Biologically-Plausible Markov Chain Monte Carlo Sampling from Vector Symbolic Algebra-Encoded Distributions

P. Michael Furlong[1,2](✉), Kathryn Simone[1,3], Nicole Sandra-Yaffa Dumont[1,3], Madeleine Bartlett[1,3], Terrence C. Stewart[5], Jeff Orchard[1,3], and Chris Eliasmith[1,2,4]

[1] Centre for Theoretical Neuroscience, University of Waterloo, Waterloo, Canada
[2] Systems Design Engineering, University of Waterloo, Waterloo, Canada
[3] Cheriton School of Computer Science, University of Waterloo, Waterloo, Canada
[4] Department of Philosophy, University of Waterloo, Waterloo, Canada
[5] National Research Council of Canada, Waterloo, Canada
{michael.furlong,kathryn.simone,ns2dumont, madeleine.bartlett,jorchard,celiasmith}@uwaterloo.ca
terrence.stewart@nrc-cnrc.gc.ca

Abstract. Vector symbolic algebras (VSAs) are modelling frameworks that unify human cognition and neural network models, and some have recently been shown to be probabilistic models akin to Kernel Mean Embeddings. Sampling from vector-embedded distributions is an important tool for turning distributions into decisions, in the context of cognitive modelling, or actions, in the context of reinforcement learning. However, current techniques for sampling from these distribution embeddings rely on knowledge of the kernel embedding or its gradient, knowledge which is problematic for neural systems to access. In this paper, we explore biologically-plausible Hamiltonian Monte Carlo Markov Chain sampling in the space of VSA encodings, without relying on any explicit knowledge of the encoding scheme. Specifically, we encode data using a Holographic Reduced Representation (HRR) VSA, sample from the encoded distributions using Langevin dynamics in the VSA vector space, and demonstrate competitive sampling performance in a spiking-neural network implementation. Surprisingly, while the Langevin dynamics are not constrained to the manifold defined by the HRR encoding, the generated samples contain sufficient information to reconstruct the target distribution, given an appropriate decoding scheme. We also demonstrate that the HRR algebra provides a straightforward conditioning operation. These results show that a generalized sampling model can explain how brains turn probabilistic latent representations into concrete actions in an encoding scheme-agnostic fashion. Moreover, sampling from vector embeddings of distributions permits the implementation of probabilistic algorithms, capturing uncertainty in cognitive models. We also note that the ease of conditioning distributions is particularly well-suited to reinforcement learning applications.

Keywords: Hamiltonian Monte Carlo · Neural Sampling · Vector Symbolic algebras

M. Wand et al. (Eds.): ICANN 2024, LNCS 15019, pp. 94–108, 2024.
https://doi.org/10.1007/978-3-031-72341-4_7

1 Introduction

Managing uncertainty is essential for organisms that operate in noisy and ambiguous environments. The capacity to encode, manipulate, and sample probabilistic information about the world's state is vital for decision-making and perception. Although formal mathematical descriptions of probability and uncertainty have been successful in machine learning and control theory, and are recognized as an organizing theory for cognition [8,15], the neurological basis of analogous computations remains poorly understood.

Prior approaches to capturing probability in neural systems have proposed methods that map random variables onto individual neurons [*e.g.*, 4,7,27,31,32]. Alternative proposals suggest that populations of neurons represent distributions over random variables [1–3,6,10,19,23,24,41,42], and allow biologically plausible sampling via linear decoding of values from neural activity [*e.g.*, 17,18,34]. However, there remains a gap between the expression of probabilistic cognitive models at the algorithmic level, and their neural implementation.

One way to bridge this gap is to use VSA; a family of algebras over high-dimensional vector spaces that provide a hypothesis about the structure of the latent representations that neural networks manipulate to perform cognition [9,13,20,22,29,37]. One such VSA is the Holographic Reduced Representation (HRR) algebra, developed by Plate [30], and more recent restricted forms for the representation of continuous data called Spatial Semantic Pointers (SSPs) [21]. SSPs are inherently probabilistic [12], providing one explanation for how organisms may represent and reason about uncertainty. However, the ability to sample from these probabilistic representations, crucial for models of decision-making, remains unexplored. It is important to note that SSP encodings are non-linear data embeddings, which may require more complex samplers than do linear embeddings, as suggested by Savin and Denève [34]. If true, this throws into doubt the feasibility of using simple recurrent dynamics to sample distributions represented by SSPs.

The method of embedding distributions using VSA representations that we rely on here, bears a strong resemblance to the techniques developed in the literature of Kernel Mean Embeddings (KMEs), recently surveyed by Muandet et al. [26]. We posit that the HRR algebra has advantages over the KME formulation for implementation in resource constrained neural networks. Specifically, the algebra preserves dimensionality, bounding resource requirements for representing arbitrary compositions of data, and provides a computationally simple method for conditioning distributions (see Sect. 3.4).

Sampling from vector-embeddings of distributions has been explored in the context of the KME literature, but these methods rely on internal knowledge of the feature-space encoding in the form of gradients [35,36,39]. Accessing this knowledge is problematic from a neural perspective for two reasons. First, while the samplers take advantage of the efficiency of KME representations of distributions, the sampling is still occurring in the underlying domain, a representation that may never be instantiated within the brain. Second, sharing knowledge about the internals of the data encoding function with the (hypothesized) circuit that performs sampling is a transport problem at least on par with the weight transport problem found in backpropagation.

Consequently, turning VSA-encoded distributions into specific decisions or actions requires a method for sampling from those distributions directly. In this paper, we consider an approach to Markov Chain Monte Carlo (MCMC) sampling from the VSA-encoded distributions using Langevin dynamics. Langevin dynamics are of interest because they can be implemented by the dynamics of recurrent neural networks [17,18,34], and because Monte Carlo sampling has been proposed as an explanation for how brains can be probabilistic, but still diverge from optimal decision-making [5,33].

The contribution of this paper is the development of a biologically plausible sampler for distributions encoded using the HRR algebra. We argue that this approach is biologically plausible on several counts. First, the VSAs we adopt as our feature space embedding were developed in the context of cognitive modelling [20,37] and are used to implement models that can be readily translated into neural networks capable of reproducing physiological data such as BOLD signals (see Eliasmith [[9], Sect. 4]). Second, the binding/unbinding operation of the chosen VSA (Plate's HRR [30], outlined in Sect. 2.1.1) can be implemented either via matrix-vector multiplication (a process readily implemented in neural networks), or as a spiking neural circuit [38]. This latter approach permits unbinding on-the-fly, allowing for flexible modification of vector-embedded distributions (*i.e.*, conditioning). Third, by not relying on knowledge of the gradient of the encoding scheme, we avoid the transport problem that would arise when implementing KME Hamiltonian-MC methods [35,36,39] in neurons. Finally, as mentioned above, the use of Langevin dynamics adds biological plausibility since they are readily implemented by neural networks, as we demonstrate in Sect. 3.5. Importantly, the advantages of biological plausibility aren't limited to cognitive modelling applications, but extend also to the implementation of neural networks on neuromorphic hardware, where they can offer benefits in terms of energy efficiency as well as for the study of embodied cognition.

In the rest of this paper, we demonstrate the utility and generality of this sampling method, by sampling from continuous, discrete, and mixed continuous and discrete distributions. We show that we are able to recover samples that approximate the target distributions of the sampling algorithms, and that we are able to sample from conditioned distributions without re-training. Finally, we present a recurrent Spiking Neural Network (SNN) implementation of the proposed algorithm. We conclude with a discussion of the implications of the combination of this particular representation, and the neurologically-plausible sampling dynamics, for cognition and reinforcement learning.

2 Methods

VSAs specify vector representations together with a set of semantically-meaningful manipulations that can specify algorithms, that can, in turn, be implemented in a neural network. Working within a VSA lets one separate the function and implementation of algorithms. In the following section, we first review the HRR algebra and its probabilistic interpretation, then we focus on the proposed algorithm. In Sect. 2.3 we turn our attention to describing a spiking neural network implementation, accomplished using the Neural Engineering Framework [NEF; 10], which is a framework for translating functions and algorithms into the activities of populations of spiking neurons.

2.1 Holographic Reduced Representations (HRRs)

2.1.1 Vector Representation and Algebra: The specific VSA we employ is HRR [30]. We choose HRR as it is dimensionality-preserving, and can represent discrete- and continuous-valued data. HRR has four operations that we use in this paper: similarity (vector dot product), bundling (vector addition), binding (circular convolution following the convention of Plate [30], and denoted as $\otimes$), and unbinding (bind with vector's pseudo-inverse, denoted as $\oslash$).

To represent continuous data, $\mathrm{X} \subseteq \mathbb{R}^m$, we project input samples $\mathbf{x} \in \mathrm{X}$ into a vector space of dimensionality d via mapping ϕ:

$$\phi_{\mathrm{X}}(\boldsymbol{\lambda}^{-1}\mathbf{x}) = \mathcal{F}^{-1}\left\{\mathrm{e}^{\mathrm{i}A\boldsymbol{\lambda}^{-1}\mathbf{x}}\right\} \tag{1}$$

where λ is a diagonal, non-negative matrix whose entries $(\lambda_1, ...\lambda_m)$ are user-specified parameters defining the length scale of the representation, A is a $d \times m$ matrix, and $\mathcal{F}^{-1}$ denotes the inverse Fourier transform. We refer to A as the *phase matrix*. It is constructed of a set of column vectors $\mathbf{a}_j$, each comprising a collection of frequencies. Elements $a_{i,j}, i = \{1, ..., (d-1)/2\}$ are drawn from a uniform distribution over the range $[-\pi, \pi]$ with $0, j = 1$. The columns of A are then forced to have conjugate symmetry, to ensure that $\phi_{\mathrm{X}}(x)$ is real-valued. These choices ensure that the Fourier components of $\phi_{\mathrm{X}}(x)$ all have magnitude one, a property that qualifies them as *unitary vectors*, which is essential for stable, iterative composition.

This method of representing continuous-valued data is a generalization of HRR's binding operator, called *fractional* binding [21,28] or fractional power encoding [11]. Following Komer et al. [21], we refer to the hypervectors resulting from this projection as SSPs. The encoding of multidimensional data can also be written as binding the results of each such mapping together (*e.g.*, $\phi_{\mathrm{X}}(\boldsymbol{\lambda}^{-1}\mathbf{x}) = \phi_{\mathrm{X}_1}(x_1/\lambda_1) \circledast \cdots \circledast \phi_{\mathrm{X}_m}(x_m/\lambda_m)$, selecting a different random unitary vector a_m for each feature m of the data.

To represent discrete data, we map each value to a random unitary vector. Specifically, the projections of values into the high-dimensional VSA space, $\phi(n) : \mathbb{Z}^+ \to \mathbb{R}^d$ are generated such that $\phi(i) \cdot \phi(j) \approx 0$ for $i \neq j$[1]. This method is used in the VSA literature to approximate categorical, as opposed to ordinal, data. In this paper, we refer to these representations as Semantic Pointers (SPs), and use "HRRs" to refer to any vector representation in the space (including SSPs, SPs, and combinations thereof).

[1] We give the mapping here for positive integers, but all discrete data can be mapped onto a subset of the integers.

In contrast to the encoding of discrete objects, similarity between SSPs is distance preserving. In fact, the similarity between two SSPs approximates a normalized sinc function: $\phi(x/\lambda) \cdot \phi(x'/\lambda) \approx \text{sinc}(\pi|\frac{x-x'}{\lambda}|)$. Next we show how to use the HRR to represent probability distributions.

2.1.2 Probabilistic Computations: In order to sample from a distribution, we first approximate that distribution. Furlong and Eliasmith [12] provide an in-depth treatment of probabilistic modelling using SSPs, which we extend in this work to represent and sample from distributions over continuous, discrete, and mixed continuous-discrete data, and demonstrate sampling from conditioned distributions. Briefly, we encode K IID samples from some underlying distribution (henceforth the target distribution) using the HRR, and compress a $K \times m$ database of samples into a d-dimensional vector representation of the probability distribution, using HRR's bundling operator, akin to a KME:

$$\boldsymbol{\mu} = \frac{1}{K\prod_{i=1}^{m}\lambda_i}\sum_{k=1}^{K}\phi_{\text{X}}\left(\text{diag}(\boldsymbol{\lambda})^{-1}\mathbf{x}_k\right). \tag{2}$$

We then use $\boldsymbol{\mu}$ to evaluate the density function, using the similarity operator between HRRs to induce an approximate sinc kernel between data points:

$$P(X = \mathbf{x}) \approx \max\left\{0, \phi_{\text{X}}(\boldsymbol{\lambda}^{-1}\mathbf{x}) \cdot \boldsymbol{\mu} - \xi\right\} \tag{3}$$

where ξ is a bias term, and the transformation $\max\{0, \cdot - \xi\}$ converts quasi-probability for sinc-based estimators to probability [14]. Approximating a distribution requires selecting parameters d, $\boldsymbol{\lambda}$, and ξ. We employ $d = 1024$ and select scalar $\boldsymbol{\lambda}$ and ξ on a per-distribution basis through hyperparameter tuning.

Probability distributions can be conditioned through HRR's unbinding operator. Unbinding has the effect of pre-evaluating one of the arguments of a probability distribution. Namely,

$$\begin{aligned}\mu_{\text{XY}} \circledast \phi_{\text{X}}^{-1}(x) &= \frac{1}{n}\sum_{i=1}^{n}\left(\phi_{\text{X}}(x_i) \circledast \phi_{\text{Y}}(y_i)\right) \circledast \phi_{\text{X}}^{-1}(x) \\ &= \frac{1}{n}\sum_{i=1}^{n}\left(\phi_{\text{X}}(x_i - x) \circledast \phi_{\text{Y}}(y_i)\right).\end{aligned}$$

Consequently, when we evaluate a probability, $\phi_{\text{X}}(x) \cdot \left(\mu_{\text{XY}} \circledast \phi_{\text{Y}}^{-1}(y)\right)$, we are effectively approximating $P(X \mid Y = y)P(Y = y)$, an unnormalized conditional distribution. Since samplers using the Metropolis acceptance criterion can sample unnormalized distributions, Hamiltonian-MC sampling is compatible with our model of conditioning.

2.2 Sampling Algorithm

The transition function of our Metropolis-Hastings sampler (HAM-MCMC-HRR) (Algorithm 1) is defined by the Langevin dynamics

$$\dot{\phi} = \gamma\nabla_{\phi}\log P(x) + \nu, \tag{4}$$

Algorithm 1. Metropolis-Hastings sampling in VSA space with Langevin transition dynamics.

```
1: function METROPOLIS-HASTINGS($N_{\text{samples}}, G, P$)
2:     $t \leftarrow 0$
3:     $\phi_t \leftarrow \phi(\mathbf{x}_0)$
4:     while $t \leq N_{\text{samples}}$ do
5:         $\nu \sim \mathcal{N}\left(\mathbf{0}, \frac{\sigma^2}{d} I\right)$
6:         $\phi' \leftarrow \phi_t + \gamma \nabla_\phi \log P(X) + \nu$
7:         if do_cleanup then
8:             $\phi_t \leftarrow \text{cleanup}(\phi_t)$
9:         end if
10:        $u \sim \mathcal{U}[0, 1]$
11:        if $u \leq \frac{P(\phi')}{P(\phi_t)}$ then
12:            $\phi_t \leftarrow \phi'$
13:        end if
14:        yield $\phi_t$
15:    end while
16: end function
```

where γ is a step size, and $\nu \sim \mathcal{N}\left(\mathbf{0}, \frac{\sigma^2}{d} I\right)$ in the d-dimensional VSA space. A variance scale factor of $\frac{1}{d}$ reflects that the variance of the dot product between randomly generated SSPs has a variance of $\frac{1}{d}$ [16,40]. The gradient of the SSP encoding of $\log P(\mathbf{x})$ is defined:

$$\begin{aligned}\nabla_\phi \log P(\mathbf{x}) &= \nabla_\phi \log \max\{0, \phi_{\text{X}}(\mathbf{x}) \cdot \mu_{\text{X}} - \xi\} \\ &= \begin{cases} \frac{1}{P(\mathbf{x})} \mu_{\text{x}} & \text{if } P(\mathbf{x}) > 0 \\ 0 & \text{otherwise.} \end{cases}\end{aligned} \tag{5}$$

Because our representations are high-dimensional, pseudo-orthogonal vectors, we are concerned that Langevin dynamics alone are not sufficient to stay close to the manifold defined by an HRR encoding. Consequently, we optionally include a cleanup memory in line with the dynamics to map samples to valid HRRss.

To decode samples from the VSA space back into the encoded domain, we find the closest point in the HRR domain in a look-up table, and then use an optimizer to find the closest point in the encoded domain to the generated sample. This decoding process outputs a value within the bounds defined for the domain. Even though samples are not decoded until the sampling process is complete, it is important to determine how big a role decoding plays in shaping the generated samples. To evaluate whether the decoder alone is sufficient to reproduce the target distribution, we test the Langevin dynamics with and without the gradient term, *i.e.*, $\gamma > 0$ or $\gamma = 0$ in Eq. (4) (see Table 1).

In each experiment, we encode $K = 10,000$ observations as SSPs and then fit the target distribution with $\boldsymbol{\mu}$, as in Eq. (2). To test the algorithm we designed a 2×2 factorial experiment with cleanup memory (with vs. without) and gradient (with vs. without) as factors. To further demonstrate the utility of the sampler, we also demonstrate sampling using conditioning in SSP space and from a mixed discrete-continuous distribution (see Sect. 3.4).

2.3 Spiking Neural Network Algorithm

To implement the sampling algorithm presented in Sect. 2.2 using a recurrent SNN, we use the methodology of the NEF. A population of $2 \times d$ Leaky Integrateand-Fire (LIF) neurons are used to represent the state of the sampler, $\phi(t) \in \mathbb{R}^d$. This population is recurrently connected with weights set to approximate

$$f(\phi(t)) = \tau\gamma\nabla_{\phi(t)} \log P(x) + \phi(t), \tag{6}$$

where $P(x)$ is the softplus function (a smooth approximation of the ReLU function used in Eq. (3)), and τ is the time constant of the recurrent connections' synaptic filter. The synaptic filter (a first-order lowpass filter) is applied to the spike train output of the population, before it is multiplied by connection weights.

The population also receives a filtered white noise signal $\tau\nu$ as input. The presence of τ in the input and dynamics is to account for the effect filtering has on the system's dynamics (see Eliasmith and Anderson [10] for more details). The state of the sampler is decoded from the neural activities at every time-step. The SSP samples are, in turn, decoded from this output.

There are several key differences between this implementation and Algorithm 1. While the SNN SSP sampler approximates the Langevin dynamics given in (4), it does not include the Metropolis acceptance criterion (lines 11–12 in Algorithm 1). Additionally, the SNN SSP sampler does not have an option for integrating cleanup in the recurrent loop (lines 7–8 in Algorithm 1). As a result, the sampler may stray from the manifold defined by the HRR encoding. These features were excluded to simplify this proof-of-concept implementation.

2.4 Baseline Algorithms

We compared our algorithm against two Metropolis-Hastings sampling baselines. In the KDE-MCMC baseline, the target distribution is estimated using a Kernel Density Estmimator (KDE) with a Radial Basis Function kernel. Candidate samples are generated using an isotropic Gaussian distribution centred at the previous sample point, *i.e.*, $\mathbf{x}_{t+1} \sim \mathcal{N}(\mathbf{x}_t, \mathrm{diag}(\boldsymbol{\sigma}))$. In the HAM-MCMC-DOMAIN baseline, sampling occurs in the feature space of the data, but the transition function is defined with respect to the HRR embedding, as: $\dot{x} = \gamma\nabla_\phi \left[\log P(x)\right] \left[\nabla_x \phi(x)\right]^T + \nu$, where $\nabla_x \phi(x)$ is the gradient of the HRR encoding at x, and $\nu \sim \mathcal{N}\left(\mathbf{0}, \sigma^2 I\right)$, implementing the method of [39] using our kernel embedding. This comparison allows us to assess the contribution of information about the encoding scheme to the performance of the sampler. As a baseline for discrete distributions, we use an MCMC sampler, using a binomial distribution to generate candidate samples. For both continuous and discrete distributions, we upper bound acceptable sampling error by comparing to inverse transform sampling (ITS) with naïve fits to a uniform and a unimodal distributions (Gaussian for continuous, binomial for discrete).

Table 1. Performance (mean KS distance between target and generated distributions ± standard deviation) of the proposed HAM-MCMC-HRR sampler, MCMC baselines and naïve sampling strategies on 5 generating tasks, and in 4 dynamic configurations. GMM, MVNM-2D, and MVNM-3D refers to a Gaussian mixture model, 2D-, and 3D-multivariate normal mixtures respectively.

Algorithm		Continuous Distributions			Discrete Distributions	
		GMM-1D	MVNM-2D	MVNM-3D	DISCRETE-1	DISCRETE-2
Uniform ITS		0.191 ± 0.001	0.420 ± 0.002	0.425 ± 0.004	0.186 ± 0.003	0.440 ± 0.003
Gaussian ITS		0.181 ± 0.003	0.422 ± 0.002	0.380 ± 0.004	N/A	N/A
Binomial ITS		N/A	N/A	N/A	0.343 ± 0.001	0.503 ± 0.001
KDE MCMC		0.069 ± 0.010	0.311 ± 0.062	0.203 ± 0.048	0.027 ± 0.007	0.055 ± 0.014
HAM-MCMC-DOMAIN		0.097 ± 0.010	0.268 ± 0.024	0.465 ± 0.030	N/A	N/A
HAM-MCMC-HRR						
No Cleanup	Gradient	0.094 ± 0.006	0.172 ± 0.008	0.283 ± 0.009	0.055 ± 0.011	0.115 ± 0.005
	No Gradient	0.535 ± 0.090	0.279 ± 0.012	0.467 ± 0.017	0.067 ± 0.014	0.443 ± 0.062
Cleanup	Gradient	0.103 ± 0.008	0.175 ± 0.009	0.381 ± 0.014	0.340 ± 0.082	0.564 ± 0.075
	No Gradient	0.168 ± 0.016	0.189 ± 0.009	0.389 ± 0.017	0.404 ± 0.095	0.356 ± 0.049
MCMC-SNN-HRR		0.122 ± 0.004	0.240 ± 0.004	0.243 ± 0.002	0.084 ± 0.002	0.223 ± 0.003

3 Results

In our experiments we drew 2,000 samples from both the proposed and baseline algorithms. We evaluated the samplers' performance by calculating the Kolmogorov-Smirnov (KS) distance between the true and sampled empirical distributions. We provide performance statistics (mean and standard error of KS distance) across 10 runs of the tuned model with different seeds in Table 1.

We optimized the SSP approximation parameters (length scale, $\boldsymbol{\lambda}$, and bias, ξ) to minimize the Kullback-Leibler divergence from 3 to the true distribution. Additionally, we optimized the two sampler hyperparameters in Eq. 4 – the gradient step size, γ, and the variance scale factor, σ – to minimize KS distance. The baseline MCMC sampler was tuned through simultaneous optimization of the KDE length scale parameters, B, and transition kernel variance, σ_{KDE}.[2]

3.1 Sampling from Continuous Spaces

To assess algorithm performance, we applied it to 1D, 2D, and 3D Gaussian Mixture Models, all multi-modal. Figure 1A shows examples of the algorithms' performance on continuous target distributions. The HAM-MCMC-HRR sampler captures the essential features of the target distributions in 1D and 2D

[2] The code for all experiments is available at https://gitlab.com/furlong/ssp-sampling. All hyperparameter searches were performed using the Tree-structured Parzen Estimator (TPE) tuner from the NNI library [25]. We rely on the SSPSpace library (https://github.com/ctn-waterloo/sspspace). KS distance was computed using functions from either scipy, ndtest (https://github.com/syrte/ndtest), or genai-evaluation (https://github.com/rajiviyer/genai_evaluation) Python packages for 1D, 2D, and 3+D data, respectively. The spiking neural network was simulated using Nengo (https://github.com/nengo/nengo).

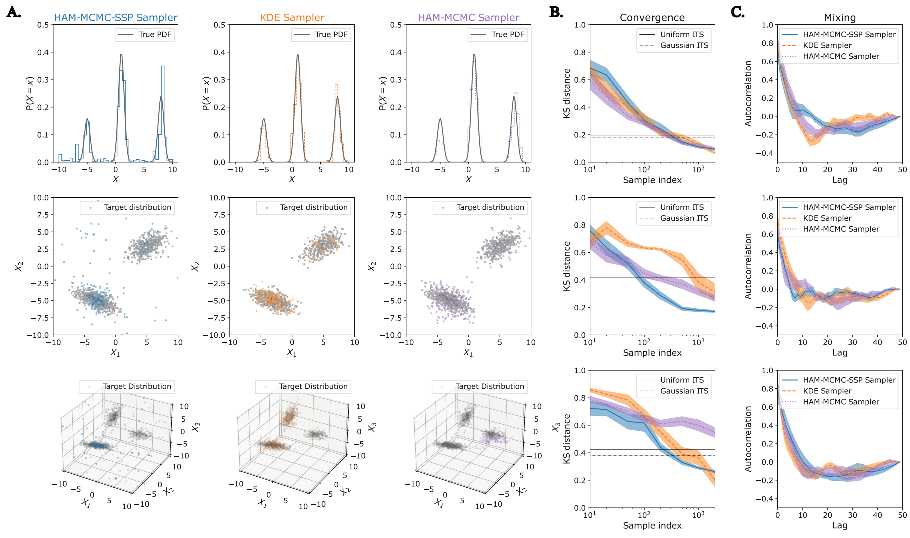

Fig. 1. Performance of three MCMC sampling algorithms on 1D (top), 2D (middle), and 3D (bottom) multimodal generation tasks. **A.** Representative distributions generated by the samplers against target distributions. **B.** Convergence onto the stationary distribution with sampling timesteps. **C.** Correlation between successive samples drawn.

spaces. On these tasks, the algorithm explores all modes of the distribution modes and draws few extraneous samples, similar to the MCMC baselines. The HAM-MCMC-HRR algorithm appears to break down on the 3D task, however, as only one mode is thoroughly explored and there is an increased contribution from low-probability samples. Here, however, its behavior is still superior to the HAM-MCMC baseline, which fails to sample from even one of the modes. The summary statistics in Table 1 show that the HAM-MCMC-HRR sampler performs similarly to the KDE MCMC baseline, either outperforming or slightly underperforming depending on the task. In terms of temporal behavior, the HAM-MCMC-HRR sampler either keeps pace with or overtakes the baselines in converging to the stationary distribution (Fig. 1B), and achieves similarly rapid decorrelation (Fig. 1C).

3.2 Sampling from Discrete Spaces

We also test the algorithm on two categorical distributions, $X \in \{1, \ldots, N\}$. In the first, $N = 10$[3] and in the second, $N = 50$[4] In these experiments, length

[3] The $N = 10$ distribution is generated by randomly selecting values $v_i \sim \mathcal{U}[0, 100]$, and specifying $P(X = n) = \frac{1}{\sum_i v_i} v_n$.

[4] The $N = 50$ distribution follows Zipf's law – i.e., $P(X = n) = \frac{1}{H_N}\frac{1}{n}$, where H_N is the N^{th} harmonic number.

scale is not a meaningful parameter, however, we do optimize the bias, transition kernel variance, and gradient step size. Results are shown in Fig. 2. The samples from the MCMC HRR algorithm fit the distribution less accurately than the samples generated from the discrete MCMC sampler (Fig. 2, bottom row). Convergence analysis suggests the sampler may be a biased estimator of the categorical distribution. Admittedly, MCMC sampling is excessive for these simple distributions. However, what we have demonstrated is the ability to sample discrete objects encoded using a VSA embedding, using a simple dynamic system, despite the fact that the encoding scheme does not have a continuous manifold.

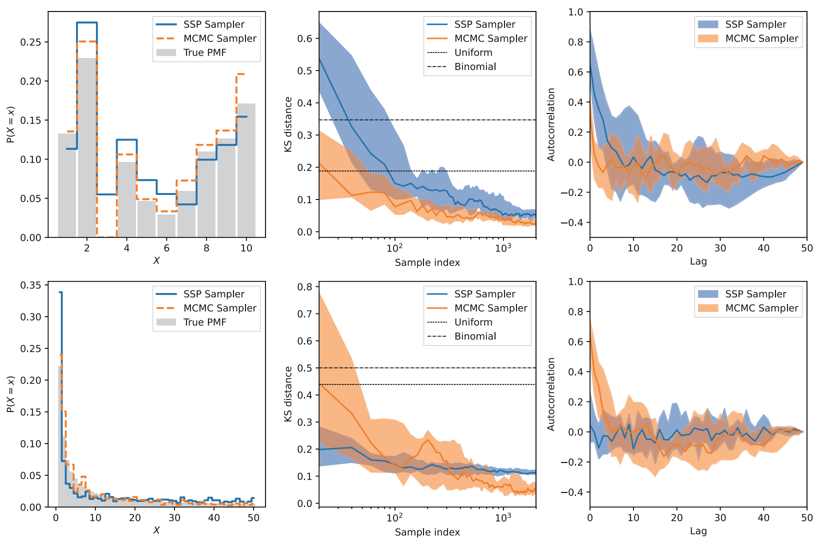

Fig. 2. Top row: results for sampling from a categorical distribution with 10 values. Bottom row: results for sampling from a 50-element categorial distribution. In both cases, the proposed algorithm converges to a smaller error than naïve sampling, and samples are not highly correlated. The proposed sampler appears to converge to a fixed error in the first 2000 samples.

3.3 Assessing the Contribution of Peri-Manifold Dynamics

Table 1 presents the results of experiments investigating the importance of staying on or near the semantic pointer manifold in reproducing the target distribution. Among the No Cleanup samplers, those with a gradient term ($\gamma > 0$) outperformed the sampler governed by pure noise ($\gamma = 0$). This confirms that the decoding scheme alone cannot account for performance and that the gradient information is necessary to sample these latent representations. This also indicates that the gradient information constrains trajectories sufficiently near the manifold such that decoded points are in the support of the target distribution.

In-the-loop Cleanup could not compensate for the lack of gradient information on any task. On the 2D task, a sampler with cleanup but without gradient information achieved a KS distance of only0.189 ± 0.009, which was surpassed by the configuration with gradient information and no cleanup (KS distance $= 0.172 \pm 0.008$). On other tasks, the performance degradation was much worse. On the continuous distributions, introducing cleanup had little effect on performance, while on the categorical sampling tasks it caused a striking degradation in performance. For example, we observe an increase in KS distance from 0.055 ± 0.011 to 0.340 ± 0.082 for the first discrete distribution, for samplers with gradient information. These findings indicate that in-the-loop cleanup is neither necessary nor sufficient for sampling from these spaces.

3.4 Leveraging VSA Operations for Probabilistic Programming

As discussed in Sect. 2.1, we are able to use unbinding to produce an unnormalized conditional distribution. Figure 3A demonstrates sampling from distributions conditioned on various values. Without changing the sampling algorithm, meaningful samples are still drawn from the conditioned distribution. As can be seen, samples are predominantly drawn from high probability regions.

We also considered a simple mixed discrete-continuous distribution using bound discrete and spatial semantic pointers. The same algorithm was successful in sampling from that distribution. Figure 3B shows the ability to reconstruct the joint distribution over the discrete variable $X \in \{A, B, C\}$ and the continuous variable $Y \in [0, 1]$[5]. Note that for the conditional distributions over Y, the SSP approximation performs worse for $X = B$ and $X = C$, than for $X = A$. This is a reflection of only using one set of length scale parameters for the encoding scheme, while the distribution's bandwidth changes. With that in mind, the samples drawn (indicated by the vertical bars) do appear to be sampling the approximated distribution.

3.5 Evaluation in a Spiking Neural Network

The SNN SSP sampler was evaluated using the same benchmarks as the algebraic SSP sampler, with results detailed in Table 1. An example of its output and spiking activity when applied to a 1D Gaussian Mixture Model is depicted Fig. 4. As anticipated, the SNN sampler underperformed compared to the closest non-spiking counterpart – Hamiltonian MCMC SSP sampler with Gradient and No Cleanup – primarily due to its absence of the Metropolis acceptance criteria and the inherent approximations introduced by spiking neurons. However, Fig. 4 illustrates the SNN model's capability to sample from all modes on a target 1D distributions. Additionally, the SNN sampler surpasses the naïve Uniform Inverse Transform Sampling (ITS) and Gaussian ITS methods and, for the MVNM-2D, surpasses the performance of the KDE MCMC baseline.

[5] The marginal distribution over $X \in \{A, B, C\}$ is defined as a multinomial distribution, and the conditional distributions over $Y \in [0, 1]$ are Beta distributions.

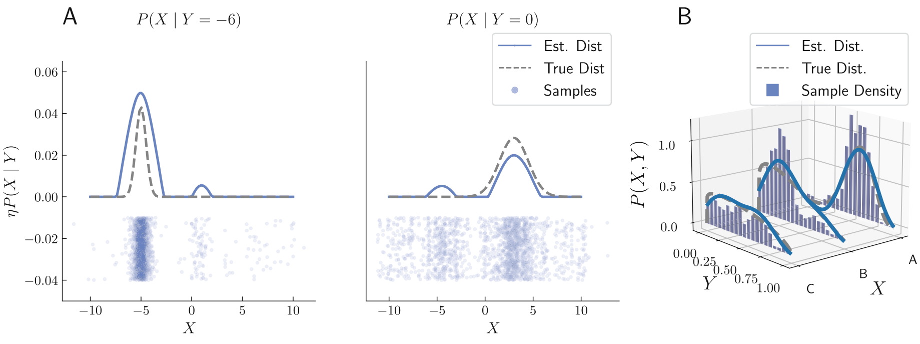

Fig. 3. A. Plots show distributions represented by $\mu_{\mathrm{XY}} \circledast \phi_{\mathrm{Y}}^{-1}(y)$, for $y \in \{-6, 0\}$, where μ_{XY} is a kernel embedding of a 2D GMM model. Blue dots (vertical positions are randomly assigned) indicate the 2000 samples drawn from the distribution. **B.** Sampling a joint distribution over the space $\{\mathrm{A}, \mathrm{B}, \mathrm{C}\} \times [0, 1]$.

4 Discussion and Conclusion

In this paper, we set out to create a biologically-plausible sampling algorithm that can sample from cognitive representations of uncertainty. We developed and tested a novel sampling algorithm that operates on VSA representations, and show that the core dynamics can be realized in an SNN. We argue for our algorithm's biological plausibility, noting that it can be implemented in neural dynamics, operates on latent brain representations, and does not require gradient knowledge, which is difficult for neural circuits to compute and transport. Our approach successfully approximates distributions across discrete, continuous, and mixed data using HRR algebra without altering the sampling algorithm.

Our experiments found cleanup memory to be unnecessary and potentially problematic for sampling (see Table 1), indicating that our approach can avoid encoding-specific constraints that would limit its generality. Relying solely on $\boldsymbol{\mu}$, our algorithm flexibly handles various data types and requires less computational effort than models with cleanup or dependent on data encoding scheme gradients.

Sampling from discrete distributions demonstrates that the method generalizes to distributions over symbols represented in high-dimensional spaces. Unlike the SSP representations of continuous data, there was no underlying manifold to navigate. Given that these discrete representations are used to represent high-level concepts in non-metric spaces, unlike SSPs, it is impressive that these distributions can be sampled by a dynamical system with a continuously-varying state. This suggests that the sampler generalizes to approximate probabilistic inference in models of high-level cognition. Of course, if biological-plausibility is not required, sampling from discrete distributions could be achieved with inverse transform sampling or normalizing flows, and with better performance.

Sampling from conditioned distributions has potential applications in Reinforcement Learning (RL). RL involves the problem of learning a policy over states and preferred actions to take in those states. Once this policy is available,

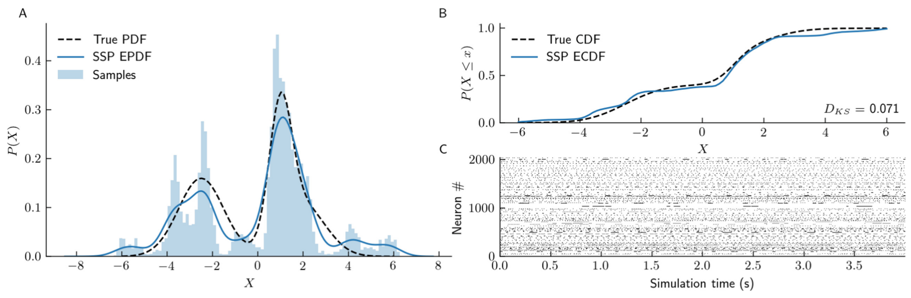

Fig. 4. A. A histogram of output of the SNN SSP sampler, along with the PDF of the true distribution and the KDE from the samples. **B.** The empirical CDF, obtained via the sampler output, versus the true CDF. C. A spike raster plot of the spike activity of all neurons in the recurrent sampler SNN over the simulation run time.

taking actions can be thought of as sampling from that policy, conditioned on the agent's current state. Implementing RL with VSA-encoded policies requires a means of sampling from VSA-encoded distributions, provided in the project git repository (see footnote 2).

The presented work extends the utility of VSAs as a probabilistic framework by providing a biologically-plausible MCMC sampler that can extract samples from non-linear, latent space encodings. This advance can support future work into constructing cognitive models that sample distributions over higher-level concepts. This would pave the way for evaluating a sampler as an explanatory model of human cognition, towards understanding deficits in probabilistic reasoning.

References

1. Anderson, C.H., Van Essen, D.C.: Neurobiological computational systems. Comput. Intell. Imitating Life **213222** (1994)
2. Anderson, J.R., John, B.E., Just, M.A., Carpenter, P.A., Kieras, D.E., Meyer, D.: Production system models of complex cognition. In: Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society, pp. 9–12 (1995)
3. Barber, M.J., Clark, J.W., Anderson, C.H.: Neural representation of probabilistic information. Neural Comput. **15**(8), 1843–1864 (2003)
4. Buesing, L., Bill, J., Nessler, B., Maass, W.: Neural dynamics as sampling: a model for stochastic computation in recurrent networks of spiking neurons. PLoS Comput. Biol. **7**(11), e1002211 (2011)
5. Chater, N., Oaksford, M.: The probabilistic mind: Prospects for Bayesian cognitive science. Oxford University Press, USA (2008)
6. Darlington, T.R., Beck, J.M., Lisberger, S.G.: Neural implementation of bayesian inference in a sensorimotor behavior. Nat. Neurosci. **21**(10), 1442–1451 (2018)
7. Deneve, S.: Bayesian spiking neurons i: inference. Neural Comput. **20**(1), 91–117 (2008)
8. Doya, K., Ishii, S., Pouget, A., Rao, R.P.: Bayesian brain: Probabilistic approaches to neural coding. MIT press (2007)

9. Eliasmith, C.: How to build a brain: A neural architecture for biological cognition. Oxford University Press (2013)
10. Eliasmith, C., Anderson, C.H.: Neural engineering: Computation, representation, and dynamics in neurobiological systems. MIT press (2003)
11. Frady, E.P., Kleyko, D., Kymn, C.J., Olshausen, B.A., Sommer, F.T.: Computing on functions using randomized vector representations. arXiv preprint arXiv:2109.03429 (2021)
12. Furlong, M., Eliasmith, C.: Fractional binding in vector symbolic architectures as quasi-probability statements. In: Proceedings of the Annual Meeting of the Cognitive Science Society, vol. 44 (2022)
13. Gayler, R.W.: Vector symbolic architectures answer jackendoff's challenges for cognitive neuroscience. arXiv preprint cs/0412059 (2004)
14. Glad, I.K., Hjort, N.L., Ushakov, N.G.: Correction of density estimators that are not densities. Scand. J. Stat. **30**(2), 415–427 (2003)
15. Goodman, N.D., Tenenbaum, J.B., Contributors, T.P.: Probabilistic Models of Cognition. http://probmods.org/v2 (2016), Accessed 23 Jan 2023
16. Gosmann, J., Eliasmith, C.: Optimizing semantic pointer representations for symbol-like processing in spiking neural networks. PLoS ONE **11**(2), e0149928 (2016)
17. Grabska-Barwinska, A., Beck, J., Pouget, A., Latham, P.: Demixing odors-fast inference in olfaction. Adv. Neural Inform. Process. Syst. **26** (2013)
18. Hennequin, G., Aitchison, L., Lengyel, M.: Fast sampling-based inference in balanced neuronal networks. Adv. Neural Inform. Process. Syst. **27** (2014)
19. Hou, H., Zheng, Q., Zhao, Y., Pouget, A., Gu, Y.: Neural correlates of optimal multisensory decision making under time-varying reliabilities with an invariant linear probabilistic population code. Neuron **104**(5), 1010–1021 (2019)
20. Kanerva, P.: Binary spatter-coding of ordered K-tuples. In: von der Malsburg, C., von Seelen, W., Vorbrüggen, J.C., Sendhoff, B. (eds.) ICANN 1996. LNCS, vol. 1112, pp. 869–873. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-61510-5_146
21. Komer, B., Stewart, T.C., Voelker, A., Eliasmith, C.: A neural representation of continuous space using fractional binding. In: CogSci, pp. 2038–2043 (2019)
22. Levy, S.D., Gayler, R.: Vector symbolic architectures: a new building material for artificial general intelligence. In: Proceedings of the 2008 Conference on Artificial General Intelligence 2008: Proceedings of the First AGI Conference, pp. 414–418 (2008)
23. Ma, W.J., Beck, J.M., Latham, P.E., Pouget, A.: Bayesian inference with probabilistic population codes. Nat. Neurosci. **9**(11), 1432–1438 (2006)
24. Ma, W.J., Beck, J.M., Pouget, A.: Spiking networks for bayesian inference and choice. Curr. Opin. Neurobiol. **18**(2), 217–222 (2008)
25. Microsoft: Neural Network Intelligence (Jan 2021). https://github.com/microsoft/nni
26. Muandet, K., Fukumizu, K., Sriperumbudur, B., Schölkopf, B., et al.: Kernel mean embedding of distributions: a review and beyond. Foundations Trends® Mach. Learn. **10**(1-2), 1–141 (2017)
27. Pecevski, D., Buesing, L., Maass, W.: Probabilistic inference in general graphical models through sampling in stochastic networks of spiking neurons. PLoS Comput. Biol. **7**(12), e1002294 (2011)
28. Plate, T.A.: Holographic recurrent networks. Adv. Neural Inform. Process. Syst. **5** (1992)

29. Plate, T.A.: Distributed representations and nested compositional structure. Ph.D. thesis, University of Toronto, Toronto, ON, Canada (1994)
30. Plate, T.A.: Holographic reduced representations. IEEE Trans. Neural Netw. **6**(3), 623–641 (1995)
31. Rao, R.P.: Bayesian computation in recurrent neural circuits. Neural Comput. **16**(1), 1–38 (2004)
32. Salinas, E., Abbott, L.: Vector reconstruction from firing rates. J. Comput. Neurosci. **1**(1–2), 89–107 (1994)
33. Sanborn, A.N., Chater, N.: Bayesian brains without probabilities. Trends Cogn. Sci. **20**(12), 883–893 (2016)
34. Savin, C., Denève, S.: Spatio-temporal representations of uncertainty in spiking neural networks. Adv. Neural Inform. Process. Syst. **27** (2014)
35. Schuster, I., Strathmann, H., Paige, B., Sejdinovic, D.: Kernel sequential monte carlo. In: Ceci, M., Hollmén, J., Todorovski, L., Vens, C., Džeroski, S. (eds.) ECML PKDD 2017. LNCS (LNAI), vol. 10534, pp. 390–409. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-71249-9_24
36. Sejdinovic, D., Strathmann, H., Garcia, M.L., Andrieu, C., Gretton, A.: Kernel adaptive metropolis-hastings. In: International Conference on Machine Learning, pp. 1665–1673, PMLR (2014)
37. Smolensky, P.: Tensor product variable binding and the representation of symbolic structures in connectionist systems. Artif. Intell. **46**(1–2), 159–216 (1990)
38. Stewart, T.C., Eliasmith, C.: Large-scale synthesis of functional spiking neural circuits. Proceedings of the IEEE **102**(5), 881–898 (2014). https://doi.org/10.1109/JPROC.2014.2306061, ISSN 0018-9219
39. Strathmann, H., Sejdinovic, D., Livingstone, S., Szabo, Z., Gretton, A.: Gradient-free hamiltonian monte carlo with efficient kernel exponential families. Adv. Neural Inform. Process. Syst. **28** (2015)
40. Voelker, A.R., Gosmann, J., Stewart, T.C.: Efficiently sampling vectors and coordinates from the n-sphere and n-ball. Centre for Theoretical Neuroscience-Technical Report **1** (2017)
41. Walker, E.Y., Cotton, R.J., Ma, W.J., Tolias, A.S.: A neural basis of probabilistic computation in visual cortex. Nat. Neurosci. **23**(1), 122–129 (2020)
42. Zemel, R., Dayan, P., Pouget, A.: Probabilistic interpretation of population codes. Adv. Neural Inform. Process. Syst. **9** (1996)