# Latent Variable Modelling with Hyperbolic Normalizing Flows

Avishek Joey Bose, Ariella Smofsky, Renjie Liao, Prakash Panangaden, William L. Hamilton

ICML 2020

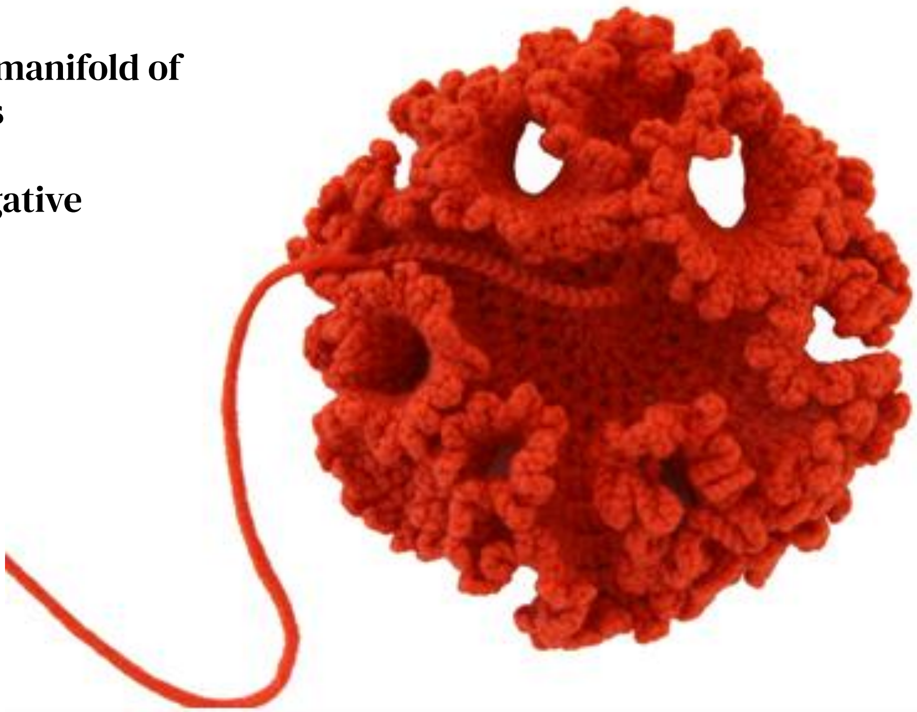
1. Why hyperbolic?
2. Related work
3. Hyperbolic normalizing flows
4. Experiments
5. Conclusion

# Why hyperbolic?

# Hyperbolic space

Riemannian manifold of
*n* dimensions

Constant negative
curvature *K*

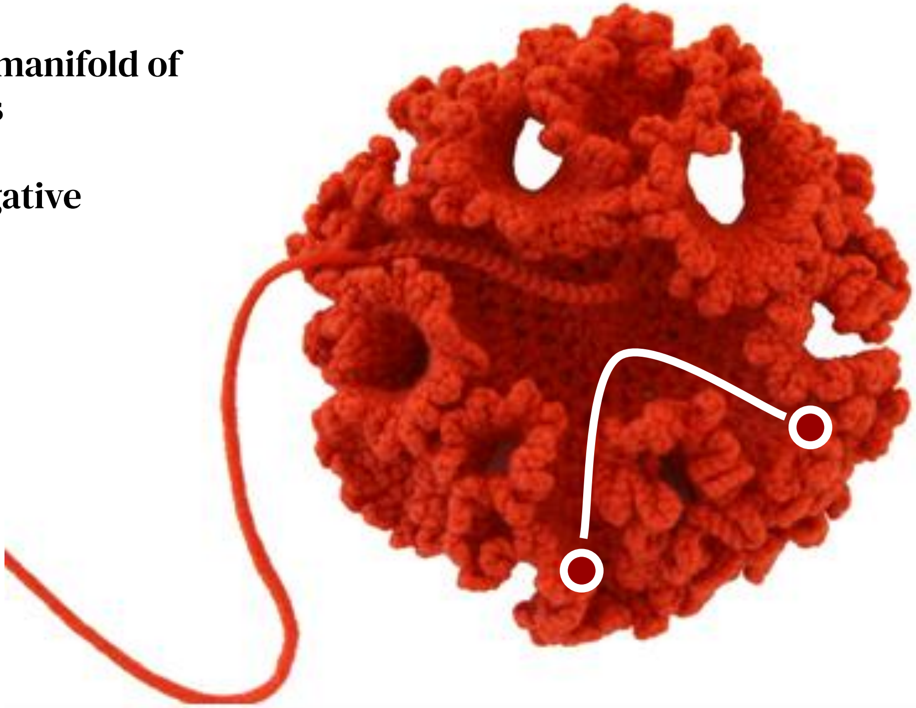


Margaret Wertheim - the Institute For Figuring

# Hyperbolic space

Riemannian manifold of
$n$ dimensions

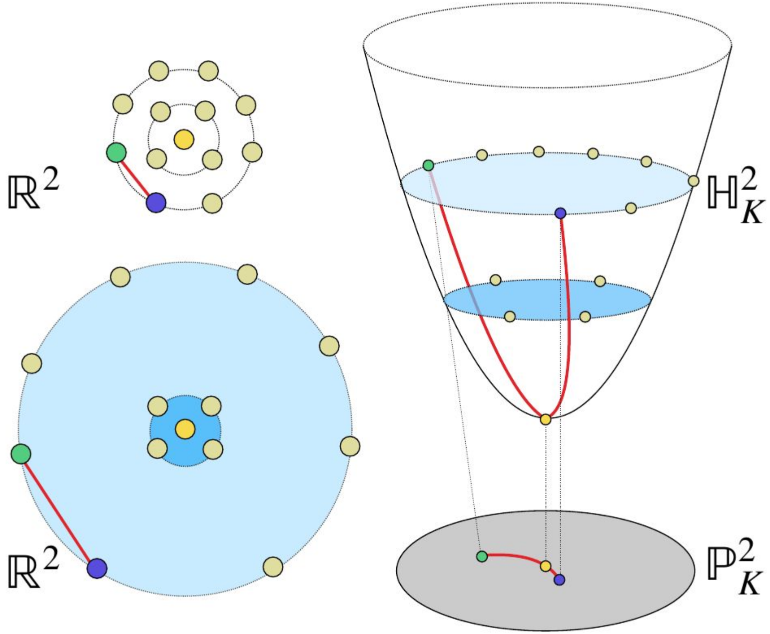
Constant negative
curvature $K$



Margaret Wertheim - the Institute For Figuring

# Hyperbolic space

**Unlike Euclidean space, geodesics go near origin**



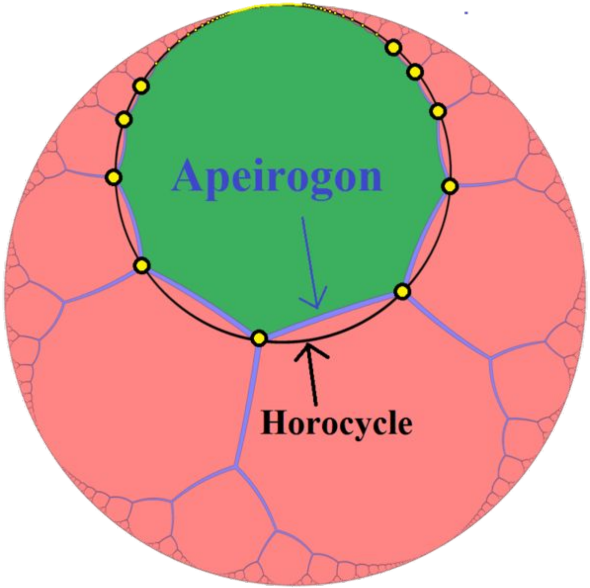

Bose et al. Latent Variable Modelling with Hyperbolic Normalizing Flows (2020)

# Hyperbolic space

Naturally models hierarchical structure:

- origin is close to everything
- the further you go, the more space there is
- non-intersecting lines are easy to draw

Good for embedding trees!




https://commons.wikimedia.org/wiki/File:Hyperbolic_apeirogon_example.png

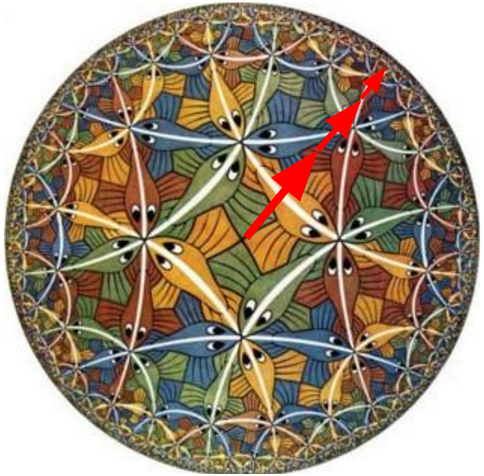
# Poincaré ball model

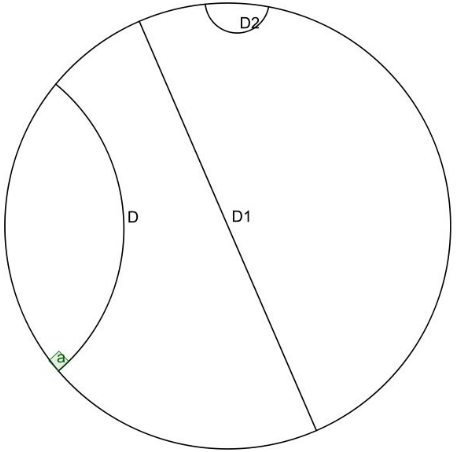
Projects manifold onto unit ball of same dimension

- boundary is at infinity
- easy to visualize
- numerically unstable

Straight lines in 2D:

- circles orthogonal to boundary
- lines through origin






Escher, M.C. *Circle Limit III* (1959)

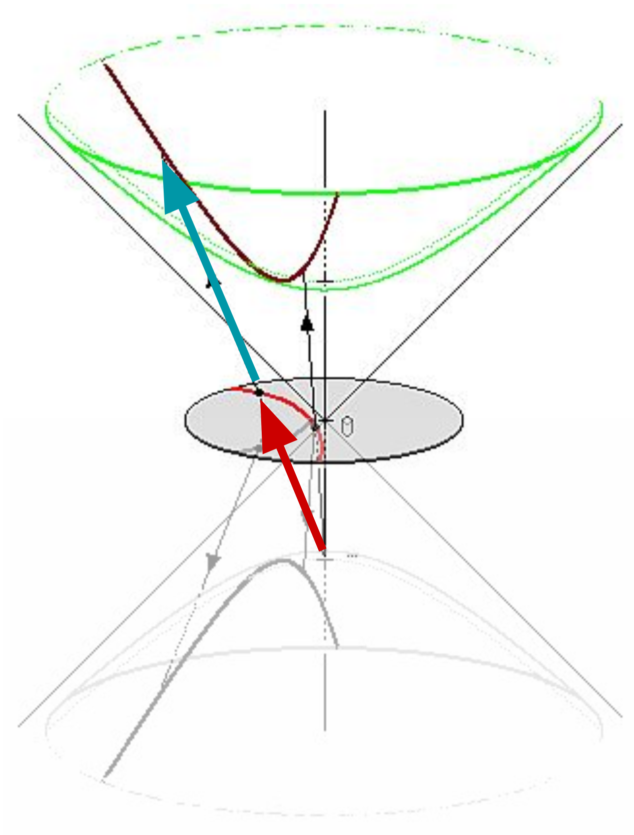
# Lorentz model

Projects onto hyperboloid in Euclidean space ($d$+1)

- a.k.a. Minkowski space
- can project to Poincaré ball at origin along a line from the center of the opposite hyperboloid

Straight lines in 2D:

- planes intersecting hyperboloid and origin



# Lorentz model

**Lorentz (Minkowski) inner product over $\mathbb{R}^{n+1}$:**

$$\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}} := -\underbrace{x_0 y_0}_{\text{“special” } 0^{\text{th}} \text{ coordinate}} + x_1 y_1 + \cdots + x_n y_n$$

# Lorentz model

**Lorentz (Minkowski) inner product over $\mathbb{R}^{n+1}$:**

$$\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}} := -x_0 y_0 + x_1 y_1 + \cdots + x_n y_n$$

**Points on the manifold satisfy:**

$$\mathbb{H}^n_K := \{x \in \mathbb{R}^{n+1} : \langle \mathbf{x}.\mathbf{x} \rangle_{\mathcal{L}} = 1/K,\ x_0 > 0,\ K < 0\}.$$

# Lorentz model

**Lorentz (Minkowski) inner product over $\mathbb{R}^{n+1}$:**

$$\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}} := -x_0 y_0 + x_1 y_1 + \cdots + x_n y_n$$

**Points on the manifold satisfy:**

$$\mathbb{H}^n_K := \{x \in \mathbb{R}^{n+1} : \langle \mathbf{x}.\mathbf{x} \rangle_{\mathcal{L}} = 1/K,\ x_0 > 0,\ K < 0\}.$$

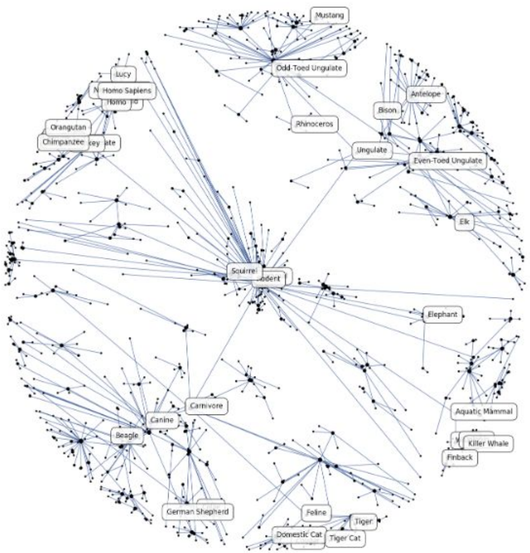
**Induced distance on the manifold:**

$$d(\mathbf{x}, \mathbf{y})_{\mathcal{L}} := \frac{1}{\sqrt{-K}} \operatorname{arccosh}(-K \langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}}).$$

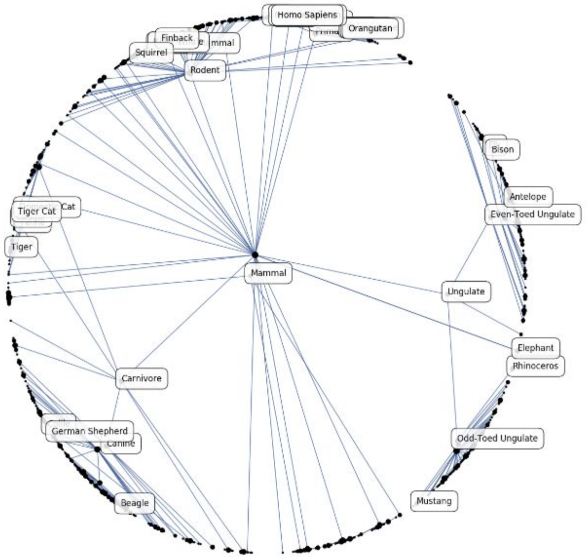
**In the tangent space of the origin, norm is identical to Euclidean norm!**

# Related work

# Hyperbolic embedding (Poincaré)



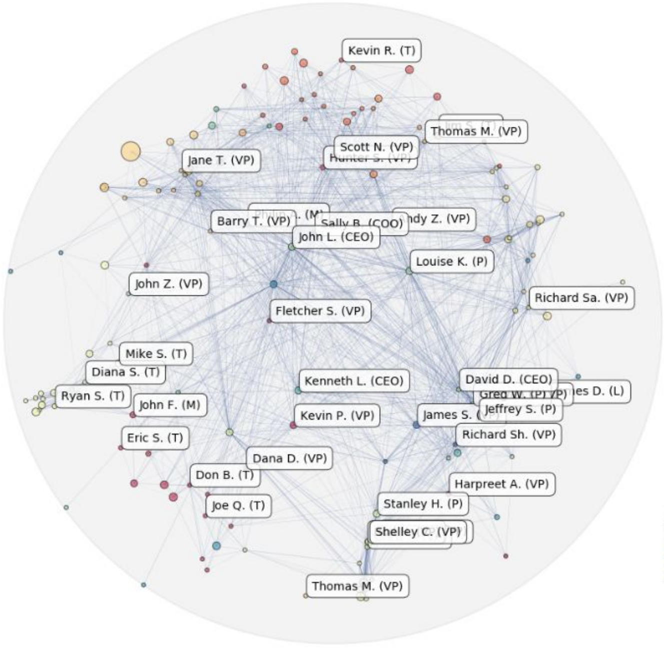

(a) Intermediate embedding after 20 epochs




(b) Embedding after convergence

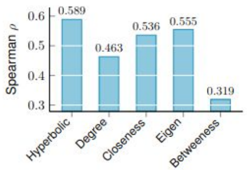
Nickel & Kiela. Poincaré embeddings for learning hierarchical representations (NeurIPS 2017)

# Hyperbolic embedding (Lorentz)




(a) Embedding of the Enron communication graph

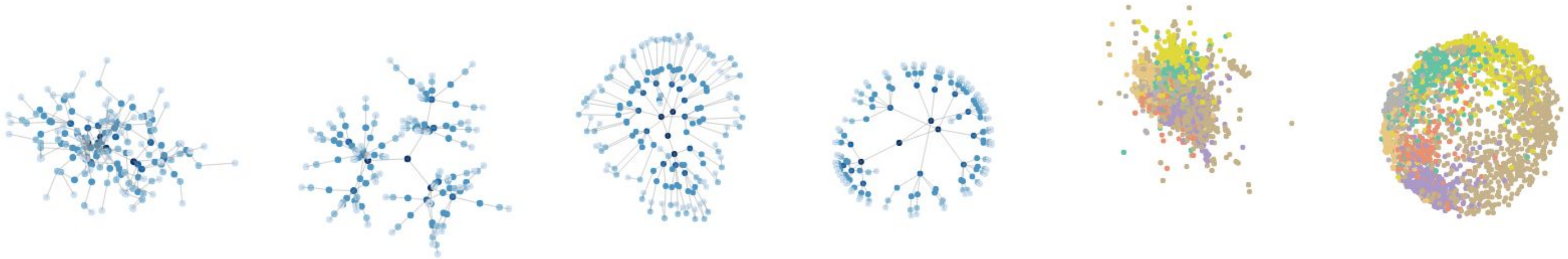

(b) Org. hierarchy




(c) Rank-order correlation

Nickel & Kiela. Learning continuous hierarchies in the lorentz model of hyperbolic geometry (ICML 2018)

# Hyperbolic graph neural networks



(a) GCN layers. (b) HGCN layers. (c) GCN (left), HGCN (right).

Chami, Ying, Ré, & Leskovec. Hyperbolic graph convolutional neural networks (NeurIPS 2019)

# Hyperbolic VAEs

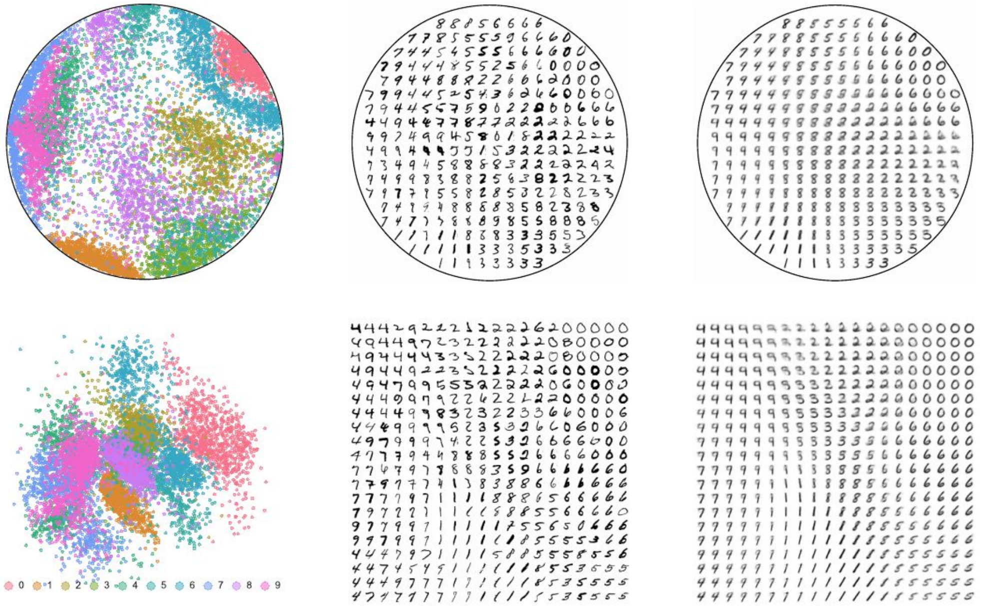



Figure 7: MNIST Posteriors mean (Left) sub-sample of digit images associated with posteriors mean (Middle) Model samples (Right) – for $\mathcal{P}^{1.4}$-VAE (Top) and $\mathcal{N}$-VAE (Bottom).

Mathieu et al. Continuous hierarchical representations with Poincaré variational auto-encoders (NeurIPS 2019)

# Normalizing flows

Use invertible transforms to turn an initial probability density into a more complex target

- Need to efficiently compute determinant of Jacobian to change density

# Normalizing flows

Use invertible transforms to turn an initial probability density into a more complex target

- Need to efficiently compute determinant of Jacobian to change density

Real non-volume-preserving (NVP) transform:

- affine transform half of input channels at a time, conditioning on other half
- triangular Jacobian (det = product of diagonal)

$$\tilde{f}^{TC}(\tilde{x}) = \begin{cases} \tilde{z}_1 & = \tilde{x}_1 \\ \tilde{z}_2 & = \tilde{x}_2 \odot \overbrace{\sigma}^{\text{non-linearity}}(\underbrace{s(\tilde{x}_1)}_{\text{scale}}) + \underbrace{t(\tilde{x}_1)}_{\text{translate}} \end{cases}$$

split input channels

# Hyperbolic Normalizing Flows

# Hyperbolic geometry

**Projection: map vector in ambient space to manifold**

$$\mathrm{proj}_{\mathbb{H}^n_K}(x) = \frac{x}{\sqrt{-K}||x||_{\mathcal{L}}}$$

**vector in $\mathbb{R}^{n+1}$**

$$\underbrace{x_0 = \sqrt{||\hat{x}||_2^2 + \frac{1}{K}}}_{}$$

**can also project from $\mathbb{R}^n$ by concatenating $0^{th}$ coordinate**

# Hyperbolic geometry

**Projection: map vector in ambient space to manifold**

$$\mathrm{proj}_{\mathbb{H}^n_K}(x) = \frac{x}{\sqrt{-K}||x||_{\mathcal{L}}}$$

$$x_0 = \sqrt{||\hat{x}||_2^2 + \frac{1}{K}}$$

**Exponential map: tangent space to manifold**

**vector in $\mathcal{T}_{\mathbf{x}}\mathbb{H}^n_K$**

$$\exp_{\mathbf{x}}^K(v) = \cosh\left(\frac{||v||_{\mathcal{L}}}{R}\right)\mathbf{x} + \sinh\left(\frac{||v||_{\mathcal{L}}}{R}\right)\frac{Rv}{||v||_{\mathcal{L}}}$$

**point in $\mathbb{H}^n_K$**

$$\underbrace{R = 1/\sqrt{-K}}_{\text{generalized radius}}$$

# Hyperbolic geometry

**Projection: map vector in ambient space to manifold**

$$\mathrm{proj}_{\mathbb{H}^n_K}(x) = \frac{x}{\sqrt{-K}||x||_{\mathcal{L}}}$$

$$x_0 = \sqrt{||\hat{x}||_2^2 + \frac{1}{K}}$$

**Exponential map: tangent space to manifold**

$$\exp_{\mathbf{x}}^K(v) = \cosh\left(\frac{||v||_{\mathcal{L}}}{R}\right)\mathbf{x} + \sinh\left(\frac{||v||_{\mathcal{L}}}{R}\right)\frac{Rv}{||v||_{\mathcal{L}}}$$

$$R = 1/\sqrt{-K}$$

**Logarithmic map: inverse of exp map (manifold to tangent space)**

$$\log_{\mathbf{x}}^K \mathbf{y} = \frac{\operatorname{arccosh}(\alpha)}{\sqrt{\alpha^2 - 1}}(\mathbf{y} - \alpha\mathbf{x})$$

$$\alpha = K\langle \mathbf{x}, \mathbf{y}\rangle_{\mathcal{L}}$$

**vector in $\mathbb{H}^n_K$**

# Hyperbolic geometry

**Projection: map vector in ambient space to manifold**

$$\text{proj}_{\mathbb{H}^n_K}(x) = \frac{x}{\sqrt{-K}||x||_{\mathcal{L}}}$$

$$x_0 = \sqrt{||\hat{x}||_2^2 + \frac{1}{K}}$$

**Exponential map: tangent space to manifold**

$$\exp_{\mathbf{x}}^K(v) = \cosh\left(\frac{||v||_{\mathcal{L}}}{R}\right)\mathbf{x} + \sinh\left(\frac{||v||_{\mathcal{L}}}{R}\right)\frac{Rv}{||v||_{\mathcal{L}}}$$

$$R = 1/\sqrt{-K}$$

**Logarithmic map: inverse of exp map (manifold to tangent space)**

$$\log_{\mathbf{x}}^K \mathbf{y} = \frac{\text{arccosh}(\alpha)}{\sqrt{\alpha^2 - 1}}(\mathbf{y} - \alpha\mathbf{x})$$

$$\alpha = K\langle \mathbf{x}, \mathbf{y}\rangle_{\mathcal{L}}$$

**Parallel transport: map from one tangent space to another**

$$\underbrace{\text{PT}_{\mathbf{x}\to\mathbf{y}}^K(v)}_{\mathcal{T}_{\mathbf{y}}\mathbb{H}^n_K} = v + \frac{\langle \mathbf{y}, v\rangle_{\mathcal{L}}}{R^2 - \langle \mathbf{x}, \mathbf{y}\rangle_{\mathcal{L}}}(\mathbf{x} + \mathbf{y})$$

$\mathcal{T}_{\mathbf{x}}\mathbb{H}^n_K$

$$(\text{PT}_{\mathbf{x}\to\mathbf{y}}^K(v))^{-1} = \text{PT}_{\mathbf{y}\to\mathbf{x}}^K(v)$$

**swap x and y for inverse**

# Hyperbolic normal distributions

**Riemannian normal: like Euclidean normal distribution, but replace norm with induced distance**

$$p(z|\bar{z},\gamma) = \frac{1}{Z(\gamma)} \exp\left( \frac{-\overbrace{d^2(z,\bar{z})}^{\textbf{distance}}}{2\gamma^2} \right)$$

# Hyperbolic normal distributions

Riemannian normal: like Euclidean normal distribution, but replace norm with induced distance

$$p(z|\bar{z},\gamma) = \frac{1}{Z(\gamma)} \exp\left( \frac{-\overbrace{d^2(z,\bar{z})}^{\text{distance}}}{2\gamma^2} \right)$$

Restricted normal: condition normal distribution in $\mathbb{R}^{n+1}$ by whether a point is on the manifold

$$p(z) = p(x \sim \underbrace{\mathcal{N}(0, I)}_{\text{ambient space } \mathbb{R}^{n+1}} \mid \underbrace{\langle \mathbf{x}, \mathbf{x} \rangle_{\mathcal{L}} = 1/K}_{\text{Lorentzian metric condition for } \mathbb{H}^n_K})$$

# Hyperbolic normal distributions

**Riemannian normal:** like Euclidean normal distribution, but replace norm with induced distance

$$p(z|\bar{z},\gamma) = \frac{1}{Z(\gamma)} \exp\left(\frac{-\overbrace{d^2(z,\bar{z})}^{\text{distance}}}{2\gamma^2}\right)$$

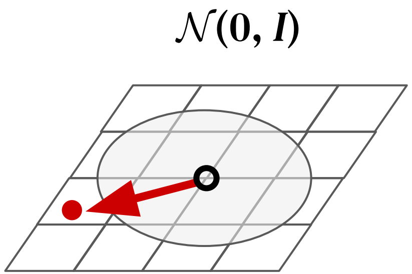
**Restricted normal:** condition normal distribution in $\mathbb{R}^{n+1}$ by whether a point is on the manifold

$$p(z) = p(x \sim \underbrace{\mathcal{N}(0, I)}_{\text{ambient space } \mathbb{R}^{n+1}} \mid \underbrace{\langle \mathrm{x}, \mathrm{x}\rangle_{\mathcal{L}} = 1/K}_{\text{Lorentzian metric condition for } \mathbb{H}^n_K})$$

**Wrapped normal:** reparameterize standard normal from tangent space at origin

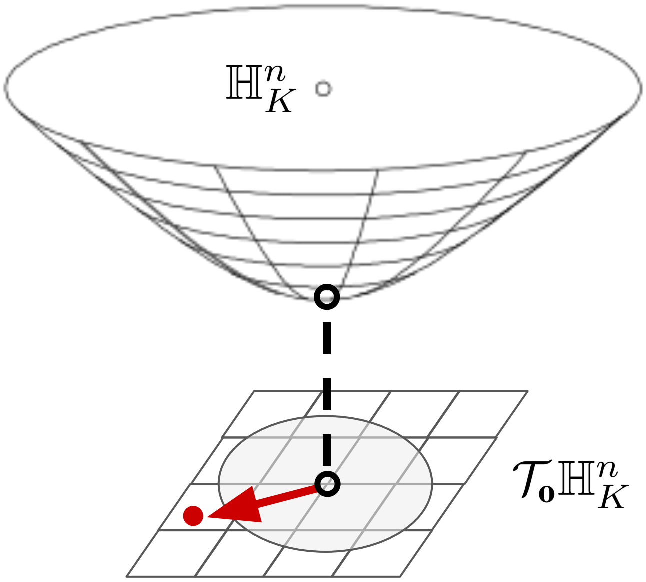
# Wrapped normal

1. Sample from $\mathcal{N}(0, I)$



# Wrapped normal

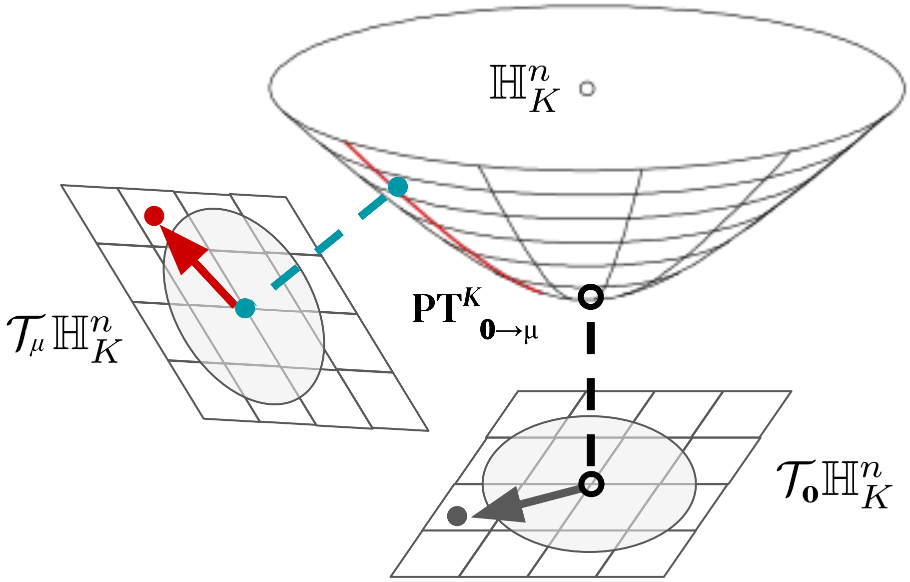
1. Sample from $\mathcal{N}(0, I)$
2. Put in tangent space at origin (concatenate 0 at 0th coordinate)




https://commons.wikimedia.org/wiki/File:Hyperboloid-2s.svg

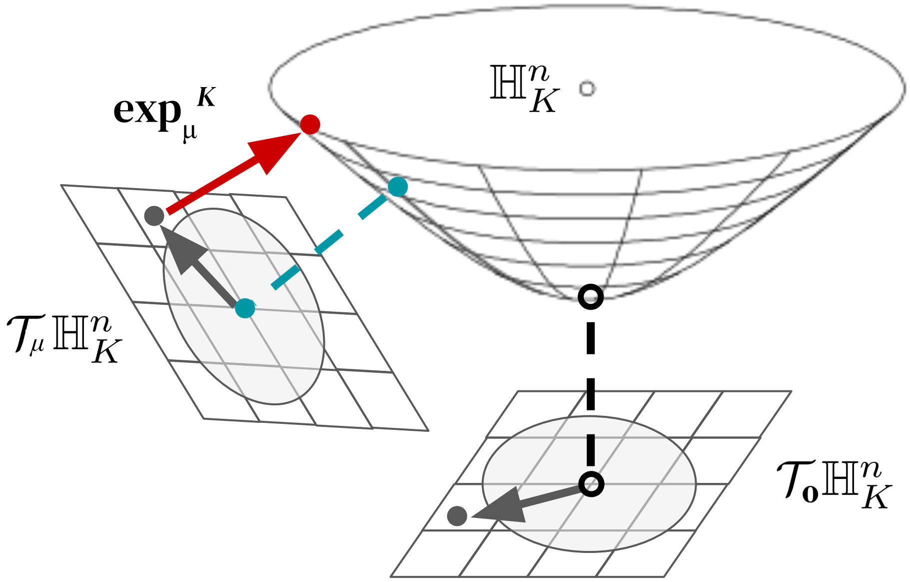
# Wrapped normal

1. Sample from $\mathcal{N}(0, I)$
2. Put in tangent space at origin (concatenate 0 at 0th coordinate)
3. Parallel transport to tangent space of another point



# Wrapped normal

1. Sample from $\mathcal{N}(0, I)$
2. Put in tangent space at origin (concatenate 0 at 0th coordinate)
3. Parallel transport to tangent space of another point
4. Map to manifold



# Wrapped normal

**Density is given by change of variable:**

$$\log p(\mathbf{z}) = \log p(v) - (n-1)\log\left(\frac{\sinh\left(\|u\|_{\mathcal{L}}\right)}{\|u\|_{\mathcal{L}}}\right)$$

# Hyperbolic normalizing flows

**Base distribution is wrapped normal**

**Use RealNVP transform**

$$\tilde{f}^{\mathcal{T}C}(\tilde{x}) = \begin{cases} \tilde{z}_1 & = \tilde{x}_1 \\ \tilde{z}_2 & = \tilde{x}_2 \odot \sigma(s(\tilde{x}_1)) + t(\tilde{x_1}) \end{cases}$$

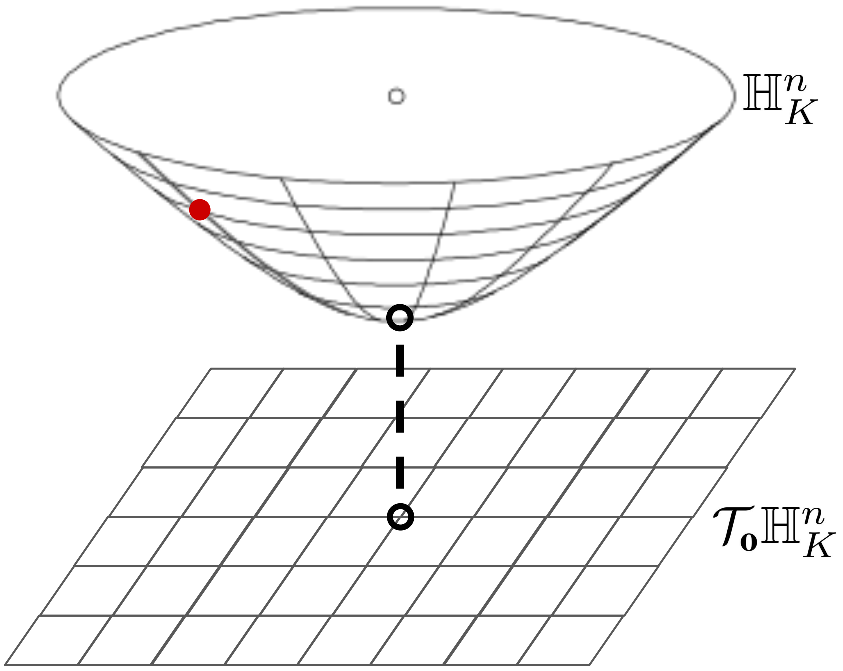
**Two ways to make this hyperbolic:**

1. **Tangent coupling (simple, fast)**
2. **Wrapped hyperboloid coupling (not tied to origin, better results)**

# Tangent Coupling

For each layer:

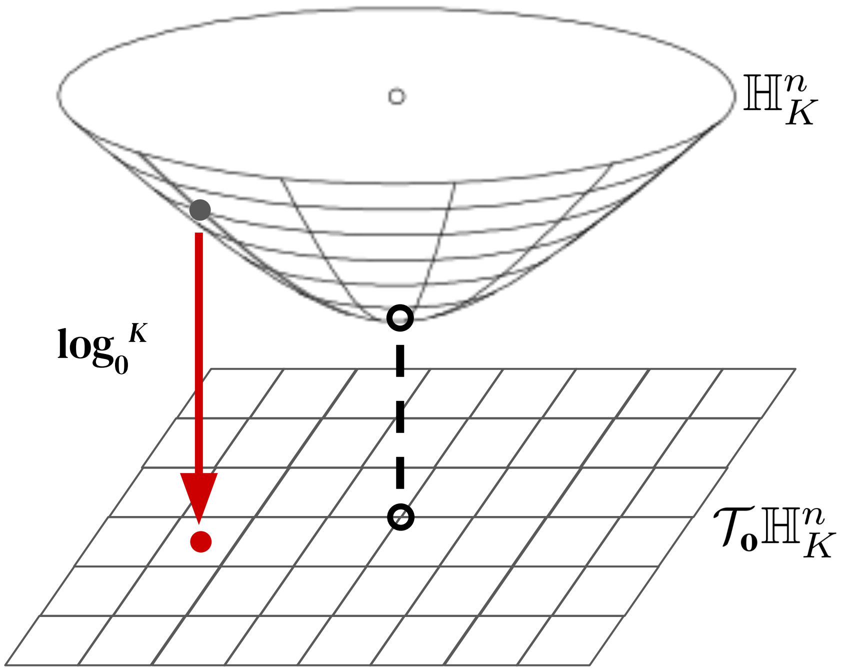
1. Sample point in manifold



# Tangent Coupling

For each layer:

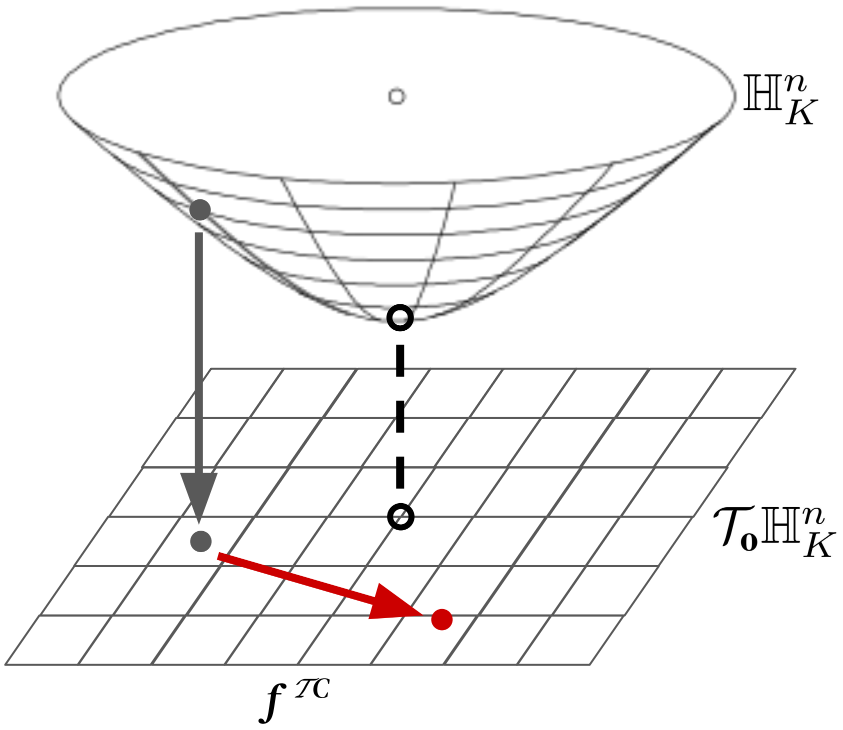
1. Sample point in manifold
2. Map to tangent space at origin



# Tangent Coupling

For each layer:

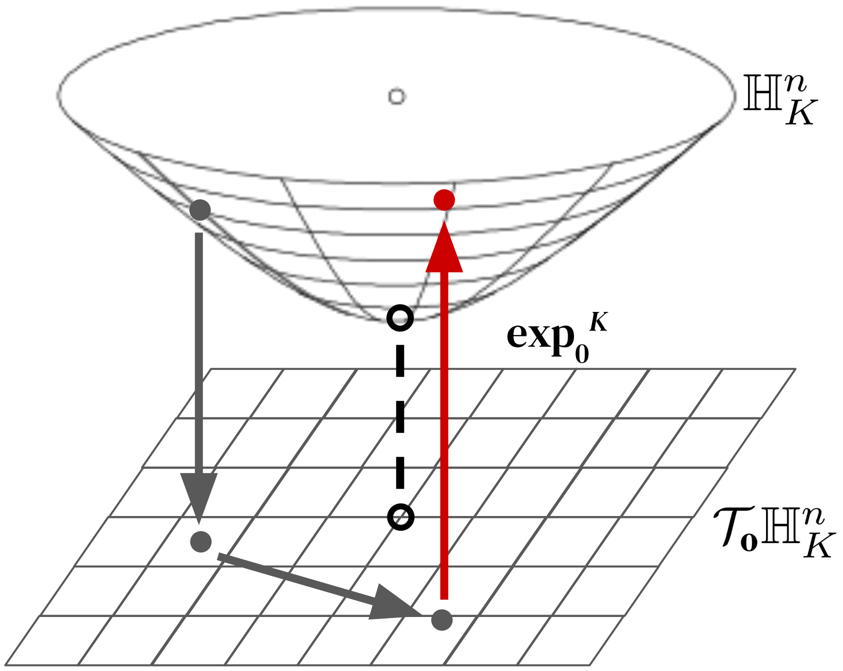
1. Sample point in manifold
2. Map to tangent space at origin
3. Apply Euclidean flow transform



# Tangent Coupling

For each layer:

1. Sample point in manifold
2. Map to tangent space at origin
3. Apply Euclidean flow transform
4. Map back to manifold



# Jacobian of $\mathcal{TC}$ layer

$$\tilde{f}^{\mathcal{TC}}(\tilde{x}) = \begin{cases} \tilde{z}_1 & = \tilde{x}_1 \\ \tilde{z}_2 & = \tilde{x}_2 \odot \sigma(s(\tilde{x}_1)) + t(\tilde{x}_1) \end{cases}$$

$$f^{\mathcal{TC}}(\mathbf{x}) = \exp_{\mathbf{o}}^{K}(\tilde{f}^{\mathcal{TC}}(\log_{\mathbf{o}}^{K}(\mathbf{x}))),$$

$$\left|\det\left(\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right)\right| = \left(\frac{R\sinh(\frac{||z||_{\mathcal{L}}}{R})}{||z||_{\mathcal{L}}}\right)^{n-1} \times \prod_{i=d+1}^{n} \sigma(s(\tilde{x}_1))_i \times \left(\frac{R\sinh(\frac{||\log_{\boldsymbol{o}}^{K}(\boldsymbol{x})||_{\mathcal{L}}}{R})}{||\log_{\boldsymbol{o}}^{K}(\boldsymbol{x})||_{\mathcal{L}}}\right)^{1-n}$$

**Exp map** **RealNVP transform** **Log map**

# Wrapped Hyperboloid Coupling

1. Sample point in manifold
2. Map to tangent space at origin
3. Apply $\mathcal{W}\mathbb{H}C$ flow transform
4. Map back to manifold

same scaling

$$\tilde{f}^{\mathcal{W}\mathbb{H}C}(\tilde{x}) = \begin{cases} \tilde{z}_1 &= \tilde{x}_1 \\ \tilde{z}_2 &= \log_{\mathbf{o}}^{K}\left(\exp_{t(\tilde{x}_1)}^{K}\left(\mathrm{PT}_{\mathbf{o}\to t(\tilde{x}_1)}(v)\right)\right) \end{cases}$$

$$v = \tilde{x}_2 \odot \sigma(s(\tilde{x}_1))$$

$$f^{\mathcal{W}\mathbb{H}C}(\mathbf{x}) = \underbrace{\exp_{\mathbf{o}}^{K}\left(\tilde{f}^{\mathcal{W}\mathbb{H}C}(\log_{\mathbf{o}}^{K}(\mathbf{x}))\right)}_{\text{same as } \mathcal{T}C}. \qquad (13)$$

# Wrapped Hyperboloid Coupling

1. Sample point in manifold
2. Map to tangent space at origin
3. Apply $\mathcal{W}\mathbb{H}C$ flow transform
4. Map back to manifold

$\mathcal{W}\mathbb{H}C$ transform:

1. Split inputs
2. Scale by non-linear factor
3. Parallel transport to new tangent space (instead of translating)
4. Map to manifold
5. Map back to tangent space at origin

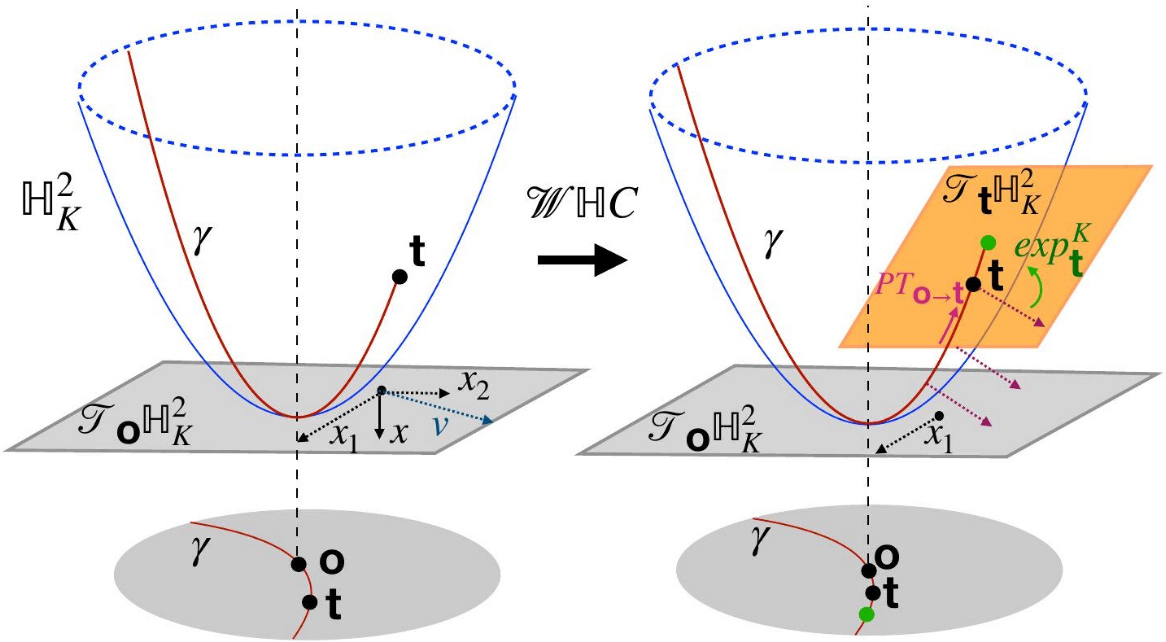
$$\tilde{f}^{\mathcal{W}\mathbb{H}C}(\tilde{x}) = \begin{cases} \tilde{z}_1 &= \tilde{x}_1 \\ \tilde{z}_2 &= \overbrace{\log_{\mathbf{o}}^{K}}^{\text{return to } \mathcal{T}_{\mathbf{o}}\mathbb{H}_K^n}\left(\exp_{t(\tilde{x}_1)}^{K}\left(\mathrm{PT}_{\mathbf{o}\to t(\tilde{x}_1)}(v)\right)\right) \end{cases}$$

$$v = \tilde{x}_2 \odot \sigma(s(\tilde{x}_1))$$

$$f^{\mathcal{W}\mathbb{H}C}(\mathbf{x}) = \exp_{\mathbf{o}}^{K}(\tilde{f}^{\mathcal{W}\mathbb{H}C}(\log_{\mathbf{o}}^{K}(\mathbf{x}))). \qquad (13)$$

"translation"

# Wrapped Hyperboloid Coupling




Bose et al. Latent Variable Modelling with Hyperbolic Normalizing Flows (2020)

# Wrapped Hyperboloid Coupling

**To guarantee $x_2$ does not affect $x_1$:**

- **find the parallel transport target $\boldsymbol{t}$ by explicitly setting components from $x_1$ to 0**

$$\boldsymbol{t} = [t_0, 0, \ldots, 0, t_{d+1}, \ldots, t_n]$$

- **find $t_0$ using the trick to project from $\mathbb{R}^n$ to the manifold:**

$$\boldsymbol{t}_0 = \sqrt{||\boldsymbol{t}||_2^2 + \frac{1}{K}}$$

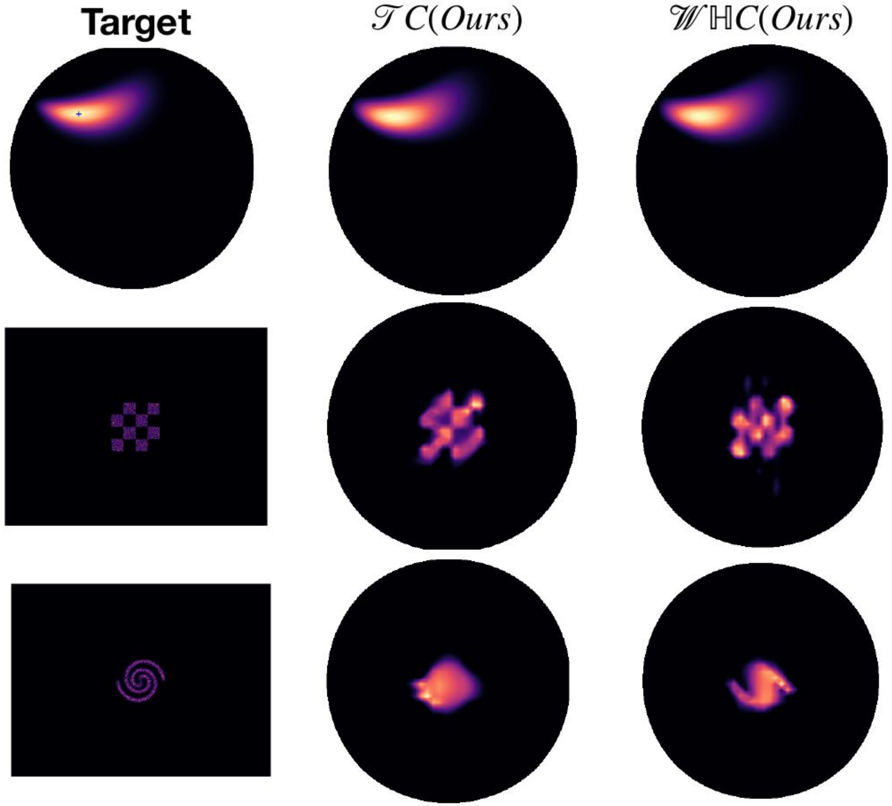
# Jacobian of $\mathcal{W}\mathbb{H}\mathcal{C}$

$$
\begin{aligned}
\left|\det\left(\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right)\right| = {} & \overbrace{\prod_{i=d+1}^{n} \sigma(s(\tilde{x}_1))_i}^{\text{RealNVP transform}} \times \overbrace{\left(\frac{R\sinh(\frac{||q||_{\mathcal{L}}}{R})}{||q||_{\mathcal{L}}}\right)^{l}}^{\text{Four maps}} \\
& \times \underbrace{\left(\frac{R\sinh(\frac{||\log_{\boldsymbol{o}}^{K}(\hat{\boldsymbol{q}})||_{\mathcal{L}}}{R})}{||\log_{\boldsymbol{o}}^{K}(q)||_{\mathcal{L}}}\right)^{-l}} \times \underbrace{\left(\frac{R\sinh(\frac{||\tilde{z}||_{\mathcal{L}}}{R})}{||\tilde{z}||_{\mathcal{L}}}\right)^{n-1}} \\
& \times \underbrace{\left(\frac{R\sinh(\frac{||\log_{\boldsymbol{o}}^{K}(\boldsymbol{x})||_{\mathcal{L}}}{R})}{||\log_{\boldsymbol{o}}^{K}(\boldsymbol{x})||_{\mathcal{L}}}\right)^{1-n}}, \qquad (15)
\end{aligned}
$$

*where* $\tilde{z} = \text{concat}(\tilde{z}_1, \tilde{z}_2)$, *the constant* $l = n - d$, $\sigma$ *is a non-linearity,* $q = \text{PT}_{\boldsymbol{o}\to t(\tilde{x}_1)}(v)$ *and* $\hat{\boldsymbol{q}} = \exp_{\boldsymbol{t}}^{K}(q)$.

# Experiments

# Density estimation




Bose et al. Latent Variable Modelling with Hyperbolic Normalizing Flows (2020)

# Density estimation

**Branching diffusion process (BDP)**

- **sample from a "binary decision tree"**

**Dynamically binarized MNIST**

- **randomly threshold pixels to {0, 1}**

**Estimate likelihood of test data with importance sampling**

| Model | BDP-2 | BDP-4 | BDP-6 |
|---|---|---|---|
| $\mathcal{N}$-VAE | $-55.4_{\pm 0.2}$ | $-55.2_{\pm 0.3}$ | $-56.1_{\pm 0.2}$ |
| $\mathbb{H}$-VAE | $-$**54.9**$_{\pm 0.3}$ | $-55.4_{\pm 0.2}$ | $-58.0_{\pm 0.2}$ |
| $\mathcal{NC}$ | $-55.4_{\pm 0.4}$ | **-54.7**$_{\pm 0.1}$ | **-55.2**$_{\pm 0.3}$ |
| $\mathcal{TC}$ | **-54.9**$_{\pm 0.1}$ | $-55.4_{\pm 0.1}$ | $-57.5_{\pm 0.2}$ |
| $\mathcal{W}\mathbb{H}\mathcal{C}$ | **-55.1**$_{\pm 0.4}$ | $-55.2_{\pm 0.2}$ | $-56.9_{\pm 0.4}$ |

*Table 1.* Test Log Likelihood on Binary Diffusion Process versus latent dimension. All normalizing flows use 2-coupling layers.

| Model | MNIST 2 | MNIST 4 | MNIST 6 |
|---|---|---|---|
| $\mathcal{N}$-VAE | $-139.5_{\pm 1.0}$ | $-115.6_{\pm 0.2}$ | $-100.0_{\pm 0.02}$ |
| $\mathbb{H}$-VAE | $*$ | $-113.7_{\pm 0.9}$ | $-99.8_{\pm 0.2}$ |
| $\mathcal{NC}$ | $-139.2_{\pm 0.4}$ | $-115.2_{\pm 0.6}$ | **-98.7**$_{0.3}$ |
| $\mathcal{TC}$ | $*$ | **-112.5**$_{\pm 0.2}$ | $-99.3_{\pm 0.2}$ |
| $\mathcal{W}\mathbb{H}\mathcal{C}$ | **-136.5**$_{\pm 2.1}$ | **-112.8**$_{\pm 0.5}$ | $-99.4_{\pm 0.2}$ |

*Table 2.* Test Log Likelihood on MNIST averaged over 5 runs verus latent dimension. * indicates numerically unstable settings.

Bose et al. Latent Variable Modelling with Hyperbolic Normalizing Flows (2020)

# Graph reconstruction

1. Disorders and disease genes: linked by known disorder-gene associations
2. SIR disease spreading model: nodes are individuals with varying susceptibility to disease

Nodes, adjacency matrix, node feature matrix:

$$\mathcal{G} = (V, A, X)$$

Encode nodes with variational graph autoencoder (VGAE):

$$q_\phi(Z \mid A, X)$$

Replace decoder with simple inner product:

$$p(A_{u,v} = 1 \mid z_u, z_v) = \underbrace{\sigma(z_u{}^T z_v)}_{\text{inner product in hyperbolic space}}$$

# Graph reconstruction

| Model | Dis-I AUC | Dis-I AP | Dis-II AUC | Dis-II AP |
|---|---|---|---|---|
| $\mathcal{N}$-VAE | $0.90_{\pm 0.01}$ | $0.92_{\pm 0.01}$ | $0.92_{\pm 0.01}$ | $0.91_{\pm 0.01}$ |
| $\mathbb{H}$-VAE | $0.91_{\pm 5e\text{-}3}$ | $0.92_{\pm 5e\text{-}3}$ | $0.92_{\pm 4e\text{-}3}$ | $0.91_{\pm 0.01}$ |
| $\mathcal{N}C$ | $0.92_{\pm 0.01}$ | $0.93_{\pm 0.01}$ | $0.95_{\pm 4e\text{-}3}$ | $0.93_{\pm 0.01}$ |
| $\mathcal{T}C$ | $\mathbf{0.93}_{\pm 0.01}$ | $0.93_{\pm 0.01}$ | $\mathbf{0.96}_{\pm 0.01}$ | $0.95_{\pm 0.01}$ |
| $\mathcal{W}\mathbb{H}C$ | $\mathbf{0.93}_{\pm 0.01}$ | $\mathbf{0.94}_{\pm 0.01}$ | $\mathbf{0.96}_{\pm 0.01}$ | $\mathbf{0.96}_{\pm 0.01}$ |

*Table 3.* Test AUC and Test AP on Graph Embeddings where Dis-I has latent dimesion 6 and Dis-II has latent dimension 2.

Bose et al. Latent Variable Modelling with Hyperbolic Normalizing Flows (2020)

# Graph generation

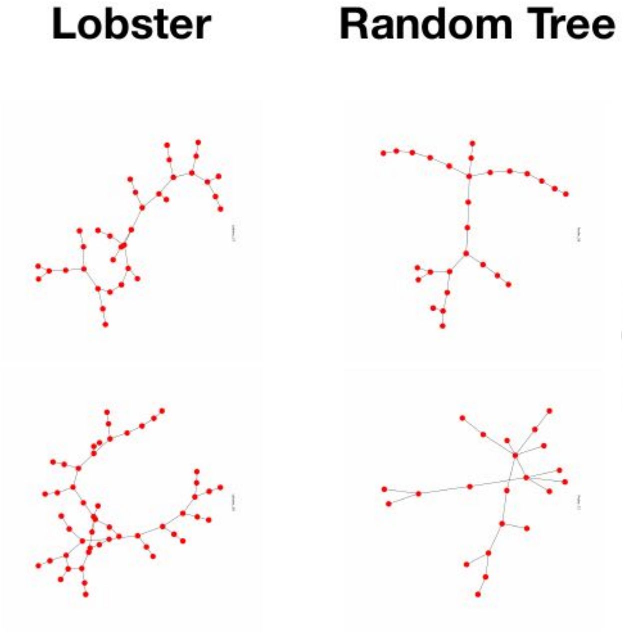
Pretrain graph autoencoder on trees

Decode edge probabilities with:

$$p(A_{u,v} = 1 \mid z_u, z_v) = \sigma((-\overbrace{d_{\mathcal{G}}(u, v)}^{\text{distance between nodes in latent space}} - b)/\tau)$$

sigmoid function

learned bias, temperature



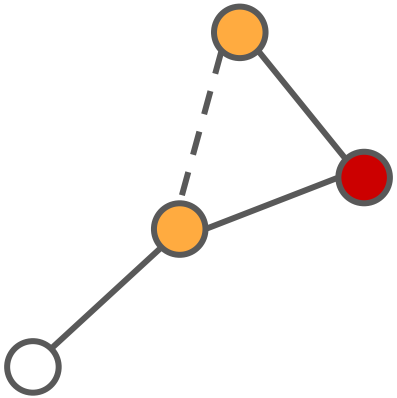

Bose et al. Latent Variable Modelling with Hyperbolic Normalizing Flows (2020)

# Evaluating generated graphs

Hard to evaluate graphs by likelihood, due to multiple isomorphic orderings

Compare statistics using maximum mean discrepancy (Wasserstein/earth mover's distance):

- degree (neighbours per node)
- local clustering coefficient ("triangles to neighbours" per node)
- global clustering coefficient (total triangles to triplets)
- spectrum (eigenvalues of graph Laplacian)
- accuracy (valid trees)



# Evaluating generated graphs

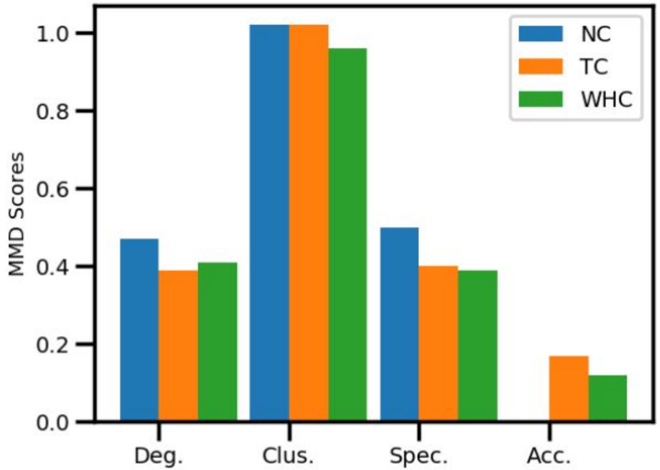
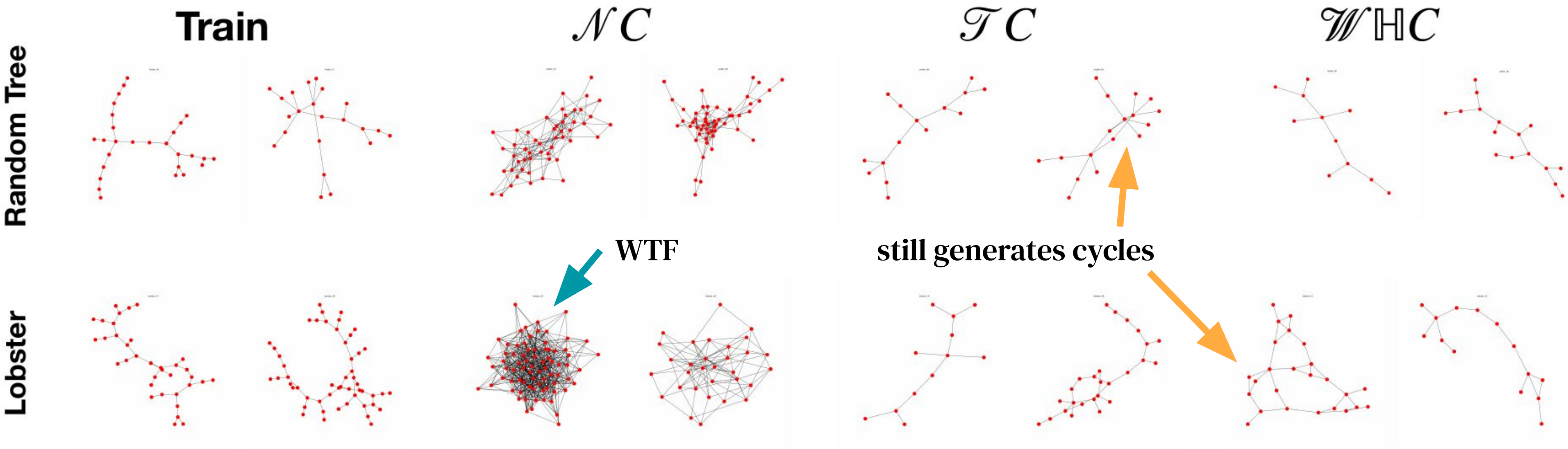



*Figure 5.* MMD scores for graph generation on Lobster graphs. Note, that $\mathcal{N}C$ achieves $0\%$ accuracy.

Bose et al. Latent Variable Modelling with Hyperbolic Normalizing Flows (2020)

# Graph generation results




Bose et al. Latent Variable Modelling with Hyperbolic Normalizing Flows (2020)

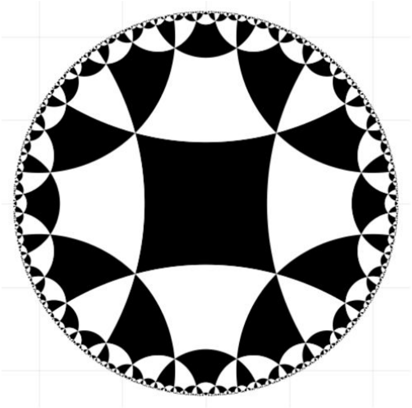
# Conclusion

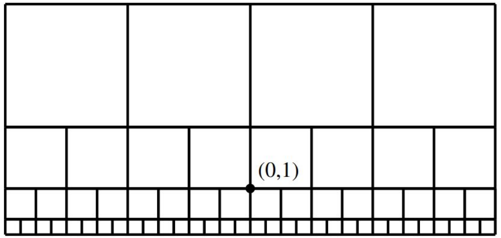
# Advantages and disadvantages

+ Good at generating/reconstructing trees
- Less effective with more latent dimensions
- Clamping for numerical stability limits depth(?)

# Further thoughts

- What problems benefit from high-dimensional hyperbolic spaces?
- How does performance and cost scale with increased model depth?
- Can tiling methods fix numerical instability?
- Hyperbolic diffusion when?






Yu & De Sa. Numerically accurate hyperbolic embeddings using tiling-based models (NeurIPS 2019)

# Thanks!