
Product Manifold Machine Learning for Physics

Nathaniel S. Woodward^{1,2*}
nswood@mit.edu

Sang Eon Park^{1,2}
sangeon@mit.edu

Gaia Grosso^{1,2,3}
gaia.grosso@cern.ch

Jeffrey Krupa⁴
jkrupa@slac.stanford.edu

Philip Harris^{1,2}
pcharris@mit.edu

¹Laboratory of Nuclear Science, Massachusetts Institute of Technology, Cambridge, MA

²The NSF AI Institute for Artificial Intelligence and Fundamental Interactions

³School of Engineering and Applied Sciences, Harvard University, Cambridge, MA

⁴SLAC National Accelerator Laboratory, Menlo Park, CA 94025

Abstract

Particle jets are collimated flows of partons which evolve as tree-like structures through stochastic parton showering and hadronization. The hierarchical nature of particle jets aligns naturally with hyperbolic space, a non-Euclidean geometry that captures hierarchy intrinsically. To leverage the benefits of non-Euclidean geometries, we develop jet analysis in product manifold (\mathcal{PM}) spaces, Cartesian products of constant curvature Riemannian manifolds. We consider particle representations as configurable parameters and compare the performance of \mathcal{PM} multilayer perceptron models across several possible representations. We find product manifold representations perform equal or better in particle jet classification than fully Euclidean models of the same latent dimension and the same approximate number of parameters. These findings reinforce the view of optimizing geometric representations as a key parameter in maximizing both performance and efficiency.

1 Introduction

Hard QCD scattering processes generate *particle jets* (jets), collimated flows of partons, which shower through fragmentation and emission of soft radiation until a final hadronization stage. Parton fragmentation, the process of high-energy partons splitting into several lower-energy partons, creates strong hierarchical relationships among the constituents of particle jets. We view this hierarchical structure as a fundamental feature of particle jets. Hyperbolic spaces provide a natural representation for hierarchies due to their exponential growth in volume with distance², however, it is unclear if this representation is optimal for all features of particle jets. This suggests that distinct sets of features may be optimally represented in different geometric spaces.

In this work, we present a new representation for particle jets through Cartesian products of constant curvature Riemannian manifolds which we refer to as product manifolds (\mathcal{PM}). Cartesian products of manifolds enable the simultaneous processing of data representations across multiple manifolds, offering several unique perspectives on the dataset. Park et. al [11] explored the use of hyperbolic geometries in the final layers of models for jet analysis, displaying the hierarchies of embedding spaces. We expand this approach by developing model architectures compatible with jets represented

*Corresponding Author

²This is understood through comparing the distortion of tree graphs embeddings in different geometric spaces. Euclidean spaces are unable to achieve comparably low-distortion embeddings compared to hyperbolic spaces [7, 12].

in product manifolds throughout the model. In this work, we focus on \mathcal{PM} representations exclusively at the *particle-level*, embedding individual jet constituents in these spaces. To utilize product manifold representations, we adapt a multilayer perceptrons (MLPs) to process data in product manifold representations of particle-level features. We develop highly generalized architectures that allow seamless use of any product manifold representation and systematic searches for the optimal one.

2 Mathematical Prerequisites

We provide an overview of concepts necessary for this work; refer to [10, 16] further details.

Riemannian Manifolds A d -dimensional Riemannian manifold (\mathcal{M}, g) is a smooth manifold, denoted as \mathcal{M}^d , together with a Riemannian metric g which determines the curvature κ at each point $x \in \mathcal{M}^d$. In this work, we focus on constant curvature manifolds, where curvature is uniform across the entire space: $\kappa < 0$ for hyperbolic spaces \mathbb{H} , $\kappa = 0$ for Euclidean spaces \mathbb{R} , and $\kappa > 0$ for spherical spaces \mathbb{S} . Since \mathcal{M}^d is a smooth manifold, each point $x \in \mathcal{M}^d$ is equipped with a tangent space $\mathcal{T}_x \mathcal{M}^d \subseteq \mathbb{R}^d$, providing a local, linear approximation of the manifold near x . The exponential (eq. 1) and logarithmic (eq. 2) maps are employed throughout the \mathcal{PM} machine learning models presented to map latent vectors between Euclidean space and the manifold in use.

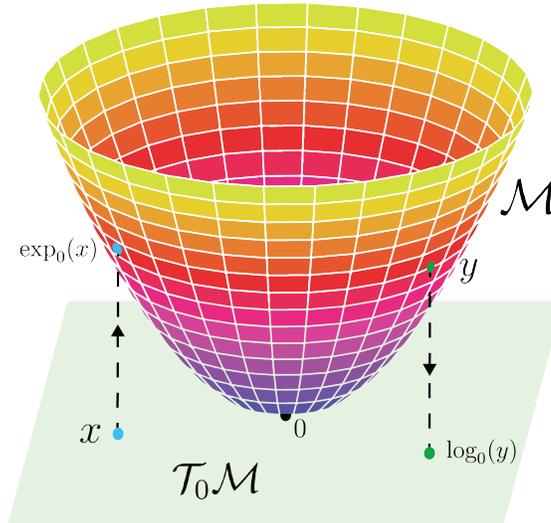


Figure 1: A graphical depiction of a manifold \mathcal{M} together with the tangent space at the point $x = 0$, denoted $\mathcal{T}_0\mathcal{M}$ and shown in green. We further illustrate the use of the exponential map, which for $x \in \mathcal{T}_0\mathcal{M}$ takes $x \rightarrow \exp_0(x) \in \mathcal{M}$, and the logarithmic map, which for $y \in \mathcal{M}$ takes $y \rightarrow \log_0(x) \in \mathcal{T}_0\mathcal{M}$. A portion of this image is adapted from [6].

$$\exp_u(\cdot) : \mathcal{T}_u \mathcal{M}^d \rightarrow \mathcal{M}^d, \quad \exp_u(v) = \gamma_{u,v}(1) = u \oplus_{\kappa} \tan_{\kappa} \left(\frac{\|v\|_u}{2} \right) \frac{v}{\|v\|_2} \quad (1)$$

$$\log_u(\cdot) : \mathcal{M}^d \rightarrow \mathcal{T}_u \mathcal{M}^d, \quad \log_u(w) = \frac{2}{\lambda_u^{\kappa}} \tan_{\kappa}^{-1}(\|(-u) \oplus_{\kappa} w\|_2) \frac{(-u) \oplus_{\kappa} w}{\|(-u) \oplus_{\kappa} w\|_2} \quad (2)$$

Product Spaces As outlined in section 1, we employ a general data representation formed by combining several distinct manifolds. The product space \mathcal{P} is defined by linking manifolds using the Cartesian product:

$$\mathcal{P} = \mathcal{M}_{\kappa_1}^{d_1} \times \mathcal{M}_{\kappa_2}^{d_2} \times \dots \times \mathcal{M}_{\kappa_n}^{d_n} \quad (3)$$

The total dimension of \mathcal{P} is equal to the sum of each individual manifold, which we denote as $d = \sum_{i=1}^n d_i$. We refer to product space representations as *product manifolds* (\mathcal{PM}).

Representation Models We utilize stereographic projection representations of hyperbolic and spherical spaces, the Poincaré ball model and the stereographic spherical projection model, respectively. We see these as the natural choice for product manifold approaches as they have a unified gyrovector formalism developed by Bachmann et. al [1].

Gyrovector Spaces Gyrovector spaces were developed by Ungar for hyperbolic spaces ($\kappa < 0$) [16] and extended to stereographic spherical projections geometries ($\kappa > 0$) by Bachmann et. al [1]. When performing calculations in non-Euclidean spaces in this work, Euclidean vector operations are replaced by gyrovector operations, shown in Table 1 which highlights the increased complexity required for gyrovector operations.

Euclidean Operations	Gyrovector Analogs	G./E. FLOPS Ratio
$x + y$	$x \oplus_{\kappa} y = \frac{(1-2\kappa\langle x, y \rangle - \kappa\ y\ _2^2)x + (1+\kappa\ x\ _2^2)y}{1-2\kappa\langle x, y \rangle + \kappa^2\ x\ _2^2\ y\ _2^2}$	5.1
$r\mathbf{x}$	$r \otimes_{\kappa} x = \tan_{\kappa}(r \tan_{\kappa}^{-1}(\ x\ _2)) \frac{x}{\ x\ _2}$	3
$ x - y $	$d_{\kappa}(x, y) = 2 \tan_{\kappa}^{-1}(\ (-x) \oplus_{\kappa} y\ _2)$	4.5
Mx	$M \otimes_{\kappa} x = \tan_{\kappa}\left(\frac{\ Mx\ _2}{\ x\ _2} \tan_{\kappa}^{-1}(\ x\ _2)\right) \frac{Mx}{\ Mx\ _2}$	2.8

Table 1: We show Euclidean vector operations and their gyrovector analogs. Floating-point operations per second (FLOPS) ratios are presented for 4D vector/gyrovector operations, bringing an $\mathcal{O}(n)$ FLOP correction.

Gromov- δ Definition of Curvature Gromov- δ hyperbolicity estimates the degree of hierarchical structure in the dataset [4], where the hyperbolicity δ is chosen such that for any points $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}$ in a δ -hyperbolic, they satisfy,

$$(\mathbf{a}, \mathbf{b})_{\mathbf{d}} \geq \min\{(\mathbf{b}, \mathbf{c})_{\mathbf{d}}, (\mathbf{a}, \mathbf{c})_{\mathbf{d}}\} - \delta, \quad \forall \mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d} \in \mathcal{M} \quad (4)$$

$$(\mathbf{x}, \mathbf{y})_{\mathbf{z}} = \frac{1}{2}(d_{\mathbb{R}}(x, z) + d_{\mathbb{R}}(y, z) - d_{\mathbb{R}}(x, y)) \quad (5)$$

where $(\mathbf{x}, \mathbf{y})_{\mathbf{z}}$ is the Gromov product in Eq. 5 for the Euclidean distance metric $d_{\mathbb{R}}$. The Gromov- δ hyperbolicity is related to the scalar curvature of the manifold through an inverse square relation.

3 Manifold Machine Learning

Our implementation relies upon Geopt [5], a package for Riemannian optimization in PyTorch.

Fully Connected Layers Ganea et. al [3] proposed utilizing gyrovector operations to develop fully connected layers applicable for Riemannian manifolds of all curvature. For weights $W = [\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n]$ with $\mathbf{v}_i \in \mathcal{M}_{\kappa}^m$, bias $\mathbf{b} \in \mathcal{M}_{\kappa}^m$, and input $\mathbf{x} \in \mathcal{M}_{\kappa}^n$ we can calculate the output $\mathbf{y} \in \mathcal{M}_{\kappa}^m$ as:

$$FC(\mathbf{x}; n, m, \kappa) = W \otimes_{\kappa} \mathbf{x} \oplus_{\kappa} \mathbf{b} \quad (6)$$

where \otimes_{κ} and \oplus_{κ} are the gyrovector matrix multiplication and vector addition, respectively. We will refer to several fully connected layers in product manifold models as \mathcal{M} -MLP (manifold MLP) and we reserve MLP for the Euclidean MLP.

Activation Functions There are many approaches in the literature to formulating activation functions in non-Euclidean spaces [1, 2, 8, 12]. In our testing we have found that RELU activation functions in non-Euclidean representations yields minor improvements compared to no activation functions. This result is unsurprising as activations inject non-linearity, however, non-Euclidean spaces are already non-linear.

LayerNorm Implementing normalization layers for non-Euclidean machine learning has remained a challenge [12]. Rigorous methods such as those proposed by [9] rely on iterative calculations which result in a detrimental bottleneck for deep models. To avoid these challenges, we implement LayerNorm in the tangent space. This provides a simple method for normalization across all manifolds considered without significant negative impacts on training or inference times.

Inter-Manifold Attention Sun et al. [15] formulated inter-manifold attention to quantify relationships between data represented across manifolds. Inter-manifold attention probabilities are calculated for each \mathcal{PM} data point using traditional Euclidean attention over the tangent space representation of each manifold’s points in the product manifold representation. We scale the product manifold representations by the attention probabilities using the gyrovector scalar multiplication operation.

4 Product Manifold Multilayer Perceptron (\mathcal{PM} -MLP)

The \mathcal{PM} -MLP architecture is a simple \mathcal{M} -MLP based model architecture, configurable for any possible product-manifold at the particle-level. A schematic view of the model’s architecture is shown in Figure 2. Individual manifolds in \mathcal{P} are shown as $\mathcal{M}_{\kappa_i}^{d_i}$ in the figure, where the curvature κ_i is a learnable parameter.

Jet constituents, each with f features, are mapped to the particle-level product manifold \mathcal{P} , processed through 3-layer \mathcal{M} -MLP with ReLU activations, a tangent space LayerNorm ($\mathcal{T}LN$), and an inter-manifold attention layer (if the particle-level representation is more than a single manifold). The outputs are mean aggregated in the tangent space and concatenated, forming a jet-level latent vector, which is processed in Euclidean space with a 3-layer MLP with ReLU activations and a final SoftMax activation for predictions. We refer to \mathcal{PM} -MLP models by their particle-level representation $\mathcal{P} = \mathcal{M}^{d_1} \times \dots \times \mathcal{M}^{d_n}$.

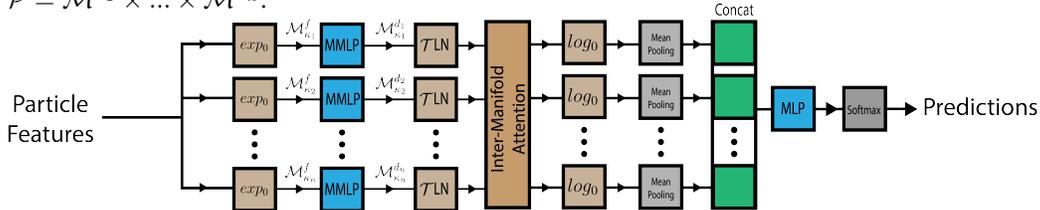


Figure 2: A schematic of the \mathcal{PM} -MLP model

5 Improving Performance in Binary Jet Tagging with \mathcal{PM} -MLP Models

Dataset We utilize the open source JETCLASS³ dataset [13, 14]. The JETCLASS dataset consists of ten classes of simulated jets, represented as point clouds of jet constituents. Model inputs are the 20 highest p_T particles⁴ in each jet with particle features as kinematic information ($\Delta\eta, \Delta\phi, \log p_T, \log E$) and particle identification⁵. We estimate the hierarchical structure of particle features for each class through the Gromov- δ hyperbolicity and find the $H \rightarrow 4q$ to be the most hierarchical, while $t \rightarrow bqq'$ is the least hierarchical.

Training Details We train all \mathcal{PM} -MLP models over ten initializations, each with 50 epochs and a batch size of 1024 samples, and select the best performing model for comparison. Training (5M), validation (200k), and test (200k) datasets are equal distribution of the signal and QCD background. We use the Riemannian Adam optimizer[5] with weight decay of $5e-4$, $\beta_1 = 0.9$, and $\beta_2 = 0.98$. In the case of fully Euclidean models, Riemannian Adam is equivalent to the Adam optimizer. The training uses a cosine annealing learning rate scheduler with an initial learning rate of $1e-3$ which remains constant for 30% of the iterations and then decays exponentially to 1% of the initial learning rate, following the training approach of [13]. Hyperbolic and spherical spaces are initialized with $\kappa = -1.2$ and $\kappa = 1$, respectively. For repeated geometries, we scale the initial curvatures to avoid redundancy. For example, in $\mathbb{H}^{\frac{d}{2}} \times \mathbb{H}^{\frac{d}{2}}$, one manifold is initialized with $\kappa = -1.2$ and the other with $\kappa = -2.4$. All models are trained on a single A100 GPU.

Results We compare binary classification performance for $H \rightarrow 4q$ vs QCD and $t \rightarrow bqq'$ vs QCD across several particle-level representations with total dimension $d = [2, 4, 8, 16, 32]$. We do not present results for fully spherical representations as they were found to be detrimental to performance. We include $\mathbb{R}^{\frac{d}{2}} \times \mathbb{R}^{\frac{d}{2}}$, which is geometrically equivalent to \mathbb{R}^d , to emphasize the impact of parallel representations and inter-manifold attention. Model sizes in this experiment range from 121 to 1.8k tunable parameters. We report classification accuracy on the test dataset and quantify uncertainty through the bootstrapped test accuracy using 50 samples with 12.5% of the test dataset per sample. Results are shown in Figure 3 for $H \rightarrow 4q$ vs QCD and $t \rightarrow bqq'$ vs QCD on the left and right, respectively. To analyze the structure of these embeddings, we present the tangent space corner plots of particle embeddings for \mathcal{PM} -MLP models with \mathbb{R}^4 and \mathbb{H}^4 geometries in Appendix A. At this

³The dataset can be found here: <https://zenodo.org/records/6619768>

⁴We restrict our analysis to a small set of particles due to the use of compact models and the inability to mask zero-padded particle inputs.

⁵Further details are listed in Table 2 of [13]

point, we cannot draft definitive conclusions from these structure of these particle embedding and we will work to better understand these space in future work.

For highly hierarchical classes ($H \rightarrow 4q$), hyperbolic representations outperforms Euclidean representations at dimensions 2 and 4, while achieving comparable performance at higher dimensions. For weakly hierarchical classes ($t \rightarrow bqq'$), the performance of \mathcal{PM} representations is generally within the uncertainty range of fully Euclidean representations. This performance aligns with expectations based on the hierarchical structure inherent in the dataset, illustrating that the Gromov- δ framework effectively underscores the advantages of hyperbolic representations in enhancing model performance. In both processes, combining Euclidean and non-Euclidean representations does not yield performance gains for \mathcal{PM} -MLP models. This suggests that low-parameter model and simple architectures may not be able to effectively leverage the additional perspectives through parallel representations. This motivates future explorations of \mathcal{PM} representations in large transformer models.

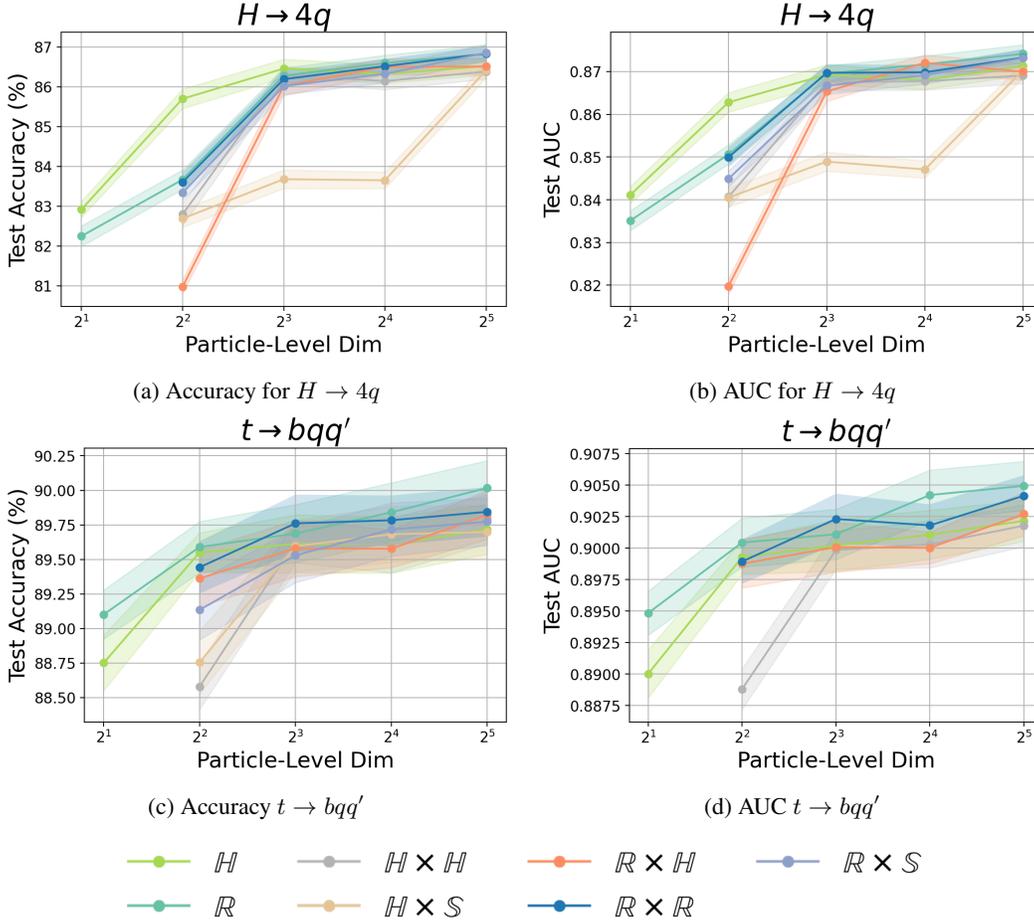


Figure 3: \mathcal{PM} representations bring performance gains in highly hierarchical signals ($H \rightarrow 4q$) and approximately equal performance in weakly hierarchical signals ($t \rightarrow bqq'$). For highly hierarchical signals, fully hyperbolic representations \mathbb{H} (shown in green) brings the largest gains.

6 Conclusion

We find that \mathcal{PM} representations at the particle-level can bring gains in jet classification tasks with no impact on total dimension and minimal adjustments to model parameters and FLOPS. Furthermore, these gains follow the predicted hierarchical structure by the Gromov- δ estimation. These results showcase the potential for \mathcal{PM} representations as a new, highly tunable representation for particle jets in physics analysis. In future works, we will explore embedding jet latent representations in \mathcal{PM} spaces and more complex model architectures through the \mathcal{PM} -Transformer model.

7 Acknowledgements

This work is supported by the National Science Foundation under Cooperative Agreement PHY-2019786 (The NSF AI Institute for Artificial Intelligence and Fundamental Interactions, <http://iaifi.org/>).

References

- [1] Gregor Bachmann, Gary Bécigneul, and Octavian-Eugen Ganea. Constant curvature graph convolutional networks. 119:486–496, 07 2020.
- [2] Taco S. Cohen, Mario Geiger, Jonas Koehler, and Max Welling. Spherical cnns, 2018.
- [3] Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. *arXiv (Cornell University)*, 31:5345–5355, 05 2018.
- [4] Valentin Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan Oseledets, and Victor Lempitsky. Hyperbolic image embeddings.
- [5] Max Kochurov, Rasul Karimov, and Serge Kozlukov. Geoopt: Riemannian optimization in pytorch, 2020.
- [6] Krishnavedala. Paraboloid of revolution, 2012. [Online; accessed 25-August-2024].
- [7] Nati Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.
- [8] Qi Liu, Maximilian Nickel, and Douwe Kiela. Hyperbolic graph neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.
- [9] Aaron Lou, Isay Katsman, Qingxuan Jiang, Serge Belongie, Ser-Nam Lim, and Christopher De Sa. Differentiating through the fréchet mean, 2021.
- [10] Barrett O’Neill. *Elementary Differential Geometry*. Taylor Francis US, 1997.
- [11] Sang Eon Park, Philip Harris, and Bryan Ostdiek. Neural embedding: learning the embedding of the manifold of physics data. *The Journal of high energy physics/The journal of high energy physics*, 2023, 07 2023.
- [12] Wei Peng, Tuomas Varanka, Abdelrahman Mostafa, Henglin Shi, and Guoying Zhao. Hyperbolic deep neural networks: A survey, 2021.
- [13] H Qu, Congqiao Li, and S J Qian. Particle transformer for jet tagging. *arXiv (Cornell University)*, 02 2022.
- [14] Huilin Qu, Congqiao Li, and Sitian Qian. JetClass: A large-scale dataset for deep learning in jet physics, jun 2022.
- [15] Li Sun, Zhongbao Zhang, Junda Ye, Hao Peng, Jiawei Zhang, Sen Su, and Philip S Yu. A self-supervised mixed-curvature graph neural network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36:4146–4155, 06 2022.
- [16] Abraham A Ungar. *Analytic Hyperbolic Geometry*. World Scientific, 2005.

A Particle-Level Embedding Visualizations

A.1 $H \rightarrow 4q$ Embeddings

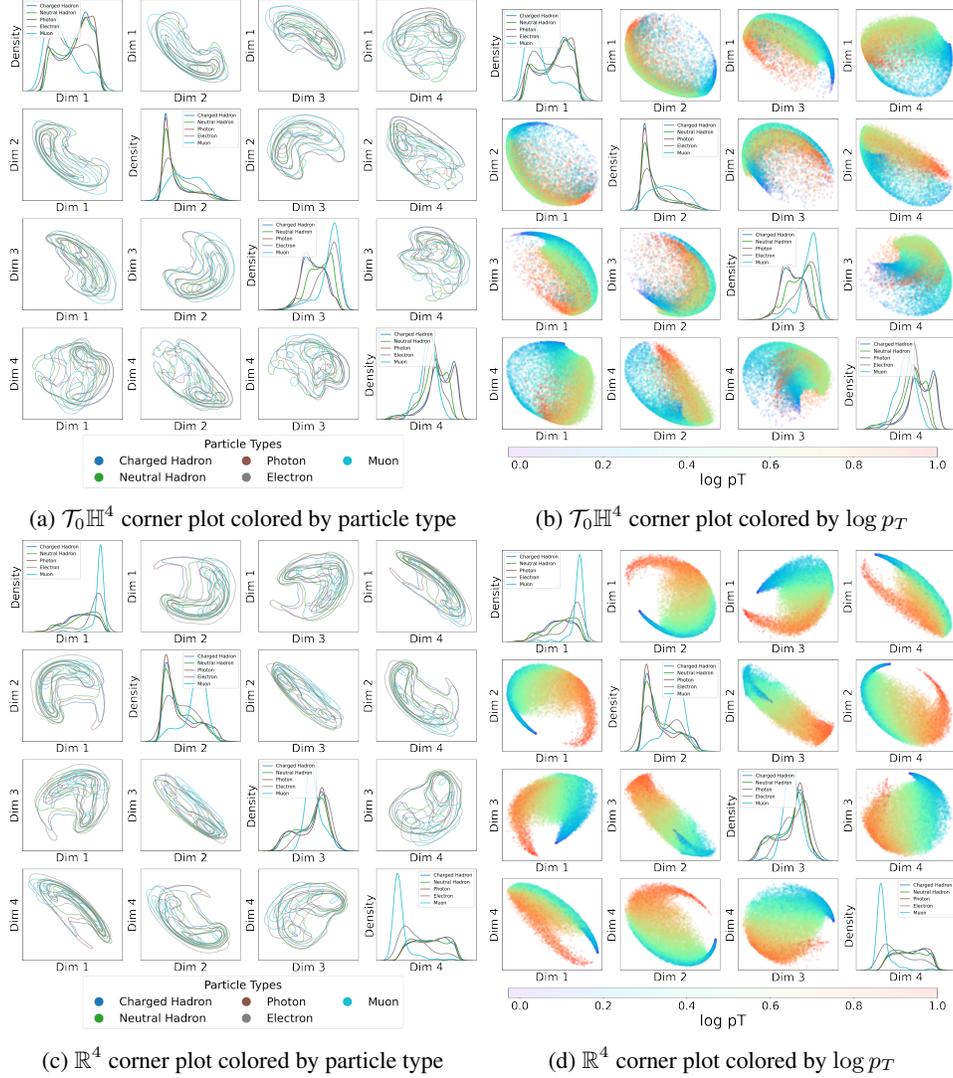


Figure 4: Corner plots for the 4D \mathcal{PM} -MLP models for the $H \rightarrow 4q$ process employing \mathbb{R}^4 and \mathbb{H}^4 geometries. For \mathbb{H}^4 , we plot results in the tangent space $\mathcal{T}_0\mathbb{H}^4$. We show corner plots for both particle-level embedding geometries for the constituents of 10k jets from the test dataset. We show the particle-level embeddings colored by both particle-type and $\log p_T$, normalized to max 1.

A.2 $t \rightarrow bqq'$ Embeddings

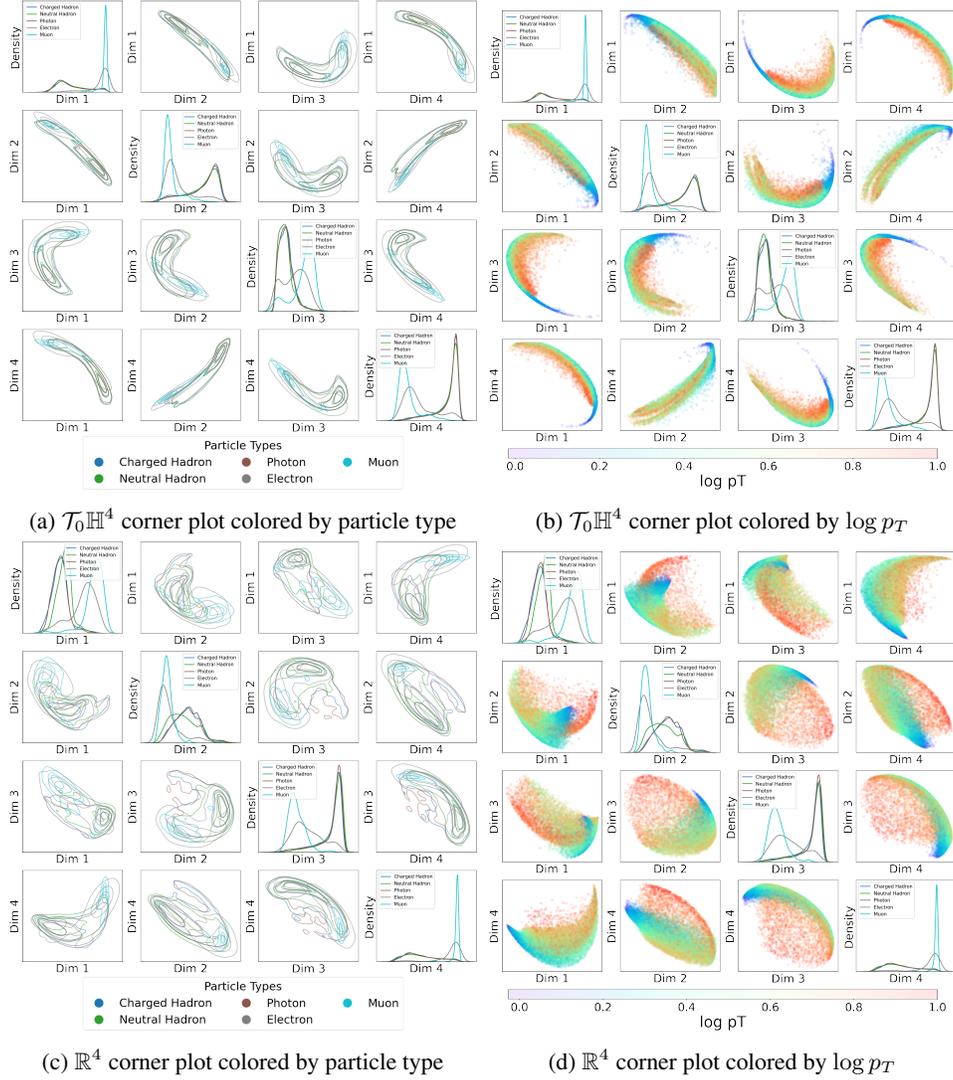


Figure 5: Corner plots for the 4D \mathcal{PM} -MLP models for the $t \rightarrow bqq'$ process employing \mathbb{R}^4 and \mathbb{H}^4 geometries. For \mathbb{H}^4 , we plot results in the tangent space $\mathcal{T}_0\mathbb{H}^4$. We show corner plots for both particle-level embedding geometries for the constituents of 10k jets from the test dataset. We show the particle-level embeddings colored by both particle-type and $\log p_T$, normalized to max 1.