}_{c_6 \tau_{\mathcal{M}}} && (38) \\ &= d\left(D + e^{c' \log(a) \cdot d}\right)\left[\frac{c\lambda^{-2}\lambda^{L-1} R_L}{\epsilon_1} + \frac{1}{\kappa \lambda^{L-1} R_L}\right] + D\$^{\eta}_{c_6 \tau_{\mathcal{M}}} && (39) \end{aligned}$$

$$\boxed{M_{\text{SMT}} \lesssim d\Big(D + e^{c' \log(a) \cdot d}\Big)\left[\lambda^{L-1}\frac{c\lambda^{-2} R_L}{\epsilon_1} + \frac{1}{\kappa \lambda^{L-1} R_L}\right] + D\,\$^{\eta}_{R_1}.} \tag{40}$$

Hierarchical traversal replaces diameter with warm-start radius. The central idea behind HMT is straightforward: after converging in layer l and hopping down to layer $l-1$, the algorithm is expected to start within $O(R_l)$ of the query's nearest landmark in the finer layer. This *warm start* means that traversal in layer $l-1$ need only cover $O(R_l) = O(\lambda^{-1} R_{l-1})$ distance, rather than the full $\mathrm{diam}^2(\mathcal{M})$ that a cold-start traversal would require.

Concretely, for a landmark $\mathbf{q}_{i_0}$ at the start of single-layer traversal, the worst-case initialization satisfies $\|\mathbf{q}_{i_0} - \mathbf{y}\|_2^2 \leq \|\mathbf{q}_{i_0} - \mathbf{x}\|_2^2 + \|\mathbf{z}\|_2^2 \leq \mathrm{diam}^2(\mathcal{M}) + \|\mathbf{z}\|_2^2$, as $\mathbf{q}_{i_0}$ and $\mathbf{x}$ lie on $\mathcal{M}$. In the hierarchical setting,

descent from layer l replaces $\mathrm{diam}^2(\mathcal{M})$ with $O(R_l^2)$ in the complexity bound for layer $l-1$. Simultaneously, the landmark spacing δ in Theorem 7.1 is replaced by R_l at each layer. This double substitution – replacing both the diameter term and the spacing – drives the complexity calculation of HMT.

HMT complexity derivation. Applying the substitutions described above to each layer and summing, the total number of scalar multiplications in HMT is

$$M_{\mathrm{HMT}} \lesssim \sum_{l=1}^{L} \left[\left(D + e^{c' \log(a) \cdot d} \right) \left(\frac{c\lambda^{-2} R_l^2}{\epsilon_1 R_l} + \frac{1}{\kappa R_l} \right) d + D \, \$_{R_l}^{\eta(l)} \right] + \underbrace{(L-1)D}_{\text{cross-layer hops}} . \tag{41}$$

We evaluate the layer sum by substituting $R_l = \lambda^{L-l} R_L$ and separating the two first-order terms:

$$M' = d\left(D + e^{c' \log(a) \cdot d} \right) \left[\frac{c\lambda^{L-2} R_L}{\epsilon_1} \underbrace{\sum_{l=1}^{L} \lambda^{-l}}_{S_1} + \frac{1}{\kappa \lambda^L R_L} \underbrace{\sum_{l=1}^{L} \lambda^l}_{S_2} \right] + \sum_{l=1}^{L} D \, \$_{R_l}^{\eta(l)}. \tag{42}$$

Both S_1 and S_2 are finite geometric series. Using the standard formula $\sum_{k=0}^{N-1} r^k = (1-r^N)/(1-r)$, we obtain

$$S_1 = \sum_{l=1}^{L} \lambda^{-l} = \frac{1-\lambda^{-L}}{1-\lambda^{-1}} \cdot \lambda^{-1} = \frac{1-\lambda^L}{\lambda^L (1-\lambda)}, \qquad S_2 = \sum_{l=1}^{L} \lambda^l = \frac{\lambda(1-\lambda^L)}{1-\lambda}. \tag{43}$$

Substituting these closed forms into Equation (42) and simplifying, the factor $(1-\lambda^L)/(1-\lambda)$ emerges as a common prefactor:

$$M' = d\left(D + e^{c' \log(a) \cdot d} \right) \left(\frac{1-\lambda^L}{1-\lambda} \right) \left[\frac{c\lambda^{-2} R_L}{\epsilon_1} + \frac{1}{\kappa \lambda^{L-1} R_L} \right] + \sum_{l=1}^{L} D \, \$_{R_l}^{\eta(l)}. \tag{44}$$

Adding back the $(L-1)D$ cost of cross-layer objective recomputation, the total is

$$\boxed{M_{\mathrm{HMT}} \lesssim d\left(D + e^{c' \log(a) \cdot d} \right) \left(\frac{1-\lambda^L}{1-\lambda} \right) \left[\frac{c\lambda^{-2} R_L}{\epsilon_1} + \frac{1}{\kappa \lambda^{L-1} R_L} \right] + \sum_{l=1}^{L} D \, \$_{R_l}^{\eta(l)} + (L-1)D.} \tag{45}$$

Comparison and discussion. Equations equation 45 and equation 40 differ in precisely the terms highlighted in blue. With $\lambda \in (0,1)$ and $R_1 = \lambda^{L-1} R_L$, the finest layer has the smallest landmark spacing and the coarsest layer the largest. In the first-order phase, SMT carries a prefactor of $\lambda^{L-1} \in (0,1)$ on the term $c\lambda^{-2} R_L/\epsilon_1$, whereas HMT carries the factor $(1-\lambda^L)/(1-\lambda) = \sum_{k=0}^{L-1} \lambda^k$, which lies strictly between 1 and L. Consequently, HMT may incur more first-order steps than SMT by a factor of $c_0(1-\lambda^L)/((1-\lambda))$ for some constant c_0. For example, with $\lambda = e^{-1/d}$, $d = 2$, and $L = 8$, we have $\lambda^{L-1} = e^{-7/2} \approx 0.03$, while $(1-\lambda^L)/(1-\lambda) \approx 2.5$. This overhead in first-order steps is the price HMT pays for its coarse-to-fine structure.

The zero-order cost $\sum_{l=1}^{L} D \, \$_{R_l}^{\eta(l)}$ in HMT is distributed across all L layers. Since $\$_{R_l}^{\eta}(\mathcal{M})$ is a covering number at radius R_l, it decreases monotonically as R_l increases, meaning coarser layers on average contribute substantially smaller zero-order costs than the finest layer. In contrast, SMT evaluates zero-order

edges exclusively at radius R_1, the finest and most expensive scale, across the entire traversal path. The net computational advantage of HMT therefore depends on whether the reduction in zero-order cost across coarser layers outweighs the additional first-order steps, a trade-off governed by the geometry of $\mathcal{M}$ and the vista numbers $\$_{R_l}^{\eta(l)}(\mathcal{M})$.

The `eHMT` variant. In the variant `eHMT` the objective value is not recomputed after each cross-layer hop; instead, the converged value from the coarser layer is carried forward as the baseline for within-layer traversal in the finer layer. This is justified when cross-layer pointers connect landmarks that are spatially close, making the change in φ negligible. The effect is to eliminate the $(L-1)D$ term entirely, reducing the total cost to

$$M_{\mathrm{eHMT}} \lesssim d\Big(D + e^{c' \log(a)\cdot d}\Big)\left(\frac{1-\lambda^L}{1-\lambda}\right)\left[\frac{c\lambda^{-2}R_L}{\epsilon_1} + \frac{1}{\kappa\lambda^{L-1}R_L}\right] + \sum_{l=1}^{L} D\,\$_{R_l}^{\eta(l)}. \tag{46}$$

Since $(L-1)D$ is linear in both L and D and grows with the number of hierarchical layers, `eHMT` is particularly beneficial in high-dimensional settings with many layers.

H Compute Resources

Compute resources. The computational experiments were performed on a Linux-based workstation powered by an Intel Core i9-10900X CPU (20 cores, 251GB RAM, Cascade Lake architecture). Training and testing seven methods shown on gravitational wave datasets takes under 1 hour.

Beyond the multiplication-count accounting used throughout this section, Table 15 reports wall-clock memory and timing on a real trained hierarchy ($D = 2048$), at the two intrinsic dimensions used in our main sweep. HMTN's per-query inference stays sub-3-ms at both $d = 2$ and $d = 8$, at 2-4$\times$ the single-layer method's memory footprint. Training is a one-time, CPU-only cost dominated by per-layer graph construction, which scales with sample count; cross-layer map construction, the structural addition unique to the hierarchy, adds only 11–14 seconds regardless of d, since it scales with landmark count rather than sample count. Training remains under an hour for the gravitational-wave datasets used here. On the much larger CIFAR-10 and CelebA patch sets used in Section C.3 (13.1M and 11.2M patches, respectively), training takes just over 11 hours, still entirely CPU-side and one-time.

Table 15: Wall-clock memory and timing for a trained HMTN hierarchy ($D = 2048$, Intel i9-10900X, 20 cores, 251GB RAM), at the two intrinsic dimensions used in the main gravitational-wave sweep.

Metric	$d = 2$ ($L = 8$)	$d = 8$ ($L = 17$)
Finest-layer landmarks	1,374	1,002
Single-layer memory	67.9 MB	149.3 MB
Full hierarchy memory	139.3 MB	578.2 MB
Memory overhead vs. single-layer	2.05$\times$	3.87$\times$
Inference (10,000 queries)	$\approx$12.4 s	$\approx$29.2 s
Per-query inference	1.24 ms	2.92 ms
Graph construction (per-layer, approx.)	$\approx$9.0 min	$\approx$31.6 min
Cross-layer map construction (exact)	11.2 s	13.6 s

Algorithm 4 `GraphConstruction`($\mathcal{G}^{(l)}, \boldsymbol{x}, R_{\text{nbrs}}$)

Input: Networks $\{\mathcal{G}^{(l)}\}_{l=1}^{L}$ with radii $R_1 < \cdots < R_L$, clean sample $\boldsymbol{x} \in \mathbb{R}^D$, radius R_{nbrs}
for $l = 1$ **to** L **do**
  $i \leftarrow$ `HierarchicalTraversal`$(\mathcal{G}_l, \boldsymbol{x})$
  if $\|\boldsymbol{q}_i - \boldsymbol{x}\| \le R_l$ **then** ▷ If clean point $\boldsymbol{x}$ is within R_l of landmark $\boldsymbol{q}_i$, update the local model
    `UpdateLocalModel` of $\boldsymbol{U}_i^{(l)}$, $\Xi_i^{(l)}$ at $\boldsymbol{q}_i$ with $\boldsymbol{x}$
  else
    $i_\star \leftarrow \arg\min_j \|\boldsymbol{q}_j - \boldsymbol{x}\|$ ▷ If clean point $\boldsymbol{x}$ is not within R_l distance of landmark $\boldsymbol{q}_i$, resort to a linear scan
    if $\|\boldsymbol{q}_{i_\star} - \boldsymbol{x}\| \le R_l$ **then** ▷ If the linear scan found a landmark such that the point $\boldsymbol{x}$ is within R_l of that landmark
      $\boldsymbol{q}_i^{(l)}$ is a local min, add $i \xrightarrow{0} i_\star$ ▷ Add a zero-order edge
      `UpdateLocalModel` (i.e. $\boldsymbol{U}_{i_\star}^{(l)}$, $\Xi_{i_\star}^{(l)}$) at $\boldsymbol{q}_{i_\star}^{(l)}$ with $\boldsymbol{x}$
    else
      `InitLandmark` Create a new landmark $\boldsymbol{q}_M \leftarrow \boldsymbol{x}$
      Connect $\boldsymbol{q}_M$ to neighboring landmarks within R_{nbrs}, initialize $\boldsymbol{U}_M^{(l)}$, $\Xi_M^{(l)}$
    end if
  end if
end for
Output: $\{\mathcal{G}_l\}_{l=1}^{L} = (V^{(l)}, \mathcal{E}^{(l)})$ ▷ The graph is a collection of vertices V and edges $\mathcal{E}$.

I ALGORITHMS WITH DETAILED DESCRIPTIONS

Algorithm 1 and Algorithm 2 in the main body of the paper describe hierarchical graph construction and traversal, respectively. Here, we provide line-by-line explanations for these procedures in Algorithm 4 and Algorithm 5, respectively.

Algorithm 5 `HierarchicalTraversal`$(\mathcal{G}_l, \boldsymbol{y})$

Input: Noisy test sample $\boldsymbol{y} \in \mathbb{R}^D$, $\{G^{(l)}\}_{l=1}^{L}$, $l \leftarrow L$, $i \leftarrow 1$ or uniformly at random.
$\varphi \leftarrow \|\boldsymbol{q}_i^{(L)} - \boldsymbol{y}\|^2$ ▷ Compute the objective at the very start.
while not globally converged **do**
  while not layer-l converged **do** ▷ Loop through each layer
    $\mathbf{g} \leftarrow \boldsymbol{U}_i^{(l)*}(\boldsymbol{q}_i^{(l)} - \boldsymbol{y})$ ▷ Compute the gradient, then identify the first-order neighbor whose
    $i^\sharp \leftarrow \arg\max_{j:\, i \xrightarrow{1} j} \langle -\mathbf{g}, \boldsymbol{\xi}_{ij}^{(l)} \rangle$ ▷ edge embedding is most aligned with negative gradient.
    if $\|\boldsymbol{q}_{i^\sharp}^{(l)} - \boldsymbol{y}\|^2 < \varphi$ **then** ▷ If moving to that first-order neighbor improves the objective,
      $i \leftarrow i^\sharp, \quad \varphi \leftarrow \|\boldsymbol{q}_i^{(l)} - \boldsymbol{y}\|^2$ ▷ move there and update i and objective
    else ▷ Otherwise do a linear scan across zero-order edges.
      $i^\flat \leftarrow \arg\min_{j:\, i \xrightarrow{0} j} \|\boldsymbol{q}_j^{(l)} - \boldsymbol{y}\|^2$ ▷ Identify the index of the best zero-order neighbor.
      if $\|\boldsymbol{q}_{i^\flat}^{(l)} - \boldsymbol{y}\|^2 < \varphi$ **then** ▷ If moving to zero-order neighbor improves objective,
        $i \leftarrow i^\flat, \quad \varphi \leftarrow \|\boldsymbol{q}_i^{(l)} - \boldsymbol{y}\|^2$ ▷ Move there are update i and φ.
      else
        layer-l converged ▷ Otherwise, we have converged in the current layer.
      end if
    end if
  end while
  if pointer $\pi^{(l)}(i)$ exists **then** ▷ Upon converting in the current layer, we hop to the next one.
    Hop down a layer via $i \leftarrow \pi^{(l)}(i)$, update $\quad l \leftarrow l - 1, \quad \varphi \leftarrow \|\boldsymbol{q}_i^{(l)} - \boldsymbol{y}\|^2$
  else ▷ If there are no cross-layer edges present,
    globally converged ▷ we have reached the bottom-most layer.
  end if
end while
Output: i