Associative Memory for Hyperdimensional Spatial Representations with Geodesic Flow Matching

by

Karim Habashy

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2026

© Karim Habashy 2026

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The ability to recover clean patterns from noisy or partial inputs, known as Associative memory, is a cornerstone of robust computation in both biological and artificial intelligence systems. While well-established for discrete data, implementing this capability for **continuous** representations remains a challenge. Existing methods typically rely on discretizing the continuous domain into capacity-limited prototypes or performing explicit decoding to the spatial domain, which is often computationally expensive and biologically implausible.

This thesis addresses this gap by reformulating cleanup as a **generative transport problem** entirely within the embedding space. **Geodesic Flow Matching** is introduced as a model that learns a continuous time-dependent velocity field to transport corrupted representations back to the valid data manifold. Standard Euclidean Flow matching is shown to be insufficient for high-dimensional normalized representations, as linear interpolants "cut through" the interior of the hypersphere, destroying the vector magnitude and phase relationships required for accurate decoding. By constraining transport dynamics to the intrinsic Riemannian geometry of the hypersphere, the proposed model preserves this structure even under severe corruption.

The framework is validated using **Spatial Semantic Pointers (SSPs)**, a biologically plausible encoding for continuous space. Benchmarks indicate that Geodesic Flow consistently outperforms Euclidean variants and classical baselines, particularly in high-noise regimes. The utility of the approach is further demonstrated through integration into a Spiking Neural SLAM system. As an online stabilizer for the path integrator, the Geodesic model prevents catastrophic drift, reducing path error by up to 72%. Furthermore, it significantly improves resource efficiency by 40%, allowing a neural population of 1,500 neurons to match the tracking accuracy of a baseline system using 2,500 neurons.

Acknowledgements

First and foremost, I would like to thank Chris. I knew I wanted to be part of this lab from the very first lecture of yours I attended. I am deeply grateful for your guidance, presence, and openness. Michael, I am grateful for your mentorship, and for pushing me to always be clear and precise in my thinking. Thank you for listening when I fell out of love with research, and when I fell back in love with it. To everyone in the lab: thank you for the camaraderie, the feedback, and for your patience in listening to me ramble endlessly about flow matching.

To my family, thank you for your unwavering support in anything I set my mind to. You instilled in me the courage to think for myself and the freedom to pursue what *I* truly want in life.

Finally, thank you to coffee for grounding me, and to running for the momentum you provided.



Dedication

To my grandfather. For the hand that held mine, and the rhythm of your feet against the asphalt.

Table of Contents

Author's Declaration ii

Abstract iii

Acknowledgements iv

Dedication v

List of Figures ix

List of Tables xii

1 Introduction 1

1.1 The Problem: Cleanup in Continuous Domains 2

1.2 Proposed Solution: Geodesic Flow Matching 3

2 Background 5

2.1 Vector Symbolic Algebras 5

2.1.1 Holographic Reduced Representations 6

2.2 Spatial Semantic Pointers 7

2.3 Neural Implementation 9

2.4 The Problem of Corrupted Representations 10

2.5 Current Solutions . . . 12
2.5.1 Normalization . . . 13
2.5.2 Grid Lookup . . . 13
2.5.3 Associative Memories . . . 13
2.5.4 Attractor Networks . . . 14
2.5.5 Optimization-Based Decoding . . . 15
2.5.6 Phase-Coupled Optimization . . . 15
2.5.7 Generative Cleanup Models . . . 16

3 Method **18**
3.1 Setup and Notation . . . 19
3.2 Transport Dynamics and Continuity Constraint . . . 19
3.3 Coupling distributions for CFM . . . 20
3.3.1 Independent Coupling . . . 20
3.3.2 Exact Optimal-Transport Coupling . . . 21
3.3.3 Entropy-Regularized Coupling . . . 21
3.4 Direct Mapping: The Feedforward Baseline . . . 23
3.5 Iterative Transport: Flow-Matching Families . . . 23
3.5.1 Euclidean Flow Matching . . . 24
3.5.2 Geodesic CFM . . . 25
3.6 Experimental Setup and Method Variations . . . 29
3.6.1 Model Architecture . . . 29
3.6.2 Time Conditioning and Unknown Corruption Levels . . . 30
3.6.3 OT and SB Coupling Solvers. . . . 30
3.7 Baselines . . . 30
3.7.1 Grid and Optimization-Based Cleanup . . . 31
3.8 Inference Time and Memory Complexity . . . 32
3.9 Summary . . . 33

4 Benchmarks and Results **35**
4.1 Experimental Setup . . . 35
4.2 Main Results: SSP restoration under noise . . . 35
4.2.1 General Trends . . . 36
4.2.2 Baseline Cleanup performance . . . 39
4.2.3 Performance Relative to Baselines . . . 41
4.3 Effect of Step Count . . . 43
4.4 Compute and Memory Requirements . . . 45
4.4.1 Memory footprint . . . 45
4.4.2 Inference runtime . . . 45
4.5 Hyperparameter Sensitivity . . . 46
4.5.1 Effect of Stochasticity and Coupling Distributions . . . 46
4.5.2 Effect of OT vs SB couplings . . . 47
4.5.3 Feedforward Model Sensitivity . . . 48
4.6 Effect of Length Scale . . . 49

5 Cleanup-Augmented SLAM: An Application of Geodesic Associative Memories **52**
5.1 The Algorithm . . . 53
5.2 Cleanup Usage within SSP-SLAM . . . 54
5.2.1 Online Path Integrator Stabilization . . . 54
5.3 Implementation details . . . 55
5.3.1 Environment and Task . . . 55
5.4 Results . . . 56
5.4.1 Localization Accuracy . . . 56
5.4.2 Map Accuracy and Efficiency Analysis . . . 59

6 Conclusion and Future Work **62**

References **64**

List of Figures

2.1 Spatial Semantic Pointer (SSP) representations ($d = 487$). (a) The 1D SSP sinc kernel encoding a scalar variable (various length scales displayed). (b) The corresponding 2D SSP representation illustrating the spatial interference pattern. 8

2.2 Conditional paths generated by flow matching models vs diffusion 17

3.1 Visualization of coupling strategies on a toy 2D dataset [48]. **(Left)** Independent coupling pairs source noise (black) with target data (blue) randomly, creating a disordered vector field with many intersecting paths. **(Center)** Exact Optimal Transport (OT) coupling assigns pairs based on geometric proximity, resulting in straight, non-intersecting trajectories that minimize total displacement. **(Right)** Entropy-regularized coupling (Schrödinger Bridge) relaxes the strict determinism of OT, producing a probabilistic mapping where mass from one source point is diffused across multiple nearby target points. 22

4.1 Mean cosine similarity ± 95% CI across noise levels. Higher dimensions (right) sustain performance at higher noise levels compared to lower dimensions (left). The Geodesic Flow model is more noise robust than other alternatives, including non-neural baselines. (Note: Error bars/shaded regions are present but may be smaller than the plot lines due to low variance.) . . 36

4.2 Performance benefits of Geodesic interpolants relative to Euclidean (Δ mean cosine ± 95% CI) vs dimensionality. 38

4.3 Qualitative comparison of inference outputs. (a) and (b) show the iterative evolution of the flow models. Note that the Geodesic Flow (a) maintains a sharp peak centered on the target, while the Euclidean Flow (b) drifts. (c) The ground truth target. (d) The direct feedforward prediction, which exhibits a diffuse similarity map due to the averaging effects of the regression objective. 39

4.4 Performance over different resolutions for grid and optimization baselines. . 41

4.5 Mean cosine similarity ± 95% CI across Noise levels averaged over dimensions showing performance relative to grid cleanup baseline. 42

4.6 Similarity map for a feedforward model trained with random couplings. The model collapses to an "average" SSP-like vector, failing to recover the correct location. 43

4.7 Average cosine similarity (±95% CI) as a function of inference step count K, averaged over noise levels. Geodesic flows benefit from multi-step integration, while Euclidean flows do not. 44

4.8 Computational scaling. (a) Neural models scale similarly to the grid baseline in memory. (b) Optimization-based cleanup (red) incurs significantly higher runtime costs as dimensionality increases, while feedforward and flow models (mostly) remain relatively efficient. 46

4.9 Effect of different sigma values between 0.01 and 10 on model performance. Introducing stochasticity degrades the structure of the target SSP 48

4.10 Relative performance benefits of sampling methods relative to deterministic Euclidean trajectories (Δ mean cosine ± 95% CI) vs dimensionality. 49

4.11 Relative performance benefits of on/off manifold sampling methods relative to deterministic Geodesic trajectories (Δ mean cosine ± 95% CI) vs dimensionality. 50

4.12 Relative performance benefits of coupling structure to random coupling on feedforward model performance (Δ mean cosine ± 95% CI) vs dimensionality. 51

4.13 Average performance over length scales. 51

5.1 Trajectory reconstruction for a resource-constrained Path Integrator ($N = 1500$ neurons). (a) The Grid Baseline fails to correct phase drift, resulting in catastrophic divergence where the agent's internal estimate (blue) completely detaches from the ground truth (dashed grey) after complex turns. (b) The Geodesic Flow model acts as a continuous attractor, successfully projecting the noisy state back to the valid manifold and maintaining a stable lock on the trajectory (red) throughout the simulation. 57

5.2 Average aggregate error for 5 trajectories given 1500 neuron population. . . 58

5.3 Localization performance at $N = 3000$ neurons. (a) Sample trajectory with PI results given 3000 neuron population. (b) Average aggregate error for 5 trajectories given 3000 neuron population. At high capacities, intrinsic stability is sufficient, but Flow cleanup still reduces variance. 59

List of Tables

3.1 Asymptotic inference-time and memory complexity of cleanup methods. . . 33

3.2 Summary of model variants and configuration choices. 34

5.1 Comparison of Landmark and Trajectory errors across Path Integrator sizes. **Landmark RMSE** measures the geometric accuracy of the learned map, while **Path Error** measures the fidelity of the agent's localization. Bold indicates best performance. 60

Chapter 1

Introduction

Robust cognition requires the ability to recover clean patterns from noisy or partial inputs. This capability, known as auto-associative memory or "cleanup," is fundamental to both biological neural networks and artificial cognitive architectures. While well-established methods exist for cleaning up discrete symbols (e.g., Hopfield networks [25]), a significant challenge remains: how to perform this restoration when the information being represented is continuous.

Vector Symbolic Algebras (VSAs), also known as Hyperdimensional Computing [27], provide a powerful framework for this challenge. By distributing information across high-dimensional vectors, VSAs allow for the manipulation of symbols through algebraic operations, offering a computational model for how neural systems such as the hippocampus and cortex encode and process information.

Spatial Semantic Pointers (SSPs) extend this framework to continuous spatial domains [30]. They encode continuous variables (e.g., coordinates) into high-dimensional vectors that respect the geometric topology of the represented space. These representations align closely with the principles of the Neural Engineering Framework (NEF) [13] and large-scale biologically plausible models like SPAUN (Semantic Pointer Architecture Unified Network) [15], which demonstrates how spiking neurons can perform complex cognitive tasks.

In biological systems, spatial information is not represented via explicit Cartesian coordinates. Instead, evidence from neuroscience suggests that stable spatial representations are maintained through distributed neural activity, such as the interference patterns observed in grid cells, which are sustained by continuous attractor dynamics [21, 7]. Consequently, computational models of these behaviors should ideally operate without decoding

to an artificial coordinate system, maintaining fidelity to the underlying biological mechanisms.

1.1 The Problem: Cleanup in Continuous Domains

While distributed representations are inherently robust to the failure of individual neurons (local noise), they are susceptible to the accumulation of noise during processing. Despite the theoretical robustness of the VSAs, biological and neuromorphic implementations face physical constraints that introduce degradation from several distinct sources:

- **Neural Variability:** Spiking noise and limited precision in neural activity introduce random perturbations to the vector representation.
- **Symbolic Cross-talk:** Algebraic operations such as bundling and binding introduce interference noise, which scales with the number of items stored in memory.
- **Phase Drift:** In recurrent neural networks used for path integration, small errors accumulate over time, causing phase drift that pushes the representation off the valid SSP manifold.

As this noise accumulates, the vector's magnitude degrades and the precise phase relationships required to encode location are destroyed, leading to a catastrophic loss of semantic and symbolic information. To maintain stable cognition, systems require a mechanism to map these noisy, corrupted representations back to the nearest valid state on the manifold.

Existing neural approaches, such as associative memories [25, 41], operate directly within the embedding space but are typically designed for discrete vocabularies. They do not generalize well to continuous manifolds, where the "vocabulary" of valid states is effectively infinite.

Beyond biological fidelity, there is a strong computational justification for performing cleanup entirely within the high-dimensional embedding space. Leaving the embedding space to perform denoising requires solving an explicit decoding problem, which effectively demands finding the global maximum in a high-dimensional similarity landscape. Because SSPs rely on frequency-based interference patterns, this landscape is inherently complex. The restriction to a finite set of frequency components inevitably results in a non-convex landscape characterized by periodicities and local extrema. Consequently, gradient-based

decoding is prone to getting stuck in local optima, rendering explicit decoding computationally unreliable without exhaustive search.

Decoding, therefore, necessitates either the discretization of the continuous domain (which scales exponentially with dimension and resolution) or the use of iterative optimization algorithms (which trades off memory for significant time complexity) to search for a global maximum. Such optimization is computationally expensive and prone to stagnation [30, 29]. By contrast, performing restoration entirely within the embedding space preserves the homogeneity of the vector-matrix operations inherent to VSAs and allows for efficient, integration-based cleanup.

1.2 Proposed Solution: Geodesic Flow Matching

This thesis formulates cleanup as a **generative transport problem** within the embedding space. The objective is to learn a continuous time-dependent velocity field that actively transports corrupted representations back to the valid SSP manifold.

The investigation highlights a hierarchy of learning dynamics. It is demonstrated that simple feedforward mappings often fail due to the discontinuous nature of random noise pairings. Optimal Transport (OT) couplings are explored as a method to provide necessary global regularization, establishing a structural correspondence between noisy and clean distributions. However, feedforward approaches still lack the local iterative refinement capabilities of flow-based models and struggle to scale to high-resolution representations.

While standard Flow Matching provides this refinement, this work demonstrates that Euclidean flows are insufficient for SSPs. Euclidean interpolants linearly connect points through the ambient space, effectively "cutting through" the unit hypersphere. Consequently, training trajectories destroy the vector magnitude and phase information critical to the representation.

To address this, **Geodesic Flow Matching** is introduced as a model with the correct geometric inductive bias. By constraining transport dynamics to the Riemannian manifold of the hypersphere, this model preserves phase structure and manifold topology, ensuring robust restoration even in high-noise regimes where the Euclidean variants degrade.

The utility of the proposed approach is demonstrated through its integration into a full Spiking Neural SLAM (Simultaneous Localization and Mapping) system. The Geodesic cleanup model is deployed as an online stabilizer for the path integrator, intercepting drifting representations and projecting them back to the manifold. Results indicate that

this on-manifold cleanup stabilizes path integration and significantly improves resource efficiency, allowing a neural population of 1,500 neurons to match the tracking accuracy of a baseline system using 2,500 neurons.

The utility of the proposed approach is demonstrated through its integration into a full Spiking Neural SLAM (Simultaneous Localization and Mapping) system. The Geodesic cleanup model is deployed as an online stabilizer for the path integrator, intercepting drifting representations and projecting them back to the manifold.

Results indicate that this on-manifold cleanup stabilizes path integration and significantly improves resource efficiency. In standard spiking networks, improving representational fidelity is expensive, as suppressing neural noise typically relies on significant increases in the necessary neural resources. By introducing the flow-based stabilizer, a neural population of only 1,500 neurons is able to match the tracking accuracy of a baseline system requiring 2,500 neurons (a 40% reduction in computational cost). This demonstrates that algorithmic cleanup can effectively substitute for raw neuronal volume, offering a scalable path toward dense, robust cognitive maps on hardware-constrained devices.

The remainder of this thesis is structured as follows:

- **Chapter 2** provides background on VSAs, SSPs, and the Conditional Flow Matching framework.
- **Chapter 3** formalizes the methodology. Cleanup is defined as a transport problem, and sampling-based regularization techniques (including Optimal Transport and Schrödinger Bridges) are derived alongside the manifold-aware Geodesic Flow formulation.
- **Chapter 4** presents benchmarks and experiments comparing the proposed flow variants against classical baselines across various dimensions and noise levels.
- **Chapter 5** applies the best-performing model to a downstream SLAM task, evaluating its efficacy in an online dynamic neural simulation.
- **Chapter 6** summarizes the contributions and discusses future directions.

Chapter 2

Background

The focus of this thesis is on associative memories for continuous representational spaces. The specific continuous space considered is derived from a type of Vector Symbolic Algebra (VSA) called a Holographic Reduced Representation (HRR) [**?**] In this chapter, we introduce the theoretical foundations of HRRs, extend them to continuous domains via Spatial Semantic Pointers (SSPs), and identify the noise challenges inherent to these representations. This motivates the need for the robust cleanup methods developed in the remainder of the thesis.

2.1 Vector Symbolic Algebras

Vector Symbolic Algebras (VSAs) are a family of representational frameworks centered around hyper-dimensional encoding of information that allows for symbolic manipulation through algebraic operations. The motivation for high vector dimensionality lies in the vast representational capacity of the embedding space. This allows an exponential number of distinct entities to coexist without significant overlap within the same representation, a necessary condition for the framework's compositionality and noise robustness.

From a neuroscience perspective, VSAs have been proposed as bio-inspired computational mechanisms. They provide models for how neural systems (e.g., hippocampus, cortex) might encode and manipulate symbolic and spatial information. A prominent example is SPAUN (Semantic Pointer Architecture Unified Network) [15], a 2.5-million neuron model of the brain. SPAUN demonstrates how VSAs can unify perception, working memory, and motor control within a biologically plausible spiking neural network. By

using high-dimensional vectors (Semantic Pointers) to pass information between distinct functional modules, the system can perform diverse tasks (e.g. handwriting recognition, logical reasoning) demonstrating the scalability of the VSA framework in complex cognitive modeling.

2.1.1 Holographic Reduced Representations

Holographic Reduced Representations (HRRs) [39] are a specific VSA implementation that employs circular convolution for binding. Unlike localist representations (one-hot vectors), HRRs rely on distributed representations where symbols are high-dimensional base vectors drawn from an independent and identically distributed (i.i.d.) source (typically a distribution on the unit hypersphere). In high dimensions, any two such randomly generated vectors are *quasi-orthogonal*, meaning their dot product approaches zero. This property is critical for minimizing interference when multiple vectors are superimposed.

The core operations for HRRs are defined as follows:

- **Similarity:** Measured via the dot product $\langle \psi, \gamma \rangle$.
- **Bundling (Superposition):** Element-wise addition $\delta = \psi + \gamma$. The result δ remains similar to both ψ and γ. This allows for the representation of sets (e.g., a "fruit bowl" vector containing "apple" + "orange").
- **Binding:** Circular convolution $\delta = \psi \circledast \gamma$. The result is dissimilar to both inputs (quasi-orthogonal) but preserves their information in a compressed format. This is used to create structures or assign roles (e.g., "Color" $\circledast$ "Red").
- **Unbinding:** To retrieve information, an approximate pseudo-inverse is used, denoted as ψ^{-1}. The unbinding operation is $\delta \circledast \psi^{-1} \approx \gamma + \text{noise}$. While often treated as an inverse, this operation is strictly exact only under specific conditions, otherwise resulting in a noise term proportional to the superposition capacity. [1]

HRRs are commonly used to represent complex symbolic data structures. For example, to represent an object that is a "Red Square", one might define random base vectors for

[1] The unbinding operator is implemented as time-reversal (complex conjugate in the Fourier domain) [43]. This corresponds to the exact algebraic inverse ψ^{-1} if and only if ψ is unitary (i.e., all Fourier coefficients have unit magnitude), as is the case for atomic SSPs. For non-unitary vectors, such as bundles, $\psi^{-1} \circledast \psi \neq I$, resulting in the noise term.

properties and values:

$$\textbf{OBJ} = (\text{SHAPE} \circledast \text{SQUARE}) + (\text{COLOR} \circledast \text{RED}).$$

This object can be queried for its color by unbinding with the COLOR vector:

$$\textbf{OBJ} \circledast \text{COLOR}^{-1} \approx \text{RED} + \text{noise}.$$

The noise term arises because COLOR^{-1} also interacts with the shape term (SHAPE $\circledast$ SQUARE), producing a random noise vector that is dissimilar to any valid symbol.

2.2 Spatial Semantic Pointers

Spatial Semantic Pointers (SSPs) extend the HRR VSA framework to continuous spatial domains. While standard HRRs use random vectors to represent discrete symbols, SSPs use continuous functions to map spatial coordinates $x \in \mathbb{R}^{d_{domain}}$ to high-dimensional vectors $\phi(x) \in \mathbb{R}^d$. An SSP can be defined in either the Fourier domain or the real domain. In the Fourier domain, each component is a phasor residing on the complex unit circle, denoted as $\mathbb{S}^1 = \{z \in \mathbb{C} \mid |z| = 1\}$. The vector is defined as:

$$\tilde{\phi}(x) = \left[e^{i\langle\theta_1, x/\ell\rangle},\ e^{i\langle\theta_2, x/\ell\rangle},\ \ldots,\ e^{i\langle\theta_d, x/\ell\rangle}\right] \in \mathbb{C}^d, \tag{2.1}$$

where $\theta_j \in \mathbb{R}^{d_{\text{domain}}}$ are frequency vectors contained in the encoding matrix $\Theta \in \mathbb{R}^{d \times d_{\text{domain}}}$, and ℓ scales the periodicity of the selected frequency vectors.

Because each individual component is constrained to S^1, the aggregate vector $\tilde{\phi}(x)$ lies on a d-dimensional hypertorus (specifically, the product of d circles, $(\mathbb{S}^1)^d$). By enforcing conjugate symmetry on the frequency vectors, an inverse Fourier transform can be applied to yield a real-valued SSP vector:

$$\phi(x) = \mathcal{F}^{-1}\{\tilde{\phi}(x)\} \in \mathbb{R}^d. \tag{2.2}$$

The resulting real vector lies on a projection of this hypertorus embedded in the unit hypersphere of $\mathbb{R}^d$.

Length Scale and Periodicity. The length scale parameter ℓ plays a critical role in the representation. It scales the input domain before projection onto the phase matrix, effectively controlling the frequency of the interference pattern. A smaller length scale

results in higher frequency oscillations, which sharpens the similarity kernel (increasing local resolution) but decreases the global period T, potentially introducing ambiguity if the domain exceeds the period.

If rows of Θ are sampled uniformly from $[-\pi, \pi]$, the dot product between two SSPs induces a sinc kernel, measuring similarity in the original embedded domain (Figure 2.1).

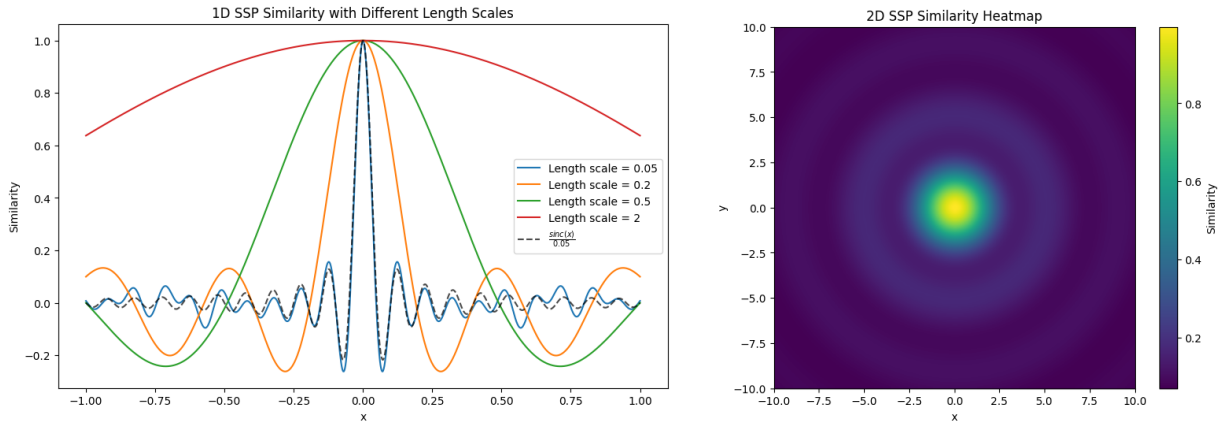


Figure 2.1: Spatial Semantic Pointer (SSP) representations ($d = 487$). (a) The 1D SSP sinc kernel encoding a scalar variable (various length scales displayed). (b) The corresponding 2D SSP representation illustrating the spatial interference pattern.

Alternatively, following Dumont et al. [10], frequency vectors can be further structured to tile the space efficiently. A sub-basis of $d_{domain} + 1$ vectors can be constructed and projected onto the 2D plane (for 2D, we have triplets - a 3D embedded space would have 4, etc.). To achieve high-dimensional encoding, such a triplet basis is scaled by a factor $s \in [s_{min}, s_{max}]$ and rotated by angles $\alpha \in [0, 2\pi/3)$. The final encoding matrix Θ is the concatenation of these scaled and rotated basis sets. This construction yields grid-cell-like interference patterns with a self-similar structure across scales, mimicking the multi-scale grid cells observed in the entorhinal cortex [21].

Locally, the SSP manifold is flat, and binding corresponds to addition in the embedded spatial domain:

$$\phi(x_1) \circledast \phi(x_2) = \mathcal{F}^{-1}\{e^{i\Theta x_1/\ell} \odot e^{i\Theta x_2/\ell}\} = \phi(x_1 + x_2). \tag{2.3}$$

Globally, periodicity arises from the frequency-based representation: there exists a translation T such that

$$e^{i\langle\theta_j,(x+T)/\ell\rangle} = e^{i\langle\theta_j,x/\ell\rangle}, \quad \text{whenever } \langle\theta_j, T/\ell\rangle = 2\pi k_j,\ k_j \in \mathbb{Z},\ \forall j. \tag{2.4}$$

A key advantage of SSPs is their ability to bind with symbolic representations (random high-dimensional vectors assigned semantic labels) to form *cognitive maps*[30]. For example, consider encoding a cat at (x_1, y_1), a mouse at (x_2, y_2), and cheese at (x_3, y_3):

$$\mathbf{M} = \mathbf{CAT} \circledast \phi(x_1, y_1) + \mathbf{MOUSE} \circledast \phi(x_2, y_2) + \mathbf{CHEESE} \circledast \phi(x_3, y_3). \tag{2.5}$$

This map can be queried for objects or their locations via approximate unbinding, e.g.,

$$\mathbf{M} \circledast \phi(x_3, y_3)^{-1} \approx \mathbf{CHEESE} + \epsilon, \tag{2.6}$$

where ϵ denotes interference from the other terms. This motivates the next section, which formally quantifies sources of noise in SSPs.

2.3 Neural Implementation

To implement these mathematical frameworks in a biologically plausible substrate, the methods outlined in the Neural Engineering Framework (NEF) [14] are used. Specifically, the NEF provides a way for translating high-dimensional vectors into spike trains (as well as recovering spike trains into these high dimensional representations).

A vector $x \in \mathbb{R}^d$ is encoded into the activity of a neural population by injecting current J_i into each neuron i:

$$J_i(x) = \alpha_i \langle e_i, x\rangle + J_{bias,i}, \tag{2.7}$$

where e_i is the neuron's preferred direction vector (encoder), α_i is a gain scaling factor, and $J_{bias,i}$ is a background current. This current drives the spiking activity a_i via a non-linear neuron model $G[\cdot]$ (e.g., Leaky Integrate-and-Fire):

$$a_i(x) = G[J_i(x)]. \tag{2.8}$$

To decode the represented vector $\hat{x}$ from the population activity, one can solve for linear decoders D that minimize the reconstruction error:

$$\hat{x} = \sum_{i=1}^{N} d_i a_i(x) = D^\top a(x). \tag{2.9}$$

These decoders are typically solved via least-squares optimization. This allows operations defined on vectors (such as binding and superposition) to be performed directly by the neural population through synaptic weight matrices derived from these decoders.

2.4 The Problem of Corrupted Representations

Although SSPs provide a powerful framework for encoding and manipulating symbolic and spatial information, implementations of cognitive models based on HRR principles introduce noise that gradually pushes representations away from the structured manifold. Off-manifold corruption is critical because VSA operations rely on the geometric properties of the embedding. If a vector drifts from the manifold (e.g., loss of unit magnitude or phase coherence), the approximate inverse operation fails, and the dot-product similarity with the correct target drops, introducing errors in the retrieval of stored information. Several distinct noise sources are identified below.

Bundling and Unbinding (cross-talk noise). Generalizing the Cat-Mouse-Cheese example, consider a memory trace with n bound objects:

$$\mathbf{M} = \sum_{i=1}^{n} \mathbf{O}_i \circledast \phi(x_i), \tag{2.10}$$

where each $\mathbf{O}_i \in \mathbb{R}^d$ is a symbolic vector and $\phi(x_i)$ is a location SSP. Querying with the inverse location $\phi(x_j)^{-1}$ yields

$$\mathbf{M} \circledast \phi(x_j)^{-1} = \underbrace{\mathbf{O}_j}_{\text{signal}} + \underbrace{\sum_{i \neq j} \mathbf{O}_i \circledast \phi(x_i - x_j)}_{\epsilon}. \tag{2.11}$$

The first term retrieves the desired object. Note that due to the approximate nature of the inverse, retrieval is not exact.

The second term constitutes cross-talk interference. Relying on the quasi-orthogonality assumption, each interference term behaves like a random direction of the same length as $\mathbf{O}_i$. Summing over $(n-1)$ such terms yields a noise distribution:

The second term constitutes cross-talk interference. Under the assumption that the atomic vectors are i.i.d., the interference term behaves as a sum of $(n-1)$ random vectors. By the Central Limit Theorem, as the number of stored items increases, the distribution of this sum converges to a multivariate Gaussian [39]:

$$\epsilon \sim \mathcal{N}\big(0, \tfrac{n-1}{d} I_d\big). \tag{2.12}$$

Thus, the *per-component variance* of the noise grows linearly with the number of stored items $(n-1)$ and shrinks inversely with dimensionality d.[2]

Accumulating Phase Drift in Recurrence. A second source of noise arises from the recurrent connections used to implement integration. When an SSP is updated over time by recurrent dynamics, small inaccuracies in spike timing or synaptic transmission accumulate as phase errors in the Fourier components. This phenomenon has been noted in neural models of path integration [55, 10], where oscillatory or attractor-based recurrent networks gradually drift away from the correct representation unless corrective mechanisms are introduced.

For a single Fourier component,

$$\tilde{\phi}_j(x_t) = e^{i\langle \theta_j, x_t \rangle},$$

the ideal update is

$$\tilde{\phi}_j(x_{t+1}) = \tilde{\phi}_j(x_t)\, e^{i\Delta\theta_{j,t}}.$$

With noise,

$$\tilde{\phi}_j(x_{t+1}) = \tilde{\phi}_j(x_t)\, e^{i(\Delta\theta_{j,t}+\delta_{j,t})}, \qquad \delta_{j,t} \sim \mathcal{N}(0, \sigma^2).$$

After t steps, the cumulative perturbation is

$$\epsilon_{j,t} = \exp\left(i \sum_{\tau=1}^{t} \delta_{j,\tau} \right).$$

Because the accumulated phase error follows a random walk $\sum_{\tau=1}^{t} \delta_{j,\tau} \sim \mathcal{N}(0,\, t\sigma^2)$, the resulting multiplicative noise term follows a wrapped distribution:

$$\epsilon_{j,t} \;\sim\; \text{WrappedNormal}(0, t\sigma^2) \quad \text{on } \mathbb{S}^1.$$

Physically, this wrapping accounts for the periodicity of the phase. For small $t\sigma^2$, the distribution is sharply concentrated near the ground truth; for large $t\sigma^2$, the tails of the Gaussian wrap around the circle and overlap, tending towards a uniform distribution where all phase information is lost.

[2] As mentioned in 2.2, similarity measurements of SSPs approximate a sinc kernel. This means that location vectors within the high-lobe regions of the kernel have nonzero similarity, leading to structured bias in the interference in addition to the random variance described above. When many items are bundled, the aggregate interference is spread across dimensions, approaching the above noise model

In higher dimensions, the Fourier-domain SSP is a vector of such noisy phasors:

$$\tilde{\phi}(x_t) = \left(\epsilon_{1,t} e^{i\langle \theta_1, x_t \rangle},\ \ldots,\ \epsilon_{d,t} e^{i\langle \theta_d, x_t \rangle}\right).$$

Since each component drifts independently, the overall noise distribution lies on the hypertorus $(S^1)^d$ as a product of independent wrapped normal distributions.

Neural Activity Noise (Spiking Variability). While the phase drift described above represents the temporal accumulation of error, the primary physical driver of these perturbations is the instantaneous variability of the neural substrate. Biological neurons exhibit limited precision due to axonal jitter, stochastic vesicle release, and neurotransmitter variability. These diverse sources can be modeled as a single additive noise term ξ_i on the ideal firing rate:

$$a_i = \bar{a}_i(x) + \xi_i, \quad \xi_i \sim \mathcal{N}(0, \sigma_a^2). \tag{2.13}$$

When decoding the estimate $\hat{x}$ (as defined in Section 2.3), this noise propagates into the vector representation:

$$\hat{x} = D^\top(\bar{a}(x) + \xi) = x + D^\top \xi. \tag{2.14}$$

The term $\eta = D^\top \xi$ represents the decoding error. Assuming the noise is independent and has equal variance across neurons ($\Sigma_a = \sigma_a^2 I_{N_{\text{tot}}}$), the variance per dimension scales as:

$$\text{Var}(\eta_k) \approx \frac{\sigma^2 d^2}{N_{\text{tot}}}. \tag{2.15}$$

Thus, *per-component noise increases linearly with dimensionality* under a fixed neuron budget N_{tot}. Since the total error is the sum across d dimensions, the total mean-squared error of the vector representation grows as d^2/N_{tot}.

Together, these effects motivate the development of dedicated cleanup models that can denoise corrupted SSPs, restoring them to the valid Clifford torus manifold while preserving both semantic and spatial structure.

2.5 Current Solutions

Approaches for cleaning up corrupted SSPs vary in implementation and biological realism. They can generally be categorized into **prototype-based methods**, which store a discrete

set of valid vectors and snap noisy inputs to the nearest one, and **manifold-based methods**, which learn the continuous geometry of the valid space. The following subsections review grid-based methods, associative memories, attractor networks, and optimization-based decoding.

2.5.1 Normalization

Across the noise sources considered (neural variability, cross-talk, and off-manifold sampling), a common stabilization strategy is to renormalize vectors back to unit length:

$$\hat{\phi} \;=\; \frac{\phi}{\|\phi\|}. \tag{2.16}$$

This operation projects corrupted vectors back onto the unit hypersphere in $\mathbb{R}^d$, ensuring that similarity measures (e.g., cosine similarity) remain well defined. However, normalization alone does not remove structured noise (such as cross-talk or phase drift), but does constrain their effect to directions on the hypersphere.

2.5.2 Grid Lookup

A simple baseline for cleanup is to construct a grid of candidate points in the original domain, encode each into an SSP, and choose the closest match to the noisy input. Formally, given a noisy vector ϕ' and a set of candidates $\{\phi(x) : x \in \mathcal{G}\}$, the cleanup is performed as follows:

$$\hat{\phi} = \arg\max_{x \in \mathcal{G}} \langle \phi(x), \phi' \rangle, \tag{2.17}$$

where $\mathcal{G}$ is the set of grid points. This method guarantees that the cleaned SSP corresponds to a valid encoding. However, the number of stored SSPs grows exponentially with the dimension of the domain, making it memory intensive and prohibitively slow for high-resolution grids. Grid lookup is therefore best understood as a computational baseline rather than a plausible neural mechanism.

2.5.3 Associative Memories

Cleanup can also be implemented as an associative memory that maps a noisy input ϕ' to a clean target ϕ. While simple linear auto-association (finding a weight matrix W such that

$\phi \approx W\phi'$) fails for large vocabularies, Stewart et al. (2011) [46] demonstrated a biologically plausible cleanup memory using the Neural Engineering Framework (NEF).

Rather than a direct linear transformation, this approach leverages a hidden layer of neurons where each neuron has a "preferred direction" vector encoding e_i corresponding to a valid symbol in the vocabulary. The activity of the neurons is governed by the dot product of the input and the encoder, passed through a non-linear activation function (such as LIF).

$$a_i(\phi') = G[\alpha_i \langle e_i, \phi' \rangle + J_{bias}] \tag{2.18}$$

The non-linearity is crucial as it acts as a threshold, filtering out low-similarity noise. The clean vector is then reconstructed by a weighted sum of these activities. While this method is fast (5-10ms) and biologically realistic, the number of neurons required scales linearly with the vocabulary size, effectively storing a discrete prototype for every valid symbol.

2.5.4 Attractor Networks

Attractor networks function by defining an energy landscape where valid representations correspond to local minima. The system evolves over time, "sliding" down the energy surface to the nearest stored memory.

Classical Hopfield Networks

Hopfield (1982) introduced an associative network based on binary neurons $V_i \in \{0, 1\}$ [25]. The network minimizes a quadratic energy function:

$$E = -\frac{1}{2} \sum_{i \neq j} T_{ij} V_i V_j \tag{2.19}$$

where T_{ij} determines the synaptic weights. The system evolves asynchronously, converging to a stable state that represents a stored memory. This classical model suffered from severe capacity limitations. For a network of size d, the number of patterns that can be stored scaled linearly with neuron count before retrieval errors became catastrophic.

Modern Hopfield Networks

Recent extensions, termed "Modern Hopfield Networks" or "Dense Associative Memories," dramatically increase storage capacity by replacing the quadratic energy function with steeper non-linearities. Ramsauer et al. (2020) proposed an energy function based on the log-sum-exp (LSE) function [41]:

$$E(\xi) = -\frac{1}{\beta} \ln \left(\sum_{i=1}^{N} \exp(\beta x_i^T \xi) \right) + \frac{1}{2} \xi^T \xi + C, \tag{2.20}$$

where $X = [x_1, \ldots, x_N]$ contains the stored patterns and β controls the temperature of the system. This formulation allows for continuous states and yields an update rule equivalent to the attention mechanism in Transformers [49]:

$$\xi^{new} = X \cdot \text{softmax}(\beta X^T \xi) \tag{2.21}$$

This model possesses exponential storage capacity. However, like the associative memory approach, it relies on storing discrete prototypes (the matrix X). For continuous domains like SSPs, this still requires discretizing the manifold into a finite set of stored attractors.

2.5.5 Optimization-Based Decoding

A computational alternative to neural attractors is to treat cleanup as a direct optimization problem. Given a noisy input ϕ', a search can be applied for the spatial location $x \in \mathcal{D}$ that maximizes similarity:

$$\hat{x} = \arg\max_{x \in \mathcal{D}} \langle \phi(x), \phi' \rangle. \tag{2.22}$$

Since the SSP similarity landscape is non-convex and periodic, gradient-based methods (like L-BFGS-B) are typically initialized with a coarse grid search. While accurate, this approach requires iterative optimization at inference time and explicit decoding to the spatial domain, making it biologically implausible.

2.5.6 Phase-Coupled Optimization

Bremer and Orchard [6] extend the optimization-based approach by exploiting the phasor structure of SSPs. Their method introduces coupling constraints across Fourier components, improving stability of decoding under noise. In this formulation, the optimization

objective incorporates not only the dot product similarity but also consistency in the relative phases of the encoded components:

$$L(x) = -\langle \phi(x), \phi' \rangle + \lambda \, \Psi(x, \phi'), \tag{2.23}$$

where Ψ penalizes deviations in phase relationships across components and λ controls the strength of this constraint. By preserving coherence in the spectral domain, this method yields more stable decoding in high-noise settings. Nevertheless, it retains the same drawbacks as direct optimization: iterative inference and the requirement to decode to the original spatial domain.

2.5.7 Generative Cleanup Models

The methods described above share a common limitation: they either store a finite set of discrete prototypes (Grid, Associative, Hopfield) or require iterative search over the spatial domain (Optimization).

This thesis proposes a fundamentally different approach: treating cleanup as a generative transport problem. Rather than snapping to the nearest stored prototype, a model is trained to learn the structure of the SSP manifold itself. The model learns a continuous vector field that transports probability mass from the distribution of corrupted vectors back to the manifold of valid SSPs.

Diffusion models

Diffusion models [23, 45] learn the *score* $\nabla_\phi \log p_t(\phi)$, representing the gradient of the log-density of the data distribution. Cleanup is performed by simulating the reverse-time stochastic differential equation (SDE):

$$d\phi_t = [f(\phi_t, t) - g^2(t) s_\theta(\phi_t, t)]dt + g(t) d\bar{w} \tag{2.24}$$

where ϕ_t represents the noisy state vector at time t, and $f(\cdot)$ and $g(\cdot)$ are the drift and diffusion coefficients of the forward process, respectively. The term $s_\theta(\phi_t, t)$ denotes the neural network parameterized by θ, which approximates the score function, and $d\bar{w}$ is the differential of a standard Wiener process (Brownian motion) evolving in reverse time.

While powerful, diffusion models rely on iterative stochastic sampling (a "random walk" guided by gradients), which can be slow and computationally expensive. Furthermore, they typically assume an unstructured Gaussian prior, which does not align well with the specific geometry of the SSP manifold.

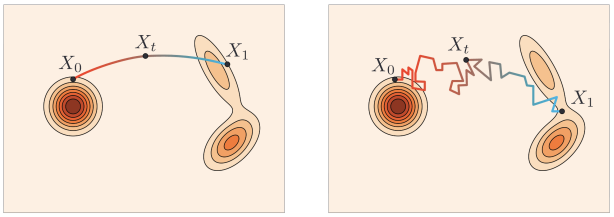


Figure 2.2: Conditional paths generated by flow matching models vs diffusion

Flow Matching models

Flow matching models [33] offer a deterministic alternative. Instead of learning a score function, they learn a time-dependent *velocity field* $v_\theta(\phi, t)$ that defines a direct trajectory from noise to data:

$$\dot{\phi}_t = v_\theta(\phi_t, t) \tag{2.25}$$

During inference, cleanup is achieved by integrating this Ordinary Differential Equation (ODE) over a fixed time interval. Unlike diffusion, flow matching allows for flexible "couplings" between the noise and data distributions, enabling straighter, more efficient inference paths (see Figure 2.2). This deterministic transport is the core focus of the proposed Geodesic Cleanup method.

Chapter 3

Method

The goal of this chapter is to formalize the proposed cleanup framework as a generative transport process between corrupted and clean SSP representations. Building on the geometric and noise analyses introduced in Chapter 2, the denoising process is now modeled as a continuous-time transformation in the SSP embedding space.

Cleanup is thus framed as learning a velocity field that continuously transports samples from a corrupted source distribution $p_0(\phi)$ toward a clean target $p_1(\phi)$ along a smooth probability path. This formulation connects naturally to recent advances in flow-based generative modeling, where objectives such as *conditional flow matching* (CFM) and its optimal-transport extensions provide simulation-free estimators of these dynamics.

The remainder of this chapter is organized as follows. Section 3.1 introduces the setup and notation. Section 3.2 formalizes the continuity constraint governing density evolution. Section 3.3 presents joint couplings between corrupted and clean SSPs. Section 3.4 introduces the direct Feedforward baseline, motivating the need for iterative refinement. Section 3.5 defines the hierarchy of flow-matching families, culminating in the proposed Geodesic Flow. Section 3.6 outlines implementation details. Section 3.7 explores the classical baselines. Finally, Section 3.8 measures computational complexity for the proposed methods as well as the baselines.

3.1 Setup and Notation

Source distribution. The corrupted inputs are represented by the distribution $p_0(\phi)$, modeled as normalized isotropic Gaussian noise projected onto the unit hypersphere:

$$\phi_0 = \frac{z}{\|z\|}, \qquad z \sim \mathcal{N}(0, I_d).$$

This provides a valid approximation to maximally corrupted SSPs, which are normalized but unstructured vectors on $\mathbb{S}^{d-1}$.

Target distribution. The clean distribution $p_1(\phi)$ consists of valid SSP encodings $\phi(x)$ of spatial samples $x \in [-L, L]^{d_{\text{domain}}}$ according to Eq. 2.2. To ensure uniform coverage of the spatial domain without grid artifacts, samples are drawn using Sobol quasi-random sequences.

While SSPs can be constructed using random frequency vectors (RandomSSP), this stochastic sampling is inefficient for low-dimensional spaces, often resulting in "spectral gaps" that yield anisotropic coverage and uneven spatial resolution. Instead, this thesis utilizes **Hexagonal SSPs** (HexSSP).

Following Dumont and Eliasmith [10], frequency components are arranged on a hexagonal lattice with fixed rotations and scales. This structure exploits the optimal packing density of the frequency domain, maximizing representational capacity and ensuring that the spatial resolution is uniform in all directions. Furthermore, this specific interference pattern approximates the grid-cell activity observed in biological navigation systems. Consequently, HexSSPs are used as the encoding basis for all subsequent experiments to ensure consistent, efficient spatial coverage.

The objective is to learn a continuous map $\phi_0 \sim p_0 \to \phi_1 \sim p_1$. We explore two paradigms for this mapping: a direct feedforward regression, and a time-dependent transport parameterized by a velocity field $v_\theta(\phi, t)$.

3.2 Transport Dynamics and Continuity Constraint

For flow-based models, the evolution of densities under a smooth velocity field $v_t(\phi)$ is governed by the *continuity equation*:

$$\partial_t p_t(\phi) + \nabla \cdot (v_t p_t) = 0. \tag{3.1}$$

This equation expresses conservation of probability mass as samples move along the flow. Each flow model defines an interpolant between source and target samples,

$$\phi_t = I(t, \phi_0, \phi_1) + \gamma(t)z, \qquad z \sim \mathcal{N}(0, I_d), \tag{3.2}$$

where I defines the mean trajectory and $\gamma(t)$ controls the noise. The instantaneous velocity field is defined as the conditional expectation of its time derivative:

$$v_t(\phi) = \mathbb{E}\big[\dot{\phi}_t \mid \phi_t = \phi\big] = \mathbb{E}\big[\partial_t I(t, \phi_0, \phi_1) + \dot{\gamma}(t)z \,\big|\, \phi_t = \phi\big]. \tag{3.3}$$

The neural velocity field $v_\theta(\phi, t)$ is trained to regress the conditional vector field generating the probability paths. The standard Conditional Flow Matching (CFM) objective minimizes the expected mean squared error (MSE) between the network prediction and the instantaneous velocity of the interpolant $\dot{\phi}_t$:

$$\mathcal{L}_{\text{CFM}} = \mathbb{E}_{t,\phi_0,\phi_1}\left[\|v_\theta(\phi_t, t) - \dot{\phi}_t\|^2\right]. \tag{3.4}$$

However, given that the data is defined on the unit hypersphere, the precise magnitude of the vector field is less critical than its orientation. To focus learning on directional alignment rather than magnitude (which is redundant on the unit sphere), we replace the standard MSE with a cosine-similarity loss:

$$\mathcal{L}_{\cos} = \mathbb{E}_{t,\phi_0,\phi_1,z}\left[1 - \frac{v_\theta(\phi_t, t)^\top \dot{\phi}_t}{\|v_\theta(\phi_t, t)\|\|\dot{\phi}_t\|}\right]. \tag{3.5}$$

3.3 Coupling distributions for CFM

Both feedforward and flow-based frameworks require sampling pairs (ϕ_0, ϕ_1) from a joint coupling $\pi(\phi_0, \phi_1)$. The choice of coupling determines the structural correspondence between noise and data.

3.3.1 Independent Coupling

The simplest choice is an independent coupling $\pi_{\text{Ind}}(\phi_0, \phi_1) = p_0(\phi_0)p_1(\phi_1)$. Samples are drawn separately, creating no explicit alignment. In this regime, samples from the source distribution are paired with target data points entirely at random, ignoring geometric proximity (Figure 3.1, Left) [35].

3.3.2 Exact Optimal-Transport Coupling

A more principled alternative is the *optimal transport (OT) coupling*, which minimizes the expected cost of transporting mass between the source and target distributions:

$$\pi^* = \arg\min_{\pi \in \Pi(p_0, p_1)} \sum_{i,j} C_{ij} \pi_{ij} \quad \text{s.t.} \quad \pi \mathbf{1} = p_0,\ \pi^\top \mathbf{1} = p_1, \tag{3.6}$$

where π^* is the optimal transport plan, $\Pi(p_0, p_1)$ is the set of valid joint distributions with marginals p_0 and p_1, and C_{ij} is a ground cost measuring the discrepancy between ϕ_0^i and ϕ_1^j.

As shown in Figure 3.1 (Center), this coupling yields globally straight, non-intersecting trajectories that respect the geometry of the manifold. In Euclidean space, the standard quadratic cost $C_{ij} = \|\phi_i - \phi_j\|^2$ yields globally straight, minimum-energy trajectories between samples (ϕ_i, ϕ_j). However, since both corrupted and clean SSPs are normalized and lie on the unit hypersphere $\mathbb{S}^{d-1}$, the appropriate distance metric is the *angular cost*:

$$C_{ij}^{\text{ang}} = \arccos(\phi_i^\top \phi_j)^2, \tag{3.7}$$

which measures squared Geodesic distance along the hyperspherical manifold. In practice, π^* is estimated within each batch, as solving the global assignment problem scales cubically with sample count [48].

3.3.3 Entropy-Regularized Coupling

The OT coupling assumes a strict, deterministic correspondence between corrupted and clean samples. However, due to the stochastic nature of the corruption, a single noise sample may plausibly correspond to multiple clean embeddings.

Introducing *entropic regularization* yields a probabilistic coupling that balances transport efficiency with stochasticity (Figure 3.1, Right). The resulting objective is equivalent to the *Schrödinger bridge problem* [8, 48]:

$$\pi_\varepsilon^* = \arg\min_{\pi \in \Pi(p_0, p_1)} \Big[\sum_{i,j} C_{ij} \pi_{ij} + \varepsilon\, H(\pi) \Big], \qquad H(\pi) = \sum_{i,j} \pi_{ij} \left(\log \pi_{ij} - 1 \right), \tag{3.8}$$

where the regularization strength is set to $\varepsilon = 2\sigma^2$. Here, σ represents the diffusion scale of the reference Brownian process. This parameter σ acts as a bridge between the coupling and the training dynamics defined in Section 3.5.

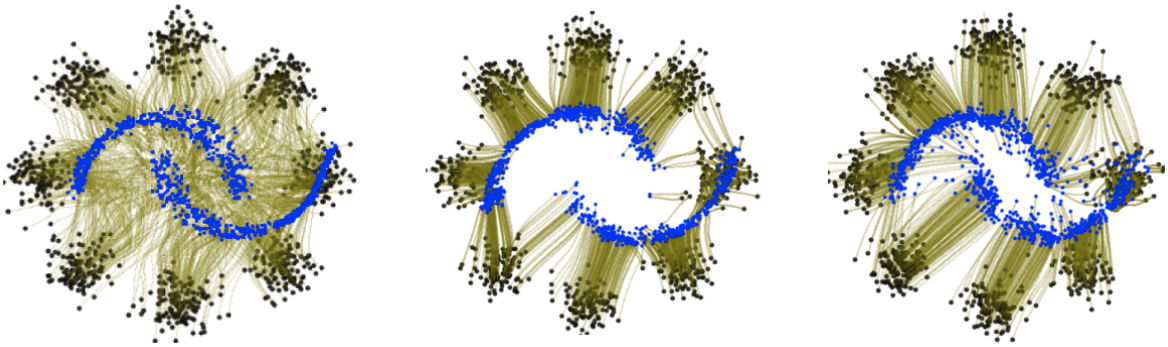


Figure 3.1: Visualization of coupling strategies on a toy 2D dataset [48]. **(Left)** Independent coupling pairs source noise (black) with target data (blue) randomly, creating a disordered vector field with many intersecting paths. **(Center)** Exact Optimal Transport (OT) coupling assigns pairs based on geometric proximity, resulting in straight, non-intersecting trajectories that minimize total displacement. **(Right)** Entropy-regularized coupling (Schrödinger Bridge) relaxes the strict determinism of OT, producing a probabilistic mapping where mass from one source point is diffused across multiple nearby target points.

The second term $H(\pi) = \sum \pi_{ij}(\log \pi_{ij} - 1)$ penalizes sparsity, encouraging the mass to spread out. The same angular cost $C_{ij} = \arccos(\phi_i^\top \phi_j)^2$ is applied here.

This coupling serves as a continuum connecting the two previous methods: as $\varepsilon \to 0$, the solution converges to the deterministic OT coupling; as $\varepsilon \to \infty$, it converges to the independent coupling, where every source point is equally likely to map to every target point.

Rather than enforcing a single deterministic mapping, this model effectively learns the most likely *stochastic process* that transforms the noisy distribution $p_0(\phi)$ into the clean target $p_1(\phi)$ while minimizing the divergence from a reference Brownian motion with diffusion scale σ. This reflects the fact that a noise sample from the unit sphere (p_0) may correspond to multiple plausible clean embeddings due to the stochastic nature of the corruption.

3.4 Direct Mapping: The Feedforward Baseline

Before introducing iterative flows, the direct feedforward formulation is considered. Here, a network f_θ is trained to map the prior distribution directly to clean SSPs in a single step:

$$\hat{\phi} = \frac{f_\theta(\tilde{\phi})}{||f_\theta(\tilde{\phi})||}.$$

The network shares the same architecture as the flow models but directly regresses the cleaned embedding by minimizing cosine similarity loss:

$$\mathcal{L}_{FF} = 1 - \frac{f_\theta(\tilde{\phi})^\top \phi_{\text{clean}}}{\|f_\theta(\tilde{\phi})\| \, \|\phi_{\text{clean}}\|}. \tag{3.9}$$

This approach highlights the importance of the coupling distributions defined in Section 3.3. With independent couplings, the mapping is discontinuous and difficult to learn. However, applying OT or SB couplings imposes geometric structure on the dataset, potentially enabling the feedforward model to discern the underlying manifold structure. While efficient (one-step inference), this approach lacks the iterative refinement capabilities of flow-based models. As such, it provides a useful baseline in the experiments described below.

3.5 Iterative Transport: Flow-Matching Families

Flow Matching enables iterative refinement and can handle complex noise distributions. The flow family is determined by:

1. The coupling $\pi(\phi_0, \phi_1)$ used to sample endpoint pairs.
2. The interpolant $I(t, \phi_0, \phi_1)$ defining the mean trajectory.
3. The noise schedule $\gamma(t)$.

Together, these determine the reference stochastic process $\phi_t = I(t, \phi_0, \phi_1) + \gamma(t)z$ and the associated target velocity $\dot{\phi}_t$ for training the neural field $v_\theta(\phi, t)$.

3.5.1 Euclidean Flow Matching

Independent CFM (I-CFM). The simplest formulation defines straight-line interpolants with no coupling structure or noise schedule:

$$\phi_t = (1-t)\phi_0 + t\phi_1, \qquad \dot{\phi}_t = \phi_1 - \phi_0. \tag{3.10}$$

Each pair (ϕ_0, ϕ_1) is drawn independently from p_0 and p_1, producing globally linear but unaligned trajectories. In experiments, this method is referred to as euc_det.

Once trained, samples are generated by integrating $\dot{\phi}_t = v_\theta(\phi_t, t)$ from $t = 0$ to $t = 1$, using numerical solvers such as Euler or Heun integration. Although I-CFM provides a strong baseline, its trajectories are constructed without regard for geometric proximity, failing to pair source and target samples based on minimum distance.

Optimal Transport Flow Matching (OT-CFM) OT-CFM improves alignment by sampling $(\phi_0, \phi_1) \sim \pi^*$ from the optimal coupling under the angular cost defined in §3.3.2 and adding constant isotropic noise to the mean interpolant during training:

$$\phi_t = (1-t)\phi_0 + t\phi_1 + \gamma(t)z, \qquad \gamma(t) = \sigma, \qquad z \sim \mathcal{N}(0, I_d). \tag{3.11}$$

The velocity $\dot{\phi}_t = \phi_1 - \phi_0$ with OT coupling defines a constant-speed, minimum energy flow with uniform stochastic thickening around each mean trajectory. The added noise ensures smoother regression targets and greater robustness to out-of-distribution perturbations. This method is referred to as euc_ot.

Schrödinger Bridge Flow Matching (SB-CFM) The Schrödinger-bridge formulation extends OT-CFM by embedding the entropy-regularized coupling π^*_ε defined in §3.3.3. The corresponding interpolant follows a Brownian bridge with *time-dependent* variance:

$$\phi_t = (1-t)\phi_0 + t\phi_1 + \gamma(t)z, \qquad \gamma(t) = \sigma\sqrt{t(1-t)}, \quad z \sim \mathcal{N}(0, I_d). \tag{3.12}$$

Differentiating gives the target velocity:

$$\dot{\phi}_t = \phi_1 - \phi_0 + \dot{\gamma}(t)z, \qquad \dot{\gamma}(t) = \tfrac{\sigma(1-2t)}{2\sqrt{t(1-t)}}. \tag{3.13}$$

The resulting trajectories represent the expected velocity field of the entropically regularized coupling, yielding smoother, uncertainty-aware flows that interpolate between

deterministic OT and purely randomly paired interpolations. This method is referred to as euc_sb.

All Euclidean models are summarized in Algorithm 1 below:

Algorithm 1 Training: Euclidean Flow Matching

Sample $t \sim \mathcal{U}(\varepsilon, 1-\varepsilon)$ and endpoints $\phi_0, \phi_1 \sim \pi(\phi_0, \phi_1)$
Draw $z \sim \mathcal{N}(0, I_d)$; define $\gamma_{\text{SB}}(t) = \sigma\sqrt{t(1-t)}$
$\mu_t \leftarrow (1-t)\phi_0 + t\phi_1$ ▷ linear mean
if $s =$ `euc_det` **then**
    $\phi_t \leftarrow \mu_t$
    $u_{\text{true}} \leftarrow \phi_1 - \phi_0$ ▷ constant drift
else if $s =$ `euc_ot` **then** ▷ constant ambient noise
    $\phi_t \leftarrow \mu_t + \sigma z$
    $u_{\text{true}} \leftarrow \phi_1 - \phi_0$
else if $s =$ `euc_sb` **then**
    $\phi_t \leftarrow \mu_t + \gamma_{\text{SB}}(t) z$
    $\text{denom} \leftarrow \max\{2t(1-t), \epsilon\}; \quad \text{corr} \leftarrow \dfrac{\phi_t - \mu_t}{\text{denom}}$
    $u_{\text{true}} \leftarrow (\phi_1 - \phi_0) + (1-2t)\,\text{corr}$ ▷ noise-aware SB drift
end if
return $(\phi_t, t, u_{\text{true}})$

3.5.2 Geodesic CFM

The previous flow-matching formulations assume Euclidean linear interpolation (or linear mean trajectories) between the source and target vectors. However, as previously mentioned, both clean and corrupted SSPs are normalized representations that lie on the unit hypersphere $\mathbb{S}^{d-1} \subset \mathbb{R}^d$. The Euclidean interpolants encountered during training therefore do not accurately reflect the data manifold encountered in downstream inference. For deterministic flows, this leads to off-manifold trajectories that cut across the sphere, while for stochastic versions, the thickened support may happen to cover parts of the manifold but without any geometric guarantees.

To ensure that all intermediate states remain on the manifold, flows can be directly defined on $\mathbb{S}^{d-1}$ using the Riemannian logarithmic and exponential maps. This allows motion along the intrinsic Geodesic of the manifold rather than through ambient Euclidean space.

Geodesic Interpolant. By construction, all Spatial Semantic Pointers are normalized unit vectors. Similarly, sampled noise is on the hypersphere. Given a start point $\phi_0 \in \mathbb{S}^d$ and target $\phi_1 \in \mathbb{S}^d$, the logarithmic map at ϕ_0 projects ϕ_1 onto the tangent space $T_{\phi_0}\mathbb{S}^d$:

$$\mathrm{Log}_{\phi_0}(\phi_1) = \frac{\theta}{\sin\theta}\Big(\phi_1 - (\phi_0^\top \phi_1)\,\phi_0\Big), \qquad \theta = \arccos(\phi_0^\top \phi_1). \tag{3.14}$$

The exponential map projects a tangent vector v back onto the hypersphere:

$$\mathrm{Exp}_{\phi_0}(v) = \cos(\|v\|)\,\phi_0 + \sin(\|v\|)\frac{v}{\|v\|}. \tag{3.15}$$

These operators ensure that the resulting trajectory $\mathrm{Exp}_{\phi_0}(v)$ remains on $\mathbb{S}^{d-1}$ for all tangent vectors v.

Using these maps, the Geodesic interpolant between ϕ_0 and ϕ_1 is defined as

$$\phi_t = \mathrm{Exp}_{\phi_0}\big(t\,\mathrm{Log}_{\phi_0}(\phi_1)\big), \qquad t \in [0,1]. \tag{3.16}$$

This trajectory follows the great-circle arc on the sphere connecting ϕ_0 and ϕ_1, ensuring that intermediate points ϕ_t maintain unit norm.

Differentiating the Geodesic interpolant gives the instantaneous tangent-space velocity:

$$\dot{\phi}_t = \mathrm{Log}_{\phi_t}(\phi_1), \tag{3.17}$$

which specifies the direction of motion along the manifold toward ϕ_1. As before, the true velocity field is the conditional expectation over endpoint pairs:

$$v_t^*(\phi) = \mathbb{E}_{(\phi_0,\phi_1)\sim\pi}[\,\dot{\phi}_t \mid \phi_t = \phi\,]. \tag{3.18}$$

The model is trained to regress the velocity field via the same flow-matching loss defined in Eq.3.5.

Thickening the Support

The deterministic Geodesic flow can be extended to stochastic variants by adding isotropic noise with respect to the tangent space of the manifold. In the simplest case, one may take any of the Euclidean interpolants described earlier (e.g., OT-CFM or SB-CFM) and replace the linear interpolation $\phi_t = (1-t)\phi_0 + t\phi_1$ with its Riemannian counterpart $\phi_t = \mathrm{Exp}_{\phi_0}\big(t\,\mathrm{Log}_{\phi_0}(\phi_1)\big)$, while maintaining the same additive noise schedule $\gamma(t)$ controlling variance. This defines a stochastic process whose mean trajectory follows the Geodesic, but whose support is thickened isotropically around it. In experiments, these methods are labelled geo_amb_const and geo_amb_sb.

Geodesic Stochastic Interpolant

In the Riemannian Neural Geodesic Interpolant (RNGI) formulation [51], stochasticity is introduced intrinsically on the manifold. Let $v = \mathrm{Log}_{\phi_0}(\phi_1)$ and $\mu_t = \mathrm{Exp}_{\phi_0}(t\,v)$ denote the Geodesic mean. Instead of adding Euclidean noise, a *tangent-space* perturbation is applied at the current Geodesic point and mapped back with the exponential map:

$$\phi_t \;=\; \mathrm{Exp}_{\mu_t}\big(\delta_t\big), \qquad \delta_t \;=\; \gamma(t)\,\xi_t, \;\; \xi_t \sim \mathcal{N}\big(0, I_{T_{\mu_t}\mathbb{S}^{d-1}}\big). \tag{3.19}$$

Here ξ_t is sampled intrinsically in the tangent space $T_{\mu_t}\mathbb{S}^{d-1}$ (e.g., by projecting a Gaussian from $\mathbb{R}^d$), so the perturbation remains tangent at each μ_t.

The corresponding instantaneous *training target* decomposes into the deterministic Geodesic velocity and, when the noise scale is time-varying, a standard SB correction in the tangent space:

$$u_{\mathrm{geo}}(t) \;=\; \frac{d}{dt}\,\mathrm{Exp}_{\phi_0}(t\,v) \;\in\; T_{\mu_t}\mathbb{S}^{d-1}, \tag{3.20}$$

$$u_{\mathrm{true}}(t) \;=\; \begin{cases} u_{\mathrm{geo}}(t) \;+\; (1-2t)\,\dfrac{\delta_t}{2\,t(1-t)}, & \text{if } \gamma(t) = \sigma\sqrt{t(1-t)}), \\ u_{\mathrm{geo}}(t), & \text{if } \gamma(t) = \sigma \end{cases} \tag{3.21}$$

This tangential diffusion preserves manifold structure while increasing sampling diversity and robustness to noise. The tangent stochastic variants are referred to as geo_tan_const and geo_tan_sb in experiments.

All Geodesic methods are summarized in Algorithm 2 below, followed by the inference procedure for all flow matching models in Algorithm 3.

Algorithm 2 Training: Geodesic Flow Matching on $\mathbb{S}^{d-1}$

Sample t, $\phi_0, \phi_1 \sim \pi(\phi_0, \phi_1)$; draw $\xi \sim \mathcal{N}(0, I_d)$; $\gamma_{\mathrm{SB}}(t) = \sigma\sqrt{t(1-t)}$
Normalize ϕ_0, ϕ_1; $\quad v = \mathrm{Log}_{\phi_0}(\phi_1)$; $\quad \mu_t = \mathrm{Exp}_{\phi_0}(t\, v)$
$u_{\mathrm{true}} = \dfrac{d}{dt}\mathrm{Exp}_{\phi_0}(t\, v)$ ▷ tangent at μ_t
if $s =$ `geo_det` **then**
    $\phi_t = \mu_t$
else if Ambient methods then
    if $s =$ `geo_amb_const` **then**
        $\phi_t = \mu_t + \sigma\xi$
    else if $s =$ `geo_amb_sb` **then**
        $\phi_t = \mu_t + \gamma_{\mathrm{SB}}(t)\xi$; $\quad \mathrm{denom} = \max\{2t(1-t), \epsilon\}$
        $\mathrm{corr} = (\phi_t - \mu_t)/\mathrm{denom}$
        $u_{\mathrm{true}} \leftarrow u_{\mathrm{true}} + (1-2t)\,\mathrm{corr}$
    end if
else if Tangent methods then
    if $s =$ `geo_tan_const` **then**
        $\xi_\top = \xi - \langle \xi, \mu_t \rangle \mu_t$
        $\phi_t = \mathrm{Exp}_{\mu_t}(\sigma\xi_\top)$
    else if $s =$ `geo_tan_sb` **then**
        $\xi_\top = \xi - \langle \xi, \mu_t \rangle \mu_t$
        $\delta = \gamma_{\mathrm{SB}}(t)\xi_\top$; $\quad \phi_t = \mathrm{Exp}_{\mu_t}(\delta)$
        $\mathrm{denom} = \max\{2t(1-t), \epsilon\}$
        $u_{\mathrm{true}} \leftarrow u_{\mathrm{true}} + (1-2t)\,\delta/\mathrm{denom}$
    end if
end if
return $(\phi_t, t, u_{\mathrm{true}})$

Algorithm 3 Inference / Cleanup Sampling Procedure

Require: Trained field $v_\theta(\phi, t)$, step count K, step size $\Delta t = 1/K$

Initialize $\phi_0 \sim p_0$ ▷ Sample from base distribution (uniform on $\mathbb{S}^d$)
for $k = 0$ to $K - 1$ **do**
    $v_k = v_\theta(\phi_k, t_k)$
    if Euclidean variant **then**
        $\phi_{k+1} = \phi_k + \Delta t\, v_k$
        $\phi_{k+1} \leftarrow \phi_{k+1}/\|\phi_{k+1}\|$ ▷ Renormalize to stay on $\mathbb{S}^d$
    else if Geodesic variant **then**
        $\phi_{k+1} = \mathrm{Exp}_{\phi_k}(\Delta t\, v_k)$
    end if
end for
return ϕ_K ▷ Cleaned SSP sample

3.6 Experimental Setup and Method Variations

This section describes all implementation choices required to reproduce the proposed cleanup framework, including data generation, network architecture, loss functions, coupling solvers, and training configurations.

3.6.1 Model Architecture

All experiments use a residual multilayer perceptron (MLP) to parameterize the velocity field $v_\theta(\phi, t)$. The model takes as input an SSP vector $\phi \in \mathbb{R}^d$ and, optionally, a time embedding $t \in [0, 1]$.

To effectively condition the network on the continuous time variable, t is mapped to a high-dimensional vector using sinusoidal positional embeddings [49].

The network architecture is composed of residual blocks, where each block consists of two fully connected layers with GELU activation, dropout, and layer normalization to stabilize optimization. The input to the network is the concatenation of the state vector and the time embedding:

$$v_\theta(\phi, t) = \mathrm{ResMLP}(\big[\phi; \mathrm{SinusoidalPosEmb}(t)\big]). \tag{3.22}$$

3.6.2 Time Conditioning and Unknown Corruption Levels

Flow matching and diffusion models typically use a scalar time input $t \in [0, 1]$ to indicate the sample's position along the trajectory connecting the source distribution p_0 to the target p_1. Effectively, t signals to the network how much of the original signal has been displaced by noise. However, in this work, the goal is to denoise corrupted SSPs where the initial magnitude of that displacement (the noise level) is unknown.

Sun et al. [47] explore the need for time conditioning during training. They demonstrate that under equal model size, removing explicit noise conditioning yields minimal or even positive performance changes. They further show that an auxiliary noise-level regressor, trained to predict a continuous or discrete noise indicator, provides only marginal benefit. These findings are also supported by Balcerak et al. [4] in Energy Matching and Wang and Du's Equilibrium Matching [50].

Two explanations are proposed for why explicit time conditioning may be unnecessary: The first is that the noisy sample input to the model already contains sufficient information about the corruption level. In this view, the model can implicitly learn to infer the time step t from the degree of corruption in its input, making a separate time variable redundant. The second possibility is that the model learns a simpler dynamic, such as a scalar field, which is less dependent on an explicit time parameter. Given this theoretical and empirical support, explicit time conditioning is not used in this work.

3.6.3 OT and SB Coupling Solvers.

All coupling distributions described in Section 3.3 are implemented using the `Python Optimal Transport (POT)` library [18]. The exact OT coupling π^* is computed via the `ot.emd2` solver, while the entropy-regularized variant π^*_ε is obtained through `ot.sinkhorn`. Each minibatch is solved independently under the angular ground cost $c_{\text{ang}}(\phi_0, \phi_1) = \arccos(\phi_0^\top \phi_1)^2$. The `emd2` method scales as $\mathcal{O}(n^3)$ but remains feasible for typical batches of $n \leq 256$, whereas the differentiable Sinkhorn implementation operates in $\mathcal{O}(n^2)$ memory.

3.7 Baselines

In addition to the proposed flow-matching variants introduced above, three baseline cleanup methods are included for comparison: the feedforward method described in Section 3.4, a grid lookup, and an optimization in the embedded space. These serve as reference models

for evaluating denoising accuracy, computational cost, and scalability under identical data and training conditions.

3.7.1 Grid and Optimization-Based Cleanup

As detailed in Chapter 2, both of these methods explicitly search the spatial domain for the location whose encoded SSP best matches a corrupted input under cosine similarity:

$$\hat{x} = \arg\max_{x \in \mathcal{D}} \langle \phi(x), \tilde{\phi} \rangle.$$

Grid Lookup Discretizes $\mathcal{D}$ into a finite grid $\mathcal{G}$ and evaluates similarity at all points. This guarantees that $\hat{\phi} = \phi(x^*)$ lies on the valid manifold but incurs exponential cost in grid resolution, limiting scalability.

Optimization-Based Cleanup Begins with a coarse grid lookup output x_{init} and refines it through continuous optimization in the spatial domain. To improve efficiency, updates are performed using the Limited-memory BFGS with Bound constraints (L-BFGS-B) algorithm, which approximates the inverse Hessian using a small history of curvature pairs rather than storing the full matrix [34]. This quasi-Newton solver provides near–second-order convergence while maintaining linear memory cost in dimension. Because the SSP manifold is non-convex, however, convergence to local maxima can occur given a poor starting point.

Algorithm 4 Grid and Optimization-Based Cleanup

Require: Noisy input $\tilde{\phi}$, encoder $\phi(x)$, grid $\mathcal{G}$, domain $\mathcal{D}$, method flag $\in \{\texttt{grid}, \texttt{optim}\}$

Compute similarity scores $s(x) = \langle \phi(x), \tilde{\phi} \rangle$ for all $x \in \mathcal{G}$
$x_{\text{init}} = \arg\max_{x \in \mathcal{G}} s(x)$
if method $=$ `grid` **then**
    return $\hat{\phi} = \phi(x_{\text{init}})$
else
    Initialize $x_0 = x_{\text{init}}$
    for $k = 1$ to n_{iter} **do**
        $\nabla_x \langle \phi(x_k), \tilde{\phi} \rangle \leftarrow$ gradient via autodiff
        $x_{k+1} = \text{Proj}_{\mathcal{D}}\big(x_k + \alpha_k \, H_k^{-1} \nabla_x \langle \phi(x_k), \tilde{\phi} \rangle\big)$ ▷ L-BFGS-B update
    end for
    return $\hat{\phi} = \phi(x_{n_{\text{iter}}})$
end if

3.8 Inference Time and Memory Complexity

Grid Lookup. For each corrupted input, this method computes similarity with every pre-encoded vector $\phi(x)$ across the discretized grid $\mathcal{G}$. Its runtime and memory both scale as $\mathcal{O}(|\mathcal{G}|\, d)$.

Crucially, the grid size is determined by $|\mathcal{G}| = \text{res}^k$, where res is the resolution (points per axis) and k is the dimensionality of the embedded space (spatial domain). Consequently, the computational cost grows exponentially with the spatial domain dimension, making this method prohibitive at fine resolutions.

Optimization-Based Cleanup. Each corrupted input is first mapped to a coarse initialization x_0 by searching a precomputed low-resolution grid $\mathcal{G}_{\text{coarse}}$, requiring $\mathcal{O}(|\mathcal{G}_{\text{coarse}}|\, d)$ memory. The initialization is then refined by m L-BFGS-B iterations, each costing $\mathcal{O}(d)$, giving total runtime $\mathcal{O}(m\, d)$. Because only a limited history of curvature pairs is retained, memory grows linearly with d rather than quadratically.

Feedforward and Flow Networks. Both the direct feedforward cleanup model and the flow-matching formulations are parameterized by the same residual MLP. Each block applies two linear layers whose widths scale with d, leading to $\mathcal{O}(d^2)$ operations per forward

pass. Feedforward requires a single network call ($\mathcal{O}(d^2)$ time and memory), whereas flow models integrate an ODE with K solver steps, resulting in $\mathcal{O}(K\,d^2)$ time and $\mathcal{O}(d^2)$ memory. The additional exponential/log map computations for Geodesic flows add only $\mathcal{O}(d)$ per step, negligible compared to network evaluation.

Table 3.1: Asymptotic inference-time and memory complexity of cleanup methods.

Method	**Time**	**Memory**
Grid Lookup	$\mathcal{O}(\text{res}^k\,d)$	$\mathcal{O}(\text{res}^k\,d)$
Optimization	$\mathcal{O}(m\,d)$	$\mathcal{O}(\lvert\mathcal{G}_{\text{coarse}}\rvert\,d)$
Feedforward	$\mathcal{O}(d^2)$	$\mathcal{O}(d^2)$
Flow Models	$\mathcal{O}(K\,d^2)$	$\mathcal{O}(d^2)$

The efficiency hierarchy depends heavily on the problem regime. For low-dimensional spatial domains ($k = 2$) with coarse resolution, Grid Lookup is often fastest. However, as spatial requirements scale, Grid Lookup becomes intractable due to the exponential term res^k.

Among the scalable methods, Feedforward cleanup is generally the fastest in the standard VSA regime explored in this work (where $d \approx 512 - 1024$), as it requires only a single pass. However, its $\mathcal{O}(d^2)$ complexity means that for extremely high-dimensional vectors (e.g., $d > 4000$), Optimization-based cleanup ($\mathcal{O}(m\,d)$) may become theoretically more efficient, provided the solution converges in few iterations ($m \ll d$).

It is important to note that the $\mathcal{O}(d^2)$ scaling of the neural methods arises from the specific choice of MLP architecture, where layer width scales linearly with input dimension d. This architecture was not optimized for computational efficiency in this work. Future studies could explore alternative architectures to achieve better scaling properties with respect to dimensionality.

3.9 Summary

The proposed cleanup framework can be understood as a hierarchy of feedforward and flow-based transport models that progressively integrate more structure into the mapping from corrupted to clean SSPs. Each variant modifies one or more of the three fundamental ingredients introduced in Section 3.5: the joint distribution over sample couplings

$\pi(\phi_0, \phi_1)$, the interpolant $I(t, \phi_0, \phi_1)$, and the noise schedule $\gamma(t)$. Table 3.2 summarizes these configurations and their defining characteristics.

Table 3.2: Summary of model variants and configuration choices.

Variant	**Coupling**	**Interpolant**	**Noise Schedule**
I-CFM	Independent	Linear	None
OT-CFM	Optimal π^*	Linear	Constant σ
SB-CFM	Entropic π_ε^*	Linear	$\sigma\sqrt{t(1-t)}$
Geo-CFM	I or OT or SB	Geodesic	None or Constant σ or $\sigma\sqrt{t(1-t)}$

In practice, the Independent CFM (I-CFM) and Optimal Transport (OT-CFM) represent two extremes of the coupling spectrum. I-CFM serves as the high-entropy baseline, providing direct linear interpolation between randomly paired endpoints. Conversely, OT-CFM enforces strict deterministic alignment, yielding minimum-energy trajectories. The Schrödinger Bridge variant (SB-CFM) acts as the continuum between these two regimes (controlled by ε).

Finally, the Geodesic CFM (Geo-CFM) is orthogonal to these choices, constraining the motion itself to the hyperspherical manifold to ensure the flow respects the intrinsic topology of the representation space.

This chapter formalized cleanup as a continuous-time generative transport problem between corrupted and clean SSP distributions. By framing denoising as flow matching under different coupling and geometric constraints, the framework unifies deterministic, stochastic, and Riemannian transport within a single velocity-based formulation. Subsequent chapters will evaluate how these variants differ empirically in terms of reconstruction accuracy and robustness to various corruption regimes.

Chapter 4

Benchmarks and Results

This chapter evaluates the proposed framework across noise levels, embedding dimensionalities, length scales, and model variants. Performance comparisons against classical baselines are presented first, followed by the main evaluation of flow-based restoration. Finally, we present analyses of computational efficiency and hyperparameter sensitivity studies on coupling distributions, stochasticity, geometry, and length scale.

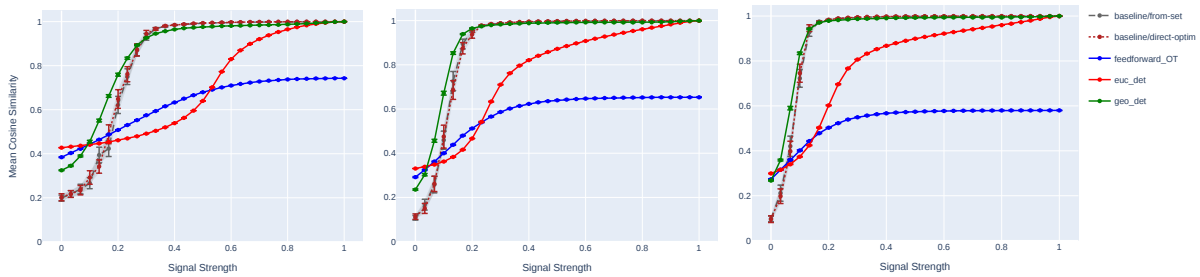
4.1 Experimental Setup

Unless otherwise specified, all models were trained using the Adam optimizer with a learning rate of 1×10^{-4}, batch size 256, and for 200 epochs. Each experiment used 20,000 paired samples drawn from the source and target distributions p_0 and p_1 described in Section 3.1. Training employed the cosine-similarity objective introduced in Eq. 3.5.

4.2 Main Results: SSP restoration under noise

This section presents the primary quantitative evaluation of the proposed cleanup framework by measuring denoising accuracy as a function of input noise level. Signal strengths $s \in [0, 1]$ were swept in increments of 0.05, interpolating between corrupted and clean SSPs either linearly or Geodesically depending on the model variant. Each configuration was evaluated over 200 bootstrap resamples to estimate 95% confidence intervals on the mean cosine similarity.

In the plots below, evaluation performance is reported for models trained on SSPs with a low length scale ($l = 0.1$) and moderate embedding dimensionality ($d = 487$). This represents a typical setting in downstream tasks: Small length scales yield high spatial resolution in the encoded domain, while a moderate dimension balances representational capacity and simulation cost.



(a) Dimension 151 (b) Dimension 487 (c) Dimension 1015

Figure 4.1: Mean cosine similarity ± 95% CI across noise levels. Higher dimensions (right) sustain performance at higher noise levels compared to lower dimensions (left). The Geodesic Flow model is more noise robust than other alternatives, including non-neural baselines. (Note: Error bars/shaded regions are present but may be smaller than the plot lines due to low variance.)

4.2.1 General Trends

Under the impact of noise, the phase alignment and rotational symmetry of an SSP vector are damaged. The method best capable of reversing these effects should in theory be capable of sustaining the impact of high noise and recovering the initial clean SSP. Across all dimensionalities and models, accuracy decreases as noise approaches 100%. Although trained models are all generative, and capable of generating clean SSP samples from pure noise, the evaluation interpolates between randomly sampled noise to SSPs. At the pure noise end, there is no correspondence to the SSP endpoint, hence the drop in performance. However, onset and severity of this decline depend strongly on both the model architecture and the embedding dimension. Higher-dimensional SSPs exhibit increased robustness and maintain high similarity values until larger corruption levels, as can be seen in the point of performance drop across the 3 plots of Figure 4.1.

Geodesic Flow The Geodesic flows developed above consistently achieve the highest performance across all noise levels, dimensions, and length scales. This is likely due to the consideration of the intrinsic geometry of the manifold when the velocity field is trained. Since the interpolation is restricted to the surface of $\mathbb{S}^{d-1}$, angular structure of the noisy vector is preserved and phase coherence can be restored. This geometric consistency allows Geodesic flows to recover structured SSP representations even at high noise levels where Euclidean flows and feedforward models deteriorate.

Euclidean Flow The Euclidean flows achieve high reconstruction accuracy only in low noise scenarios. At weak levels of corruption, the linear trajectory between a noisy ϕ_t and clean ϕ_1 remains close to the surface of the hypersphere. This enables the learned field to map a corrupted vector back to a clean state. However, as noise increases, a Euclidean interpolant will more often cut through the center of the hypersphere, rendering denoising unstable. Geometrically, the midpoint of the trajectory moves closer to the origin, causing the vector magnitude $\|\phi_t\|$ to drop significantly. Since SSP phase information is encoded in the complex arguments of the Fourier components, this magnitude collapse destroys the phase structure required to define a location.

Near the origin, the directionality of the vector field becomes ambiguous and unstable. Consequently, the flow drifts, and reconstructs a vector in the general neighborhood of the original, but increasingly far from it as the noise level rises. The comparative effect between the 2 types of flows can be seen in Figure 4.2.

Feedforward-OT This model trains on minibatch optimal transport couplings, which aligns noisy and clean samples to minimize transport cost. This imposes a coherent structure on the dataset that allows the network to generalize beyond endpoints and perform non-trivial denoising. In high-noise regimes, a corrupted input may plausibly correspond to multiple clean locations (multi-modality). The expected value of these modes is a superposition, or bundle, rather than a single valid SSP. As a result, the feedforward output often manifests as a "blurry" vector containing artifacts of multiple locations (visible in the diffuse similarity maps in Figure 4.3d), failing to commit to a single precise coordinate.

Qualitative Behaviour via Similarity Maps The quantitative trends discussed above are reflected in the structure of the similarity maps produced by each model. Figures 4.3a–4.3d visualize the cosine-similarity landscape over the spatial domain for a representative SSP corrupted with 60% noise. The Geodesic flow produces a sharply peaked map centered precisely at the true location, with minimal secondary structure. This indicates

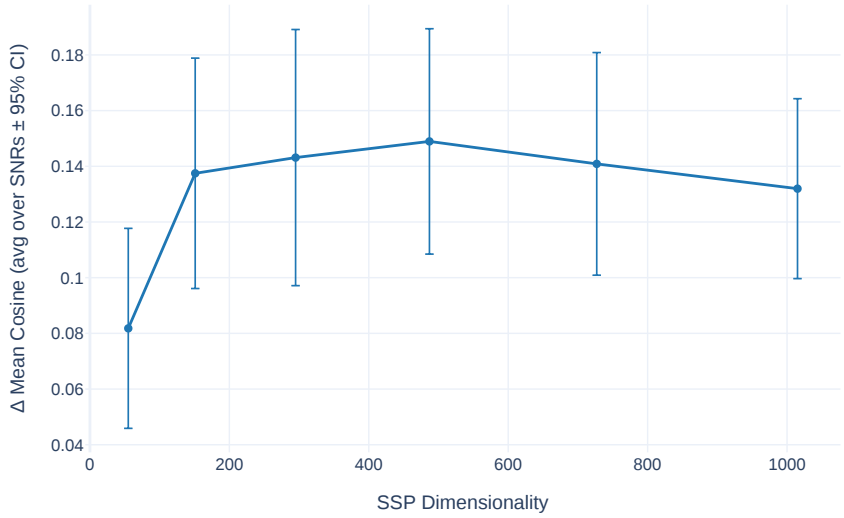
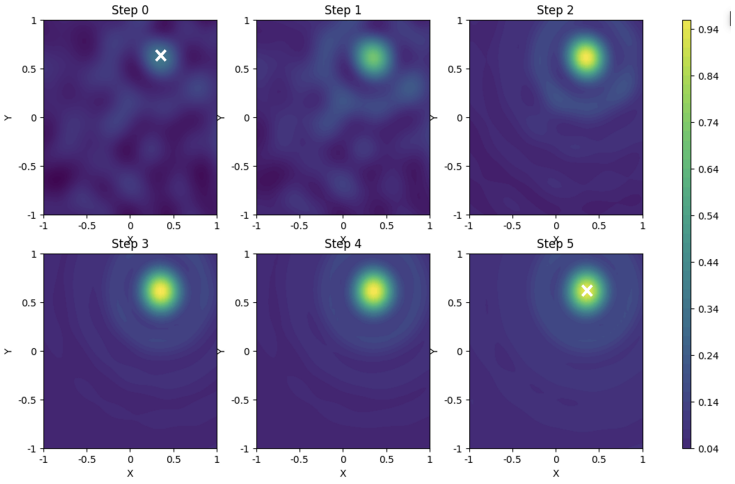


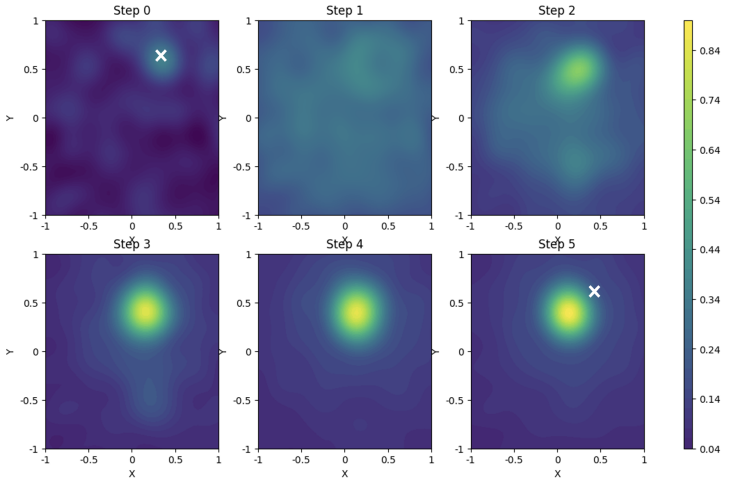
Figure 4.2: Performance benefits of Geodesic interpolants relative to Euclidean (Δ mean cosine $\pm$ 95% CI) vs dimensionality.

that the model not only reconstructs the correct point on the manifold but also recovers the characteristic interference pattern of the underlying hexagonal SSP basis. The Euclidean flow, in contrast, yields a similarity map whose dominant peak is shifted relative to the ground truth. This follows directly from the off-manifold interpolation path: the trajectory cuts through the interior of the sphere, causing the flow to settle near the intended Geodesic target due to loss of angular information.

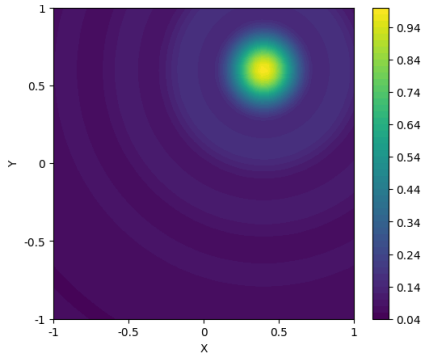
Finally, the feedforward OT variant produces a diffuse, broadened similarity map with multiple moderate-probability regions, reflecting the global nature of the learned mapping.



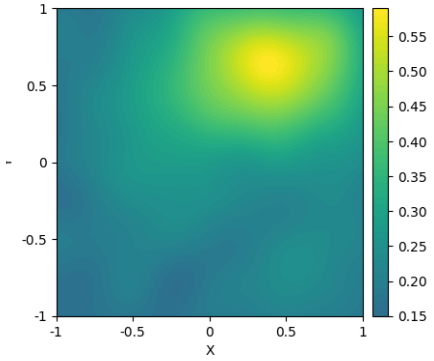
(a) Geodesic Flow



(b) Euclidean Flow



(c) Ground Truth (Target)



(d) Feedforward OT

Figure 4.3: Qualitative comparison of inference outputs. (a) and (b) show the iterative evolution of the flow models. Note that the Geodesic Flow (a) maintains a sharp peak centered on the target, while the Euclidean Flow (b) drifts. (c) The ground truth target. (d) The direct feedforward prediction, which exhibits a diffuse similarity map due to the averaging effects of the regression objective.

4.2.2 Baseline Cleanup performance

To better understand the above comparison between flow models and the baselines, it is essential to characterize the performance of classical SSP cleanup methods (grid lookup

and optimization-based cleanup). These methods do not learn transport fields; instead, they explicitly search for the spatial coordinate whose SSP encoding best matches the corrupted input under the dot product. As such, their failure mode occurs under high noise, where all SSPs have roughly the same similarity. These non-neural methods provide reference points for evaluating the neural cleanup models.

Grid Lookup Various grid resolutions ranging from coarse (4×4) to fine (256×256) were evaluated. Figure 4.4 displays the performance for resolutions up to 64×64; higher resolutions (128×128 and 256×256) yielded no performance gains and are omitted for clarity.

As expected, performance increases smoothly with grid resolution. Coarse grids quantize the space too aggressively, leading to misidentification even at low noise levels (reaching a low performance plateau). Intermediate grids (16×16 and 32×32) perform reasonably well but still undersample the domain to produce an output close enough to the ground truth SSP. At 64×64, lookup is sufficiently dense to recover the correct location perfectly across most noise levels, achieving the best possible accuracy at a length scale of 0.1. Beyond this point, higher grid resolutions become computationally demanding with diminishing returns. Therefore, across all experiments, the 64×64 resolution was used as the gold-standard baseline.

Direct Optimization Starting from a coarse 4×4 grid initialization, this baseline achieves performance parity with the 64×64 grid. The coarse grid identifies the correct basin of attraction, and gradient refinement locates the nearby optimum. While this achieves high accuracy with low memory, it comes at the cost of extended iterative computation.

Like the grid lookup method, both baselines fail in the high noise regime. As the noise vector dominates, the correlation with the true SSP is attenuated until it becomes indistinguishable from the noise floor. In this regime, the similarity landscape is dominated by spurious correlations, causing the argmax operation to select random locations where the noise vector coincidentally aligns with the manifold.

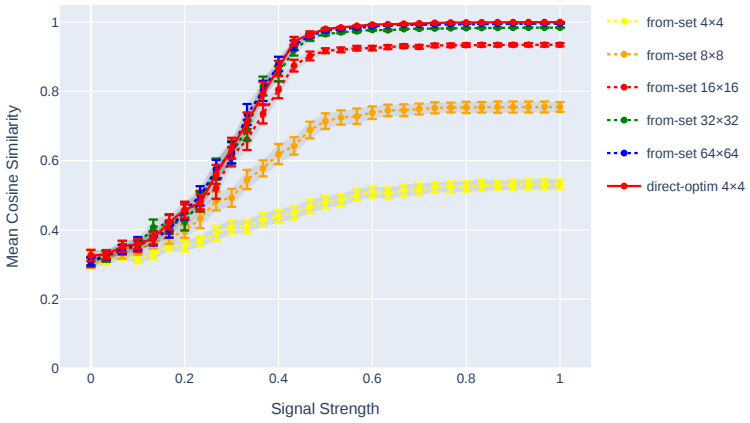


Figure 4.4: Performance over different resolutions for grid and optimization baselines.

4.2.3 Performance Relative to Baselines

This section quantifies the performance gap between the trained models and the 64×64 grid-lookup baseline by averaging cleanup performance over SSP dimensionalities and length scales. The grid-lookup method performs perfectly until the aforementioned drop induced by high noise. For clarity, performance observations from Figure 4.5 are split into 3 sections for low, medium, and high noise regimes.

Low noise At low noise, the flows do as well as the grid lookup: given a perfect SSP, the trained velocity fields serve as an attractor near the clean point on the manifold. This is consistent with the observations in Section 4.2, where both Euclidean and Geodesic flows reach cosine similarity ≈ 1 as noise approaches zero. The feedforward model with OT couplings reaches a certain plateau of performance. Typically, results under similarity maps show smoothed peaks, despite the SSP angular structure still being preserved.

Medium noise Between $\sim 40\% - 70\%$ noise, the Geodesic matches the baseline performance. Euclidean flows, however, perform on average 20% worse. This is likely due to the loss of angular information during a straight trajectory through the hypersphere, which introduces ambiguity in the target regions. The feedforward model suffers from a similar effect: despite aligned couplings, the model is unable to generate local regularization of inference trajectories in the intermediate space between distributions, leading to incorrect cleaned up locations.

High noise At the pure noise end, all models perform 10-20% better than the baseline. At this point, performance should be more influenced by random chance; any noise can correspond to the SSP target. This is highlighted by the performance of the feedforwad model with no coupling structure. As seen in Figure 4.6, the model output has high similarity to the full domain. Other models exhibit similar artifacts to different extents, alternating between generating a random clean SSP and a vector with high similarity across the domain. At this noise level, the cleanup task is ill-defined, since many outputs are possible for a given input. The significance of this noise level is that network based outputs can have higher similarity to the ground truth without necessarily being an SSP, which partially explains the higher performance compared to baseline. Notably, the feedforward model with no coupling structures overfits to the mentioned output, regardless of noise level.

When the noise level is at 0-25%, the Geodesic flow continues to outperform the baseline by 15% on average. This effect may be explained by the fact that angular denoising in the embedding space directly can yield some benefit compared to a decoding based similarity calculation approach, where the similarity landscape is flat and the max function picks with peaks not corresponding to a ground truth output.

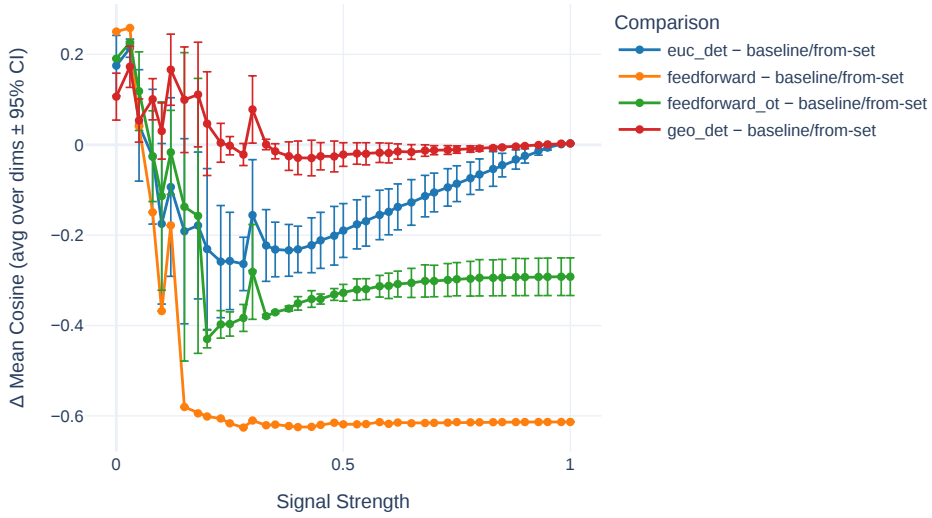


Figure 4.5: Mean cosine similarity ± 95% CI across Noise levels averaged over dimensions showing performance relative to grid cleanup baseline.

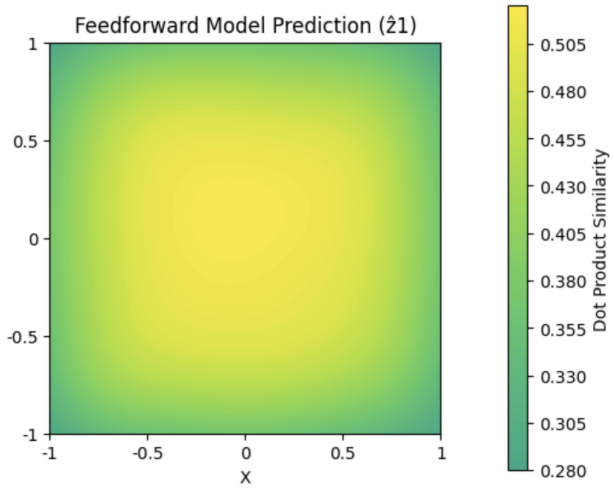


Figure 4.6: Similarity map for a feedforward model trained with random couplings. The model collapses to an "average" SSP-like vector, failing to recover the correct location.

4.3 Effect of Step Count

Since flow models learn a dynamic mapping across the distributions, it is worth exploring average performance as a function of inference step count, especially for the non-linear Geodesic model. In principle, in a perfectly learned flow field, increased numerical resolution (i.e., more ODE steps) improves integration accuracy, though at the cost of runtime. Models are evaluated at varying step counts ($K = [1, 25]$) to quantify the tradeoff between accuracy and computational cost, using Euler updates for Euclidean flows and exponential-map updates for Geodesic flows (Algorithm 3).

To isolate the effect of step count from noise, results are averaged over all signal strengths, producing a single performance curve per model (Figure 4.7).

Euclidean Flows Surprisingly, the Euclidean flow exhibits near-identical performance across all evaluated step counts. Using a single step ($K = 1$) performs essentially as well as using $K = 25$. This robustness is a direct consequence of the Flow Matching training objective, which regresses a vector field onto straight-line conditional trajectories ($\phi_t = (1 - t)\phi_0 + t\phi_1$).

Theoretically, Optimal Transport (OT) coupling should improve 1-step performance by "straightening" the global vector field (reducing trajectory crossings). However, we observed that even Independent (random) coupling achieved optimal 1-step reconstruction. We hypothesize this is due to the high dimensionality of the vector space ($d \geq 512$). In such spaces, randomly sampled vectors are quasi-orthogonal with high probability (the "Blessing of Dimensionality"), meaning that even randomly paired trajectories rarely intersect. Consequently, the learned vector field is globally simple and locally constant, allowing a single Euler step to recover the target almost perfectly regardless of the coupling method.

Geodesic Flows In contrast, the Geodesic model benefits substantially from increased step count, saturating around $K = 5$. This behaviour reflects the curvature of the underlying manifold: The target Geodesic trajectory between ϕ_0 and ϕ_1 follows a great-circle arc. When the learned field approximates this curvature, smaller step sizes better resolve the nonlinearity of the path. Because the Riemannian flow matching training regresses tangent-space velocities, the learned field is intrinsically smooth.

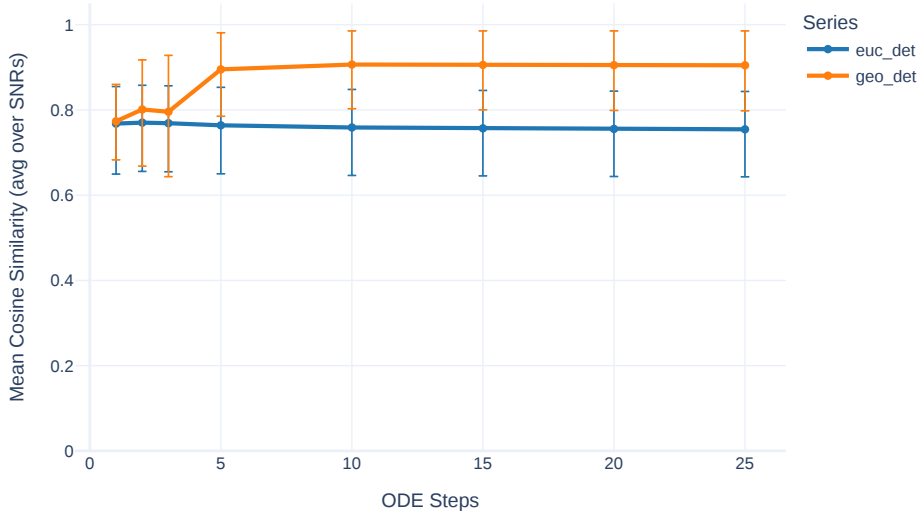


Figure 4.7: Average cosine similarity (±95% CI) as a function of inference step count K, averaged over noise levels. Geodesic flows benefit from multi-step integration, while Euclidean flows do not.

4.4 Compute and Memory Requirements

A central motivation for learning-based cleanup is efficiency: grid-based lookup requires large memory footprints, while optimization-based cleanup incurs substantial compute costs. This section compares the computational and memory requirements of the baseline methods with the network-based approaches. As mentioned in Section 3.6.1, the different model variants are network agnostic. Grid lookup leverages a 64×64 grid, while the direct optimization method uses a 4×4 coarse initialization. In terms of memory, for a given SSP dimension, the model size is fixed regardless of method (whether it be feedforward, or any of the flows). In terms of compute, the same applies. However, the number of forward passes is also factored in the computation. All runtime experiments were conducted on CPU using a batch size of 1, ensuring a direct comparison between iterative optimization and neural ODE integration without leveraging GPU parallelism.

4.4.1 Memory footprint

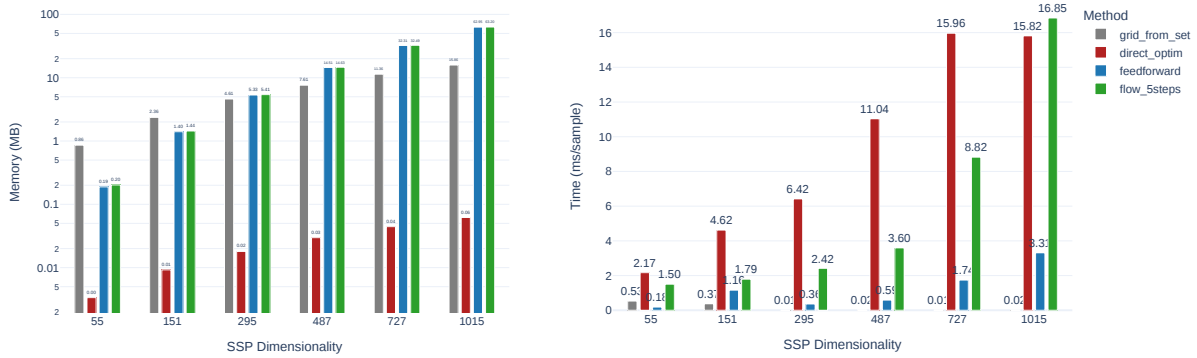
Figure 4.8a reports memory usage as a function of SSP dimension. A 64×64 grid stores 4096 SSPs, each of dimension d. Memory therefore grows linearly with d, and exponentially in grid resolution, but in these experiments the grid resolution itself is fixed and large. A similar trend is shown for the direct optimization method, where the memory is taken up by the coarse grid as well as a number of samples in the embedded dimension (2) to guide the gradient of the optimization (negligible footprint).

The memory footprint of flow and feedforward models scales with the parameter count of the MLP used to model the velocity field or direct mapping. To maintain sufficient representational capacity at higher SSP dimensionalities, the network size was scaled with d. As a result, neural models typically match or slightly exceed the memory footprint of the 64×64 grid baseline at high dimensions. It should be noted that the memory–accuracy tradeoff in model architecture scaling was not optimized, and future work should investigate lightweight architectures tailored for SSP cleanup.

4.4.2 Inference runtime

Figure 4.8b reports CPU inference time as a function of SSP dimension. Optimization-based cleanup is the slowest across nearly all dimensions, requiring repeated evaluations of SSP encodings and iterative gradient-based refinement. This overhead grows rapidly as dimensionality increases, leading to higher runtimes than any other method in all but the

largest tested dimension where compute is matched. Comparatively, the 1 step methods (Euclidean flow and feedforward) are over an order of magnitude faster on average, and the grid based lookup is even faster.



(a) Memory footprint (log scale) vs SSP dimensionality.

(b) Inference time vs SSP dimensionality.

Figure 4.8: Computational scaling. (a) Neural models scale similarly to the grid baseline in memory. (b) Optimization-based cleanup (red) incurs significantly higher runtime costs as dimensionality increases, while feedforward and flow models (mostly) remain relatively efficient.

4.5 Hyperparameter Sensitivity

This section evaluates several factors influencing model behaviour: coupling distributions, stochasticity in the interpolant, feedforward sensitivity to coupling, geometric constraints, and manifold structure. Where relevant, results are averaged across length scales and SSP dimensionalities to highlight general trends.

4.5.1 Effect of Stochasticity and Coupling Distributions

Both Euclidean and Geodesic flows can be trained using either:

- Exact OT couplings, computed using the POT (Python Optimal Transport) EMD solver (`ot.emd`), modified to obtain minimum-cost matchings under the angular distance.

- Entropic Schrödinger bridge (SB) couplings, obtained via the POT Sinkhorn algorithm (`ot.sinkhorn`), which adds entropy regularization controlled by the Sinkhorn temperature λ.

In either case, training includes interpolant noise via the method dependant variance schedule $\gamma(t)$, scaled by the hyperparameter σ. The combined effects of coupling choice and injected stochasticity are evaluated below.

Effect of noise scale σ Across all tested models and dimensions, the smallest σ consistently yielded the best performance. As it increases, accuracy visibly degrades. This reflects the unique structure of the SSP manifold: Noise thickens the target distribution around the interpolant. For natural image manifolds or other unstructured priors, this often improves sample diversity and robustness. For SSPs, however, the manifold is extremely structured. Adding noise effectively teaches the model to map toward a shell around the target, blurring the geometric structure needed for precise location recovery. In the SB case, although $\gamma(t) \to 0$ as $t = 1$, the entropic regularization in the Sinkhorn plan blurs the couplings, causing similar degradation.

An example can be seen in Figure 4.9 with different values used to train the geo-amb-sb model at a dimensionality of 487. It can be seen that the smallest sigma values result in the best model (which match the performance of no coupling at all as seen in Figure 4.1b).

4.5.2 Effect of OT vs SB couplings

The smallest σ value was used across all model variants. Relative to the Geodesic/Euclidean models trained with random couplings, there is no significant performance difference with introduced coupling structures (see Figures 4.10 and 4.11). This is likely because flow matching's regression objective already encourages local path consistency, and does not benefit from the additional structure imposed by either OT or SB couplings. The conditional velocity field at a point ϕ_t depends only on local geometric structure, and does not benefit from the specific pairing of ϕ_0 and ϕ_1. When conditioned on ϕ_t, the expected velocity $v_\theta(\phi_t)$ averages out the differences introduced by various couplings. As a result, the learned flow field reflects the geometry of the SSP manifold rather than the global alignment of endpoints. This explains why coupling choice has minimal impact on flow models, even though it is critical for feedforward models.

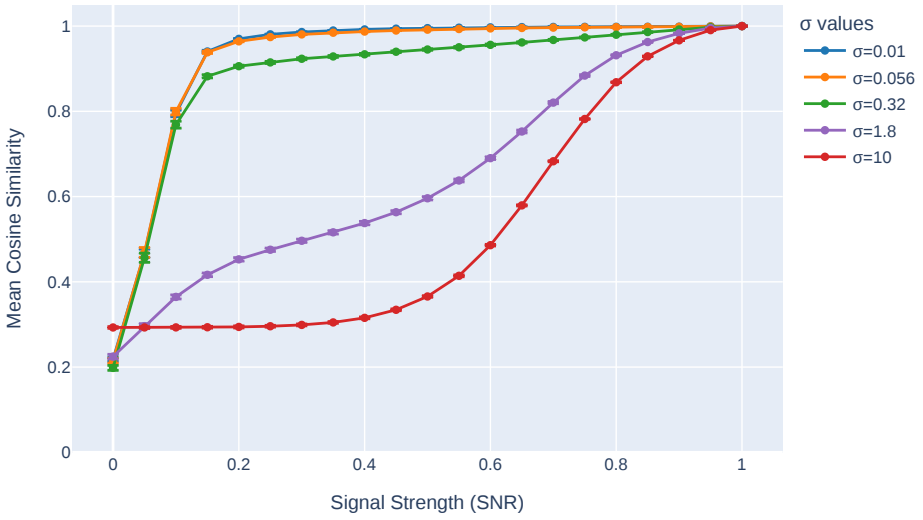


Figure 4.9: Effect of different sigma values between 0.01 and 10 on model performance. Introducing stochasticity degrades the structure of the target SSP

4.5.3 Feedforward Model Sensitivity

Feedforward models solve a fundamentally different learning problem than flows. While flow matching learns a *local* velocity field $u_\theta(\phi_t) = \dot{\phi}_t$ defined on intermediate states ϕ_t, the feedforward model attempts to learn a *global* mapping $f_\theta(\phi_0) = \phi_1$. This distinction creates dramatically different sensitivities to how training pairs (ϕ_0, ϕ_1) are constructed (see Figure 4.12).

Random Couplings (Discontinuity). Random couplings across the distributions yield a discontinuous learning problem. Because there is no geometric correspondence across the dataset, two nearby corrupted samples $\phi_0^{(i)}$ and $\phi_0^{(j)}$ may be assigned to entirely unrelated clean SSPs $\phi_1^{(i)}$ and $\phi_1^{(j)}$. As seen in 4.6, the feedforward model collapses to predicting the mean of the clean SSP distribution, i.e., $f_\theta(\phi_0) \approx \mathbb{E}_{\phi_1 \sim p_1}[\phi_1]$. This effect is observed irrespective of SSP dimensionality, length scale, or noise level at inference.

OT Couplings (Continuity). In contrast, OT couplings identify a globally optimal transport plan between p_0 and p_1. This yields a pairing that minimizes total transport cost across the dataset. Importantly, nearby corrupted samples are typically paired with

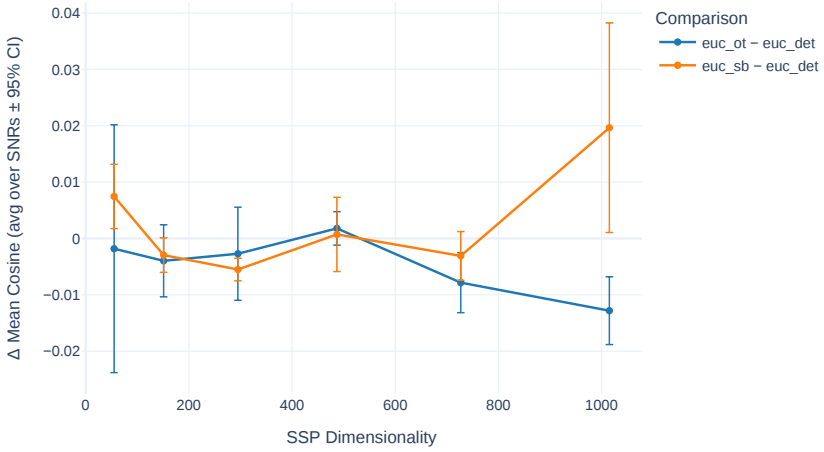


Figure 4.10: Relative performance benefits of sampling methods relative to deterministic Euclidean trajectories (Δ mean cosine $\pm$ 95% CI) vs dimensionality.

nearby clean SSPs. This converts the originally discontinuous mapping into a relatively geometrically aligned one, allowing the mapping f_θ to generalize to corrupted SSPs not explicitly seen during training. In effect, OT couplings reveal the "fiber structure" connecting corrupted and clean representations that is learned by the flows. However, as noted in Section 4.2, even with this continuity restored, the feedforward model is still limited by the "averaging" effect, preventing it from achieving the precision of generative flows.

Schrödinger Bridge Couplings (Smeared targets). Finally, Schrödinger Bridge couplings introduce entropy regularization. For large λ, the plan is excessively diffuse, giving rise to the same problem as random couplings. Any given starting ϕ_0 can lead to many ϕ_1's. The ambiguity created in the targets blurs the manifold structure for the network. For small λ, the Sinkhorn solver fails to converge and defaults to nearly random couplings. This effect was especially observed in higher dimensions (where performance reaches a plateau indistinguishable from the random coupling baseline) .

4.6 Effect of Length Scale

The SSP length scale l plays a central role in determining the geometry of the encoded space. Smaller values of l correspond to higher spatial frequencies in the Fourier-domain

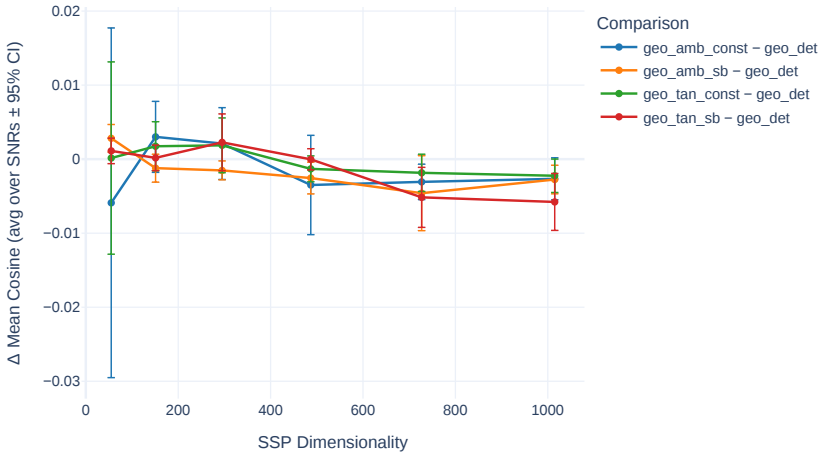


Figure 4.11: Relative performance benefits of on/off manifold sampling methods relative to deterministic Geodesic trajectories (Δ mean cosine $\pm$ 95% CI) vs dimensionality.

construction, producing strongly oscillatory representations in which nearby spatial points are sharply distinguished. In this regime, cosine similarity approximates a narrow sinc kernel: two SSPs corresponding to locations only slightly apart may have very low similarity unless they fall within the high-lobe region. While this property improves the representational resolution of SSPs, it also makes the task of cleanup harder. Models must learn to reconstruct phase relationships, and larger embedding dimensions amplify this challenge because the phases are encoded across more components. Consequently, both feedforward and flow models exhibit lower robustness at small length scales, with the feedforward model degrading most dramatically (see Figure 4.13).

As the length scale increases, the kernel broadens: nearby spatial points produce SSPs with higher cosine similarity and the interference pattern becomes smoother. This softening of the representational landscape reduces the difficulty of cleanup. All models (grids, optimization, flow models, and feedforward networks) achieve higher accuracy at 100% noise simply because any SSP is relatively similar to any other (see Figure **??** for effects of length scale on the similarity kernel). For flow models, the benefit of Geodesic constraints persists, but the overall difficulty of the task decreases. For the feedforward model trained with OT couplings, increasing l dramatically improves performance: the mapping from corrupted to clean SSPs becomes smoother and more amenable to approximation by a fixed-depth MLP, particularly in high dimensions where small-l representations exhibit extreme oscillatory structure. In contrast, the feedforward model trained with random

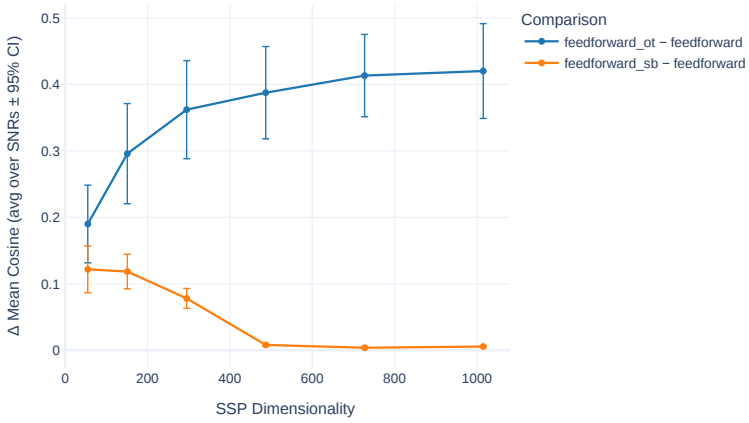


Figure 4.12: Relative performance benefits of coupling structure to random coupling on feedforward model performance (Δ mean cosine $\pm$ 95% CI) vs dimensionality.

couplings continues to collapse, but with a higher baseline similarity due to the broader kernel.

These trends collectively show that length scale interacts strongly with the structure of the SSP manifold and with model architecture. Small l emphasizes high-frequency content that is difficult for the feedforward networks but well handled by Geodesic flows. Large l simplifies the manifold, making all cleanup tasks easier at the cost of a less expressive embedding space.

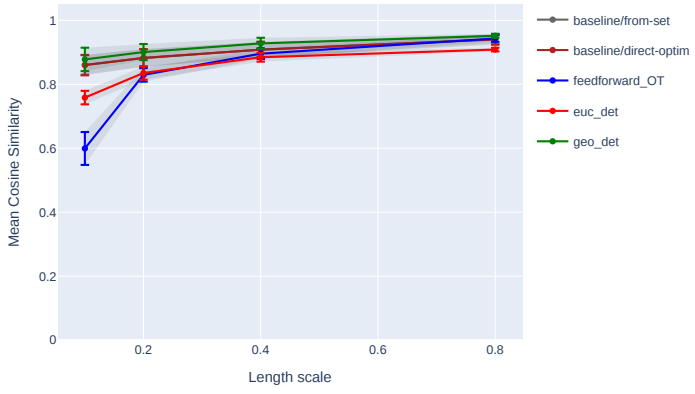


Figure 4.13: Average performance over length scales.

Chapter 5

Cleanup-Augmented SLAM: An Application of Geodesic Associative Memories

In the previous chapters, it was established that noise inevitably corrupts SSP representations at various levels of a cognitive pipeline. Specifically, sources relating to spiking variability in neural algorithms, VSA operations such as bundling and unbinding (introducing phase cross-talk noise), and phase noise in recurrently connected networks were analyzed for the nature of distortion effects (see Sec. 2.4). Additional desiderata for biological plausibility mandated maintaining representations in the embedding dimension during vector manipulation. This motivated the need for structured cleanup mechanisms entirely in the embedding space. In Chapters 3 and 4, Geodesic and Euclidean flows and a feedforward implementation were introduced and evaluated against decoding-based classical alternatives (grid lookup and direct optimization). It was shown that the Geodesic flow overall outperformed all methods across dimensions, length scales, and noise levels.

While these controlled experiments provide clear evidence of reconstruction improvements, they do not directly indicate whether such improvements translate to utility in a real algorithmic task that relies on stable SSP representations. Cleanup is meaningful only insofar as it improves downstream inference.

To address this gap, this chapter evaluates whether cleanup enhances the performance of the Spiking Semantic SLAM system introduced by Dumont et al. (2023) [11]. The structure of the chapter is as follows: First, the problem of simultaneous localization and mapping (SLAM) is introduced. This is followed by a description of the integration

strategy, focusing on the stabilization of the path integrator. Implementation details are then given. Finally, results are presented demonstrating that the proposed Geodesic Flow model significantly enhances robustness, reducing path error by up to 72% and allowing for a 40% reduction in neural resources while maintaining equivalent tracking accuracy.

5.1 The Algorithm

When navigating a new environment, agents are tasked with the simultaneous problem of tracking their own position in a dynamically evolving map through external cues. This requires reconciliation of noisy sensory and semantic information, uncertain motion estimates, and imperfect memory into an internal representation of the environment. In SSP-SLAM, SSPs are used to encode continuous maps, perform path integration using a noisy spiking representation, and learn associations between landmark representations and their positions.

Localization. The localization module maintains a continuously updated estimate of the agent's position, encoded as an SSP. This is implemented using a recurrent path-integration network built from velocity-controlled oscillators (VCOs) that approximate the ideal transport dynamics:

$$\dot{\phi}(x(t)) = \mathcal{F}^{-1} e^{iAx(t)} \odot iA\dot{x}(t) \tag{5.1}$$

These oscillators track the Fourier components of the SSP. As the agent moves, the network integrates velocity into an allocentric position estimate. However, due to the recurrent nature of the spiking network, this estimate accumulates phase drift and neural approximation error, causing the representation to drift off the valid SSP manifold.

Landmark Perception. The landmark-perception module transforms moment-to-moment sensory observations into allocentric feature locations. The model receives landmark identities and their egocentric directions/distances and represents these as SSP "object-vector" encodings. By binding the current pose estimate SSP with these egocentric vectors, the system computes allocentric SSPs for each landmark.

Environment map. The environment map is implemented as a heteroassociative memory network that learns associations between semantic pointers denoting objects/landmarks and their allocentric SSP locations. Learning occurs through biologically inspired rules to

store stable mappings. Over time, the memory accumulates a semantic–spatial map: given a landmark identity, the network can recall its estimated location; given a location, it can recall the features bound to it.

Loop closure. Loop closure aligns the drifting path-integration estimate with the map's stored spatial knowledge. When the agent revisits a known landmark, the system computes the discrepancy between the recalled position and the current PI-derived estimate: $\phi_{error} \approx \phi_{map} \circledast \phi_{PI}^{-1}$. This error signal is fed back to the path-integrator to correct its internal state.

5.2 Cleanup Usage within SSP-SLAM

As identified in Section 2.4, noise corrupts the neural algorithm at multiple stages. While cleanup could theoretically be applied at any readout point (e.g., querying the map or decoding loop closure signals), the **Path Integrator (PI)** is identified as the critical point of failure.

Drift in the PI is the root cause of map corruption: if the agent believes it is in the wrong location, it will write incorrect coordinates for landmarks into its memory. Therefore, the cleanup model is integrated as an online stabilizer for the PI. By ensuring the PI output remains on the valid manifold, it can be ensured that the downstream mapping and loop closure modules receive high-fidelity inputs.

5.2.1 Online Path Integrator Stabilization

The primary injection point for the cleanup model is the interface between the PI and the rest of the system. In the standard SSP-SLAM implementation, the recurrent VCO population produces a noisy estimate of the agent's position, $\tilde{\phi}(x_t)$. As the simulation progresses, phase drift causes this vector to degrade.

Critically, this noisy vector $\tilde{\phi}(x_t)$ is the primary input for the Landmark Perception module. To derive the allocentric location of an observed landmark (ϕ_{land}), the system must bind the agent's position with the egocentric object vector (ϕ_{obj}):

$$\phi_{land} \approx \tilde{\phi}(x_t) \circledast \phi_{obj} \tag{5.2}$$

When the noisy position estimate $\tilde{\phi}(x_t)$ is bound with the object semantic pointer, the noise propagates directly into the resulting landmark representation ϕ_{land}. Consequently, this

bound vector no longer accurately reflects the specific object-location. If this representation were queried (unbound) to determine which object resides at a given location, the result would either be a noisy approximation of the ground truth object or, in cases of significant drift, an entirely incorrect or unrecognizable identity.

To prevent this, the cleanup model intercepts the PI output. The cleanup operation then maps the noisy state back to the valid SSP manifold $\phi(\tilde{x}_t) = f_\theta(\tilde{\phi}(x_t))$. In the network implementation, this cleaned result drives the grid cell population, ensuring that the associative memory receives high-fidelity allocentric vectors even when the internal state of the PI begins to degrade.

Note that the cleanup operation does not reset the PI state. Instead, it ensures the environment map is accurately constructed. From there, loop closure can then update the PI state: when a landmark is revisited, the map recalls the *clean* location stored previously. This target is then compared against the drifting PI state to generate a precise error signal, which is fed back to the VCOs to correct for phase drift.

5.3 Implementation details

Experiments were run using the Nengo simulator, adapting Dumont et al.'s SSP-SLAM implementation [11] to include flow models. The experiments were designed to assess both the online stability of the path integrator and the offline accuracy of the learned map under varying regimes of representational precision.

5.3.1 Environment and Task

The agent navigated a random trajectory in a 2D environment bounded within a normalized region ($x, y \in [-1, 1]$) for $T = 60$ seconds. The environment contained $N = 50$ randomly distributed landmarks. Inputs included the 2D velocity signal $\dot{x}(t)$, landmark semantic pointers, and SSP displacement vectors.

Varying Representation Noise To evaluate neural resource efficiency, the number of neurons in the path integrator's recurrent population was varied across trials ($N_{neurons} \in [1000, 3000]$ at 500 neuron increments). Lower neuron counts result in faster accumulation of phase drift. This setup treats neuron count as a proxy for the signal-to-noise ratio (SNR) of the system.

Cleanup Configuration Two cleanup methods were compared:

1. **Grid Baseline:** A discrete lookup over a dense 64×64 grid of pre-encoded SSPs. This resolution was chosen to provide a near-ideal discretization of the space (as shown in Chapter 4).

2. **Geodesic Flow:** The Geodesic Flow Matching model introduced in Chapter 3. This specific variant was leveraged as it was identified as the highest-performing generative model.

5.4 Results

This section analyzes the performance of the different SLAM configurations. First, online tracking performance of the integrator was compared. Downstream consequences were then evaluated by analyzing the constructed environment maps.

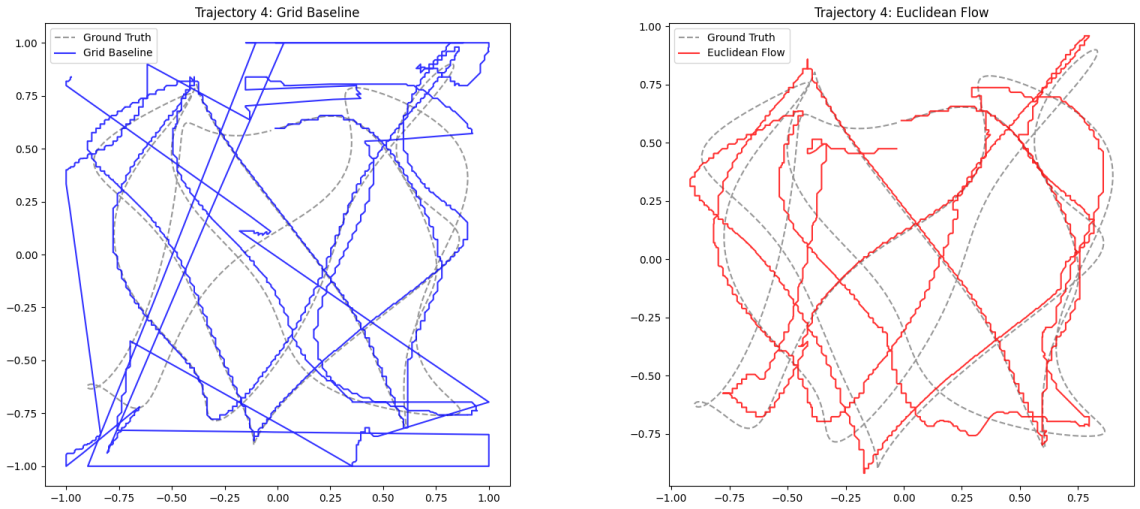
5.4.1 Localization Accuracy

The ability of the agent to localize itself while mapping the environment is the foundation of the SLAM system. Without cleanup, the PI relies entirely on the representational capacity of its recurrent population to maintain the SSP.

Figure 5.1 compares the ground truth trajectory against the PI output for a resource-constrained network ($N = 1500$). While both methods accumulate error over time (an expected outcome for open-loop integration), they exhibit fundamentally different failure modes.

The Grid baseline (Figure 5.1a, blue) tracks accurately when the internal error is low. However, once the accumulated drift pushes the state vector sufficiently far from the valid manifold, the method fails catastrophically. Because grid lookup operates by "snapping" the noisy state to the nearest stored prototype , a large deviation can cause it to lock onto a spatially distant, incorrect attractor. This results in discontinuous jumps in the position estimate, causing the agent to completely lose track of its trajectory (visible as the erratic, jagged path in the latter half of the simulation).

The geodesic flow (Figure 5.1b, red), on the other hand, is still able to maintain trajectory coherence under graceful degradation due to the continuous correction signal it provides through the learned velocity field.



(a) Grid Baseline

(b) Geodesic Flow (Ours)

Figure 5.1: Trajectory reconstruction for a resource-constrained Path Integrator ($N = 1500$ neurons). (a) The Grid Baseline fails to correct phase drift, resulting in catastrophic divergence where the agent's internal estimate (blue) completely detaches from the ground truth (dashed grey) after complex turns. (b) The Geodesic Flow model acts as a continuous attractor, successfully projecting the noisy state back to the valid manifold and maintaining a stable lock on the trajectory (red) throughout the simulation.

This divergence is further quantified in Figure 5.2, which plots the spatial error over the duration of the trajectory ($N = 5$ trials). While the Grid baseline suffers from cumulative drift, resulting in a mean trajectory Error of 0.249 meters, the Geodesic flow keeps the error bounded and stable throughout the full duration of the simulations with a mean error of only 0.076 meters. Consequently, the Geodesic Flow reduces the tracking error by 75%; more importantly, it prevents the catastrophic localization failure observed in the baseline, allowing for successful navigation.

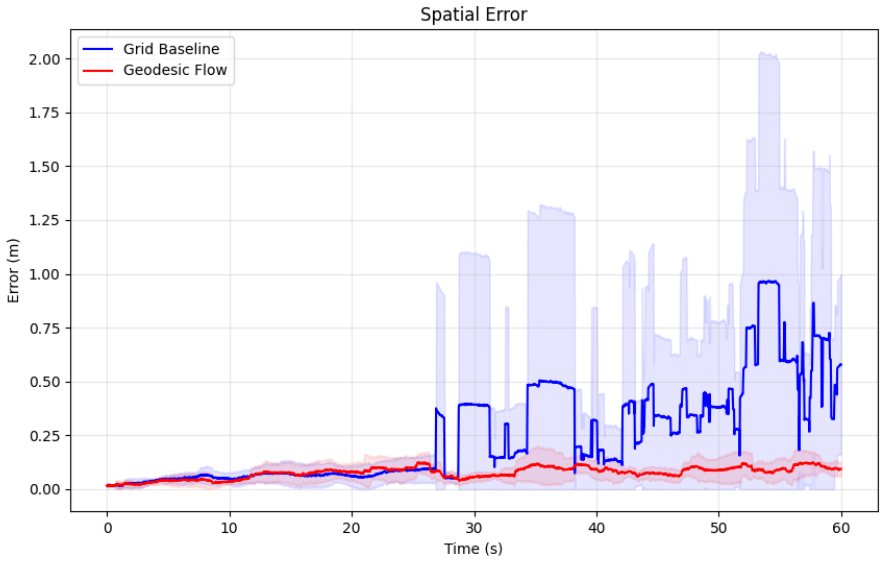


Figure 5.2: Average aggregate error for 5 trajectories given 1500 neuron population.

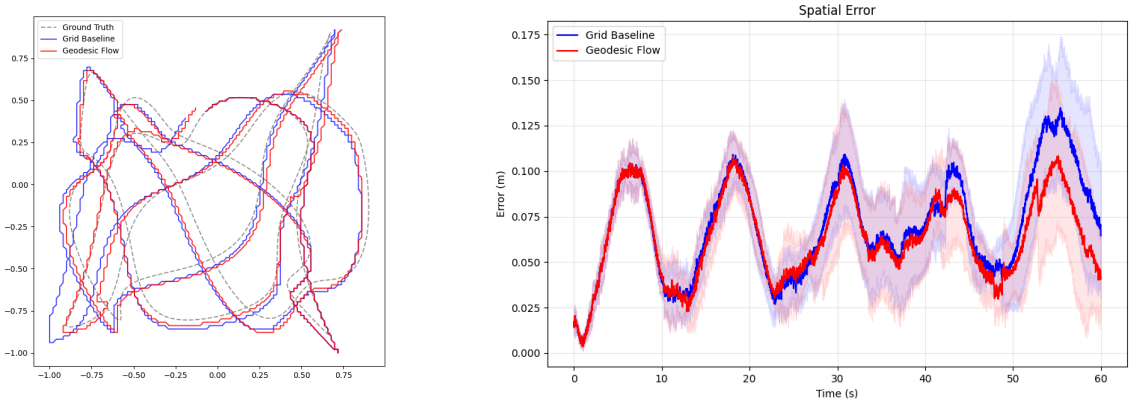
It is notable that the Grid baseline fails here despite using a high-resolution (64×64) lookup, which performed near-perfectly in static benchmarks (Chapter 4). This discrepancy highlights the fundamental difference between static denoising and dynamic error correction.

The grid method performs a "hard snap" to the nearest discrete prototype. In a recurrent system, this quantization effectively acts as a noise amplifier: if instantaneous noise pushes the state slightly across a decision boundary, the grid snaps it to an incorrect neighbor (similarly to phase drift in Sec 2.4). This process converts small, continuous perturbations into larger, discrete position errors. Rather than being averaged out, these amplified errors are subsequently integrated into the path estimate, causing the trajectory to diverge rapidly from the ground truth. The Geodesic flow, by contrast, provides a continuous manifold-aware correction field, effectively "pulling" the drifting state back to the manifold without the instability introduced by hard quantization boundaries.

It is notable that the Grid baseline fails here despite using a high-resolution (4) lookup, which performed near-perfectly in static benchmarks (Chapter 4). This discrepancy may highlight the difference between static denoising and a learned dynamic attractor. The grid method performs a "hard snap" to the nearest discrete prototype. In a dynamical system accumulating drift, if the state is pushed slightly closer to an incorrect neighbor, the

grid snaps it to the wrong attractor, introducing a minor discontinuous jumps that disrupt the velocity integration. The Geodesic flow, by contrast, provides a continuous manifold-aware correction field, effectively "pulling" the drifting state back to the manifold without introducing discrete discontinuities.

At $N = 3000$ (Figure 5.3), the recurrent network has sufficient capacity to maintain stability on its own, and the corrections made by the cleanup are applied to mildly corrupted representations. Here, the Geodesic flow and the grid cleanup are tied in performance (0.064 ± 0.011 and 0.07 ± 0.01, respectively). This indicates that Geodesic flow cleanup is most critical in a noisy model, effectively allowing a smaller network to emulate the performance of a larger one.



(a) (b)

Figure 5.3: Localization performance at $N = 3000$ neurons. (a) Sample trajectory with PI results given 3000 neuron population. (b) Average aggregate error for 5 trajectories given 3000 neuron population. At high capacities, intrinsic stability is sufficient, but Flow cleanup still reduces variance.

5.4.2 Map Accuracy and Efficiency Analysis

The SLAM system is evaluated on the two distinct but coupled objectives: Localization (estimating the agent's trajectory) and Mapping (estimating the positions of environmental features). Table 5.1 summarizes these errors across $N = 5$ trials.

- Path Mean Error (Localization): computed as the Mean Absolute Trajectory Error.

This is the Euclidean distance between the estimated agent position p_t and ground truth p_t^*, averaged over all time steps T in the trajectory: $\frac{1}{T}\sum_{t=1}^{T}\|p_t - p_t^*\|$.

- Landmark RMSE (Mapping): computed as the Root Mean Square Error of the estimated positions for all N observed landmarks. This quantifies the metric accuracy of the constructed map: $\sqrt{\frac{1}{N}\sum_{i=1}^{N}\|l_i - l_i^*\|^2}$.

Table 5.1: Comparison of Landmark and Trajectory errors across Path Integrator sizes. **Landmark RMSE** measures the geometric accuracy of the learned map, while **Path Error** measures the fidelity of the agent's localization. Bold indicates best performance.

PI Neurons	Method	Landmark RMSE (m)	Path Mean Error (m)
1000	Grid	0.834 ± 0.155	0.586 ± 0.121
	Geodesic Flow	$\mathbf{0.393 \pm 0.036}$	$\mathbf{0.162 \pm 0.055}$
1500	Grid	0.512 ± 0.294	0.249 ± 0.239
	Geodesic Flow	$\mathbf{0.308 \pm 0.098}$	$\mathbf{0.076 \pm 0.026}$
2000	Grid	0.281 ± 0.120	0.061 ± 0.011
	Geodesic Flow	0.286 ± 0.080	0.067 ± 0.014
2500	Grid	0.295 ± 0.127	0.083 ± 0.017
	Geodesic Flow	0.283 ± 0.088	0.078 ± 0.009
3000	Grid	0.271 ± 0.129	0.070 ± 0.010
	Geodesic Flow	0.251 ± 0.099	0.064 ± 0.011

The results confirm the trend observed in the trajectory plots. At $N = 1000$, Geodesic flow reduces path RMSE by $\approx 72\%$ and Landmark RMSE by $\approx 53\%$ compared to the grid baseline. Crucially, the flow-augmented system at $N = 1500$ achieves a Path RMSE of 0.076m, which is comparable to the Grid-baseline performance at $N = 2500$ (0.083m). This implies that the Geodesic cleanup can reduce the required neural population size by nearly 40% while maintaining equivalent tracking accuracy.

The performance gap at lower neuron counts is likely due to the discrete nature of the grid lookup. Snapping a drifting state to the nearest valid grid point introduces small, discontinuous jumps in the state space. If the state drifts slightly closer to an incorrect attractor, the grid method reinforces the error, leading to the divergence in Figure 5.1. In contrast, the Geodesic Flow model provides a continuous correction field. By projecting the state back to the manifold via a smooth Geodesic trajectory, it corrects phase drift

without introducing state discontinuities. This allows the cleanup mechanism to integrate seamlessly with the dynamics governing the neural oscillators.

Most significantly, these results highlight the efficiency of the proposed approach. In neuromorphic hardware contexts where neurons and synapses are the primary cost constraints, the observed 40% reduction in resource requirements validates the inclusion of a dedicated, manifold-aware cleanup circuit.

In summary, this chapter confirms that formulating cleanup as a generative transport problem provides not just higher reconstruction fidelity, but prevents catastrophic error accumulation in recursive integration. This validates the thesis hypothesis that respecting the intrinsic geometry of the embedding space is essential for maintaining trajectory consistency in robust neuro-symbolic computation..

Chapter 6

Conclusion and Future Work

We have presented a novel framework for building associative memories for Spatial Semantic Pointers, shifting the paradigm of cleanup from a search in the embedded domain to continuous generative transport directly in the embedding space. It was demonstrated that the intrinsic geometry of the representation cannot be ignored. While Euclidean interpolants failed to preserve phase information, Geodesic flows, which restricted transport to the Riemannian manifold of the hypersphere, provided robust restoration even in high-noise regimes.

The utility of this approach was validated in a closed-loop Spiking Semantic SLAM system. The proposed model successfully stabilized path-integration estimates against phase drift, preventing catastrophic localization failure. Crucially, this stabilization allowed a resource-constrained neural population to match the tracking performance of a baseline system with nearly twice its allocated neural resources, demonstrating that manifold-aware cleanup is a key enabler for efficient neuromorphic computing.

While this work establishes the foundations of flow-based cleanup, several avenues remain for future investigation.

The current implementation relied on standard Artificial Neural Networks (ANNs). To fully align with the neuromorphic context of SSPs, future work may convert the learned velocity fields into Spiking Neural Networks (SNNs) using frameworks such as SNNTorch, enabling deployment on neuromorphic hardware [16]. Furthermore, investigating biologically plausible alternatives to backpropagation for learning vector fields would further close the gap between the mathematical framework and neural realism.

A distinct challenge arises when applying cleanup to "bundled" representations of SSPs. In the current formulation, a bundle is treated as a noisy deviation from a single valid SSP,

causing the flow to collapse the superposition into a single (highest likelihood) peak. Since a single SSP vector inherently encodes a quasi-probability distribution over space, a bundle effectively represents a multi-modal distribution. In the current formulation, the flow tends to collapse this superposition into a single peak (the maximum likelyhood) performing a "winner-take-all" cleanup.

Alternatively, one might desire a cleanup process that samples single SSPs in proportion to their saliency within the bundle. Achieving this requires precise control over the data likelihood. As noted in recent literature, deterministic velocity matching lacks theoretical guarantees for controlling the divergence between model and target distributions in such complex regimes [1]. This is explored in the **Stochastic Interpolants**, which incorporate a diffusive term and a supplementary score model to the flow to achieve strict control over the output density, allowing for probabilistic recovery of constituent items.

Finally, in neural representations such as SLAM, the associative memory network could be replaced with a bundle connected to a denoising network that removes noise while preserving the multi-modal structure of the maintained cognitive map (i.e., keeping "everything else").

Beyond cleanup, the flow matching framework offers the potential to synthesize novel SSP representations, moving from denoising to generation. The model could be extended to generate novel SSP sequences conditioned on high-level constraints, such as generating a trajectory given a start and end goal. Additionally, techniques such as CurlyFM could be adapted to learn arbitrary dynamics directly on the hypersphere surface [38]. This would allow the system to simulate valid motion trajectories across SSPs entirely in the embedding space, enabling applications in imaginary rollout and path planning without decoding to Euclidean space.

Finally, the computational efficiency of the employed network was not the primary focus of this work. Future research should explore lightweight architectures suited to leverage equivariances in the SSP structure. This would allow neural cleanup models to better compete with grid-lookup baselines on raw computational throughput while maintaining the superior flexibility and continuous mapping capabilities demonstrated in this thesis.

References

[1] Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

[2] Luca Ambrogioni. In search of dispersed memories: Generative diffusion models are associative memory networks. *ArXiv*, abs/2309.17290, 2023.

[3] Applied Brain Research. Nengo dl, 2024. Accessed: 2024-04-16.

[4] Michal Balcerak, Tamaz Amiranashvili, Antonio Terpin, Suprosanna Shit, Lea Bogensperger, Sebastian Kaltenbach, Petros Koumoutsakos, and Bjoern Menze. Energy matching: Unifying flow matching and energy-based models for generative modeling. *arXiv preprint arXiv:2504.10612*, 2025.

[5] A. Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, and Dominik Lorenz. Stable video diffusion: Scaling latent video diffusion models to large datasets. *ArXiv*, abs/2311.15127, 2023.

[6] Alicia Bremer and Jeff Orchard. Improved cleanup and decoding of fractional power encodings. In *2025 Neuro Inspired Computational Elements (NICE)*, pages 1–9. IEEE, 2025.

[7] Vassilis Cutsuridis and Thomas Wennekers. Hippocampus, microcircuits and associative memory. *Neural networks : the official journal of the International Neural Network Society*, 22 8:1120–8, 2009.

[8] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

[9] Imant Daunhawer, Thomas M. Sutter, Kieran Chin-Cheong, Emanuele Palumbo, and Julia E. Vogt. On the limitations of multimodal vaes. *ArXiv*, abs/2110.04121, 2021.

[10] Nicole Sandra-Yaffa Dumont and Chris Eliasmith. Accurate representation for spatial cognition using grid cells. In *Annual Meeting of the Cognitive Science Society*, 2020.

[11] Nicole Sandra-Yaffa Dumont, P. Michael Furlong, Jeff Orchard, and Chris Eliasmith. Exploiting semantic information in a spiking neural slam system. *Frontiers in Neuroscience*, 17, 2023.

[12] Nicole Sandra-Yaffa Dumont, Andreas Stöckel, P. Michael Furlong, Madeleine E. Bartlett, Chris Eliasmith, and Terrence C. Stewart. Biologically-based computation: How neural details and dynamics are suited for implementing a variety of algorithms. *Brain Sciences*, 13, 2023.

[13] Chris Eliasmith. Neural engineering: Computation, representation, and dynamics in neurobiological systems. *Trans. Neur. Netw.*, 15(2):528–529, March 2004.

[14] Chris Eliasmith and Charles H. Anderson. *Neural Engineering: Computation, Representation, and Dynamics in Neurobiological Systems*. MIT Press, Cambridge, MA, 2003.

[15] Chris Eliasmith, Terrence C Stewart, Xuan Choo, Trevor Bekolay, Travis DeWolf, Yichuan Tang, and Daniel Rasmussen. A large-scale model of the functioning brain. *science*, 338(6111):1202–1205, 2012.

[16] Jason K Eshraghian, Max Ward, Emre Neftci, Xinxin Wang, Gregor Lenz, Girish Dwivedi, Mohammed Bennamoun, Doo Seok Jeong, and Wei D Lu. Training spiking neural networks using lessons from deep learning. *arXiv preprint arXiv:1906.09395*, 2021.

[17] Kilian Fatras, Younes Zine, Szymon Majewski, Rémi Flamary, Rémi Gribonval, and Nicolas Courty. Minibatch optimal transport distances; analysis and applications. *arXiv preprint arXiv:2101.01792*, 2021.

[18] Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, Léo Gautheron, Nathalie T.H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021.

[19] P. Michael Furlong. Fractional binding in vector symbolic architectures as quasi-probability statements. 2022.

[20] P. Michael Furlong and Chris Eliasmith. Modelling neural probabilistic computation using vector symbolic architectures. *Cognitive Neurodynamics*, pages 1–24, 2023.

[21] Richard J. Gardner, Erik Hermansen, Marius Pachitariu, Yoram Burak, Nils A. Baas, Benjamin A. Dunn, May-Britt Moser, and Edvard I. Moser. Toroidal topology of population activity in grid cells. *Nature*, 602:123 – 128, 2021.

[22] Jonathan Ho. Classifier-free diffusion guidance. *ArXiv*, abs/2207.12598, 2022.

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

[24] Benjamin Hoover, Hendrik Strobelt, Dmitry Krotov, Judy Hoffman, Zsolt Kira, and Duen Horng Chau. Memory in plain sight: A survey of the uncanny resemblances between diffusion models and associative memories. *arXiv preprint arXiv:2309.16750*, 2023.

[25] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982.

[26] John J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, 79 8:2554–8, 1982.

[27] Pentti Kanerva. Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. *Cognitive Computation*, 1(2):139–159, 2009.

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

[29] Spencer J Kent, E Paxon Frady, Friedrich T Sommer, and Bruno A Olshausen. Resonator networks, 2: Factorization performance and capacity compared to optimization-based methods. *Neural computation*, 32(12):2332–2388, 2020.

[30] Brent Komer, Terrence C Stewart, Aaron R Voelker, and Chris Eliasmith. A neural representation of continuous space using fractional binding. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 41, 2019.

[31] Dmitry Krotov and John J. Hopfield. Dense associative memory for pattern recognition. *ArXiv*, abs/1606.01164, 2016.

[32] Y. Li, Xin xin Lu, Yaqing Wang, and De-Yu Dou. Generative time series forecasting with diffusion, denoise, and disentanglement. *ArXiv*, abs/2301.03028, 2023.

[33] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *ArXiv*, abs/2210.02747, 2022.

[34] Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

[35] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *ArXiv*, abs/2209.03003, 2022.

[36] Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, and Q. Liu. Instaflow: One step is enough for high-quality diffusion-based text-to-image generation. *ArXiv*, abs/2309.06380, 2023.

[37] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. *ArXiv*, abs/2102.09672, 2021.

[38] Katarina Petrović, Lazar Atanackovic, Viggo Moro, Kacper Kapuśniak, İsmail İlkan Ceylan, Michael Bronstein, Avishek Joey Bose, and Alexander Tong. Curly flow matching for learning non-gradient field dynamics. *arXiv preprint arXiv:2510.26645*, 2025.

[39] Tony A Plate. Holographic reduced representations. *IEEE Transactions on Neural networks*, 6(3):623–641, 1995.

[40] Tony A. Plate and Geoffrey E. Hinton. Distributed representations and nested compositional structure. 1994.

[41] Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, et al. Hopfield networks is all you need. *arXiv preprint arXiv:2008.02217*, 2020.

[42] Bernhard Schölkopf, Koji Tsuda, and Jean-Philippe Vert. *Kernel methods in computational biology*. MIT press, 2004.

[43] Julius O Smith. *Mathematics of the Discrete Fourier Transform (DFT), with Audio Applications*. W3K Publishing, second edition, 2007. https://ccrma.stanford.edu/~jos/st/Conjugation_Reversal.html.

[44] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *ArXiv*, abs/2010.02502, 2020.

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

[46] Terrence C. Stewart, Yichuan Tang, and Chris Eliasmith. A biologically realistic cleanup memory: Autoassociation in spiking neurons. *Cogn. Syst. Res.*, 12(2):84–92, 2011.

[47] Qiao Sun, Zhicheng Jiang, Hanhong Zhao, and Kaiming He. Is noise conditioning necessary for denoising generative models? *arXiv preprint arXiv:2502.13129*, 2025.

[48] Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *arXiv preprint arXiv:2302.00482*, 2023.

[49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[50] Runqian Wang and Yilun Du. Equilibrium matching: Generative modeling with implicit energy-based models. *arXiv preprint arXiv:2510.02300*, 2025.

[51] Jiawen Wu, Bingguang Chen, Yuyi Zhou, Qi Meng, Rongchan Zhu, and Zhi-Ming Ma. Riemannian neural geodesic interpolant. *arXiv preprint arXiv:2504.15736*, 2025.

[52] Ling Yang, Zhilong Zhang, Shenda Hong, Runsheng Xu, Yue Zhao, Yingxia Shao, Wentao Zhang, Ming-Hsuan Yang, and Bin Cui. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 56:1 – 39, 2022.

[53] Chenshuang Zhang, Chaoning Zhang, Sheng Zheng, Mengchun Zhang, Maryam Qamar, Sung-Ho Bae, and In-So Kweon. A survey on audio diffusion models: Text to speech synthesis and enhancement in generative ai. *ArXiv*, abs/2303.13336, 2023.

[54] Linqi Zhou, Aaron Lou, Samar Khanna, and Stefano Ermon. Denoising diffusion bridge models. *ArXiv*, abs/2309.16948, 2023.

[55] Eric A Zilli, Motoharu Yoshida, Babak Tahvildari, Lisa M Giocomo, and Michael E Hasselmo. Evaluation of the oscillatory interference model of grid cell firing through analysis and measured period variance of some biological oscillators. *PLoS computational biology*, 5(11):e1000573, 2009.