# *Visualizing model behavior*

Visualizing a high-dimensional model manifold must be done by torturing it into two or three dimensions. If the model manifold is a hyperribbon, this can be done elegantly by aligning the important long ‘stiff’ axes properly and projecting the points on the manifold onto them.

# *Visualizing least-squares model manifolds*

Least-squares models have a behavior space in $\mathbb{R}^M$. We can view the model manifold using tools of linear algebra:

- SVD: Singular value decomposition
- PCA: Principal component analysis
- MDS: Multidimensional scaling

For nonlinear least-squares models, both the parameter space and the behavior space are linear vector spaces, and we can use linear algebra methods to rotate the manifold optimally for projection.

# Eigenvalues and matrix factorization

For symmetric and Hermetian matrices (used in physics), we use eigenvalues and eigenvectors. These can be viewed as a *matrix factorization*

$$M = V^T \Lambda V = \begin{pmatrix} \hat{e}_1 \\ \hat{e}_2 \\ \cdots \\ \hat{e}_N \end{pmatrix} \begin{pmatrix} \lambda_1 & 0 & \cdots \\ 0 & \lambda_2 & \cdots \\ \cdots & & \\ \cdots & \cdots & \lambda_N \end{pmatrix} \begin{pmatrix} \hat{e}_1, & \hat{e}_2, & \cdots & \hat{e}_N \end{pmatrix}.$$

So $\lambda_n$ squeezes the orthonormal direction $\hat{e}_n$. This can be written $M_{\alpha\beta} = V_{\delta\alpha}\Lambda_{\delta\delta'}V_{\delta'\beta}$, where $V^TV = V_{\delta\alpha}V_{\delta\beta} = \delta_{\alpha\beta}$ is a rotation that aligns the axes to the squeezed directions.

Physicists are used to symmetric and Hermitian matrices, for which eigenvalues and their eigenvectors characterize their behavior. We can view them as a way to factorize a matrix into three components, rotating it to align with the stretched directions. Matrix factorization is a common and flexible method in machine learning. If our Jacobian were square and symmetric, we could just use the coefficients along the three largest eigenvalues as coordinates for a 3D plot of the model manifold.

# Decomposing the metric tensor part 1

We can use this to decompose our metric tensor $g_{\alpha\beta} = J^T J = J_{i\alpha} J_{i\beta}$. Remember $J_{i\alpha}$ is the MxN Jacobian that takes small displacements in parameters into displacements in behavior (i.e., in the tangent space to the model manifold):

$$g = J^T J = V^T \Lambda V$$

Here the columns of V form an orthonormal basis for parameter space that are aligned with the stiff and sloppy eigendirections.

We have been studying the stiff and sloppy directions in parameter space. This introduces an eigenvector matrix decomposition of the metric, with V rotating the axes to align with the stiff directions.

# Decomposing the metric tensor part 2

What about $JJ^T = J_{i\alpha}J_{j\alpha}$, the MxM matrix in behavior space? It gives us an orthonormal decomposition

$$JJ^T = U^T \Lambda U = \begin{pmatrix} \hat{f}_1 \\ \hat{f}_2 \\ \cdots \\ \hat{f}_M \end{pmatrix} \begin{pmatrix} \lambda_1 & 0 & \cdots & & 0 \\ 0 & \lambda_2 & \cdots & & 0 \\ \cdots & & & & \\ 0 & \cdots & \lambda_N & \cdots & 0 \\ 0 & \cdots & 0 & \cdots & 0 \\ \cdots & & & & \\ 0 & \cdots & 0 & \cdots & 0 \end{pmatrix} \begin{pmatrix} \hat{f}_1, & \hat{f}_2, & \cdots & \hat{f}_M \end{pmatrix}$$

U has columns $\hat{f}_m$ along the tangent of the m[th] most important axis of the hyperribbon. For monomials it is the m[th] longest axis of the hyperellipsoid.

Decomposing J JT gives us the basis in behavior space that sits along the tangent of the model manifold, giving the directions that are stretched and squeezed by the model y_theta.

# Singular value decomposition

For any matrix J (non-symmetric, non-square, …) we can find its *singular value decomposition*.
U and V are as before, and the MxN matrix $\Sigma$ has the square roots of the eigenvalues $\sigma_\alpha = \sqrt{\lambda_\alpha}$ along the diagonal:

$$J = U\Sigma V^T$$

$$J^T J = g = V\Sigma \cancel{U^T U} \Sigma V^T = V\Sigma^2 V^T = V\Lambda V^T$$

$$JJ^T = U\Sigma^2 U^T = U\Lambda U^T$$

$$J_{i\alpha} = U_{ij}\Sigma_{j\beta}V_{\alpha\beta}$$

The singular values $\sigma_\alpha$ give the amount that axis is squeezed by the mapping J.

It should be no surprise that we can derive this from a matrix product decomposition of our Jacobian J. It squeezes e_alpha by sigma_alpha and rotates it into f_alpha. If the singular values sigma_alpha span many orders of magnitude, this yields a sloppy model in parameter space with a hyperribbon in data space.

# Principal Component Analysis

Singular value decomposition is widely used to analyze data. Suppose we have a MxN data matrix with N columns $\mathbf{E}_\alpha$ describing experiments yielding M behaviors: $D = \left(\mathbf{E}_1, \ldots, \mathbf{E}_\mathbf{N}\right)$. We presume the behaviors are centered, so that the vectors $\mathbf{E}_\alpha$ sum to zero. (This is the only difference from SVD.) Then $D = U\Sigma V^T = U_{ij}\Sigma_{j\beta}V_{\alpha\beta}$ , …, and $V\Sigma$ gives the coordinates along the principal components. Plotting the first three coordinates (with the largest singular values $\sigma_1, \sigma_2, \sigma_3$) allows one to view the distribution of results in behavior space.

Principal component analysis takes data from experiments and uses SVD to rotate it so as to visualize the most important three axes. It is extensively used in big data applications, in systems biology and many other fields. We also use it to visualize our model manifolds, with data given by sampling many predictions throughout parameter space.

PCA is a standard tool for pulling out the directions of largest variation in high-dimensional data sets. We argue that it is so often useful because of sloppiness — only a few long directions need to be visualized in a flat hyperribbon for least-squares models fit to data.

# Multidimensional Scaling

MDS is PCA when you don't have vectors, but do have a 'squared distance'.

The singular vectors $V$ in PCA are also the eigenvectors of $M = D^T D$, so $M_{\alpha\beta} = \mathbf{E}_\alpha \cdot \mathbf{E}_\beta$. The matrix of squared distances $\widetilde{M}_{\alpha\beta} = -(1/2)(\mathbf{E}_\alpha - \mathbf{E}_\beta)^2$ has the same dot products, but with some extra terms that become zero for centered data (subtracting the mean of experiments).

MDS defines coordinates $V\Sigma$ for a matrix $\widetilde{M}_{\alpha\beta} = -(1/2)\, d(\mathbf{E}_\alpha, \mathbf{E}_\beta)$, using its eigenvectors and the square roots of its eigenvalues, for a general distance measure (after 'centering' in a sneaky way).

The negative eigenvalues give pure imaginary distances, which turn out to be quite useful…

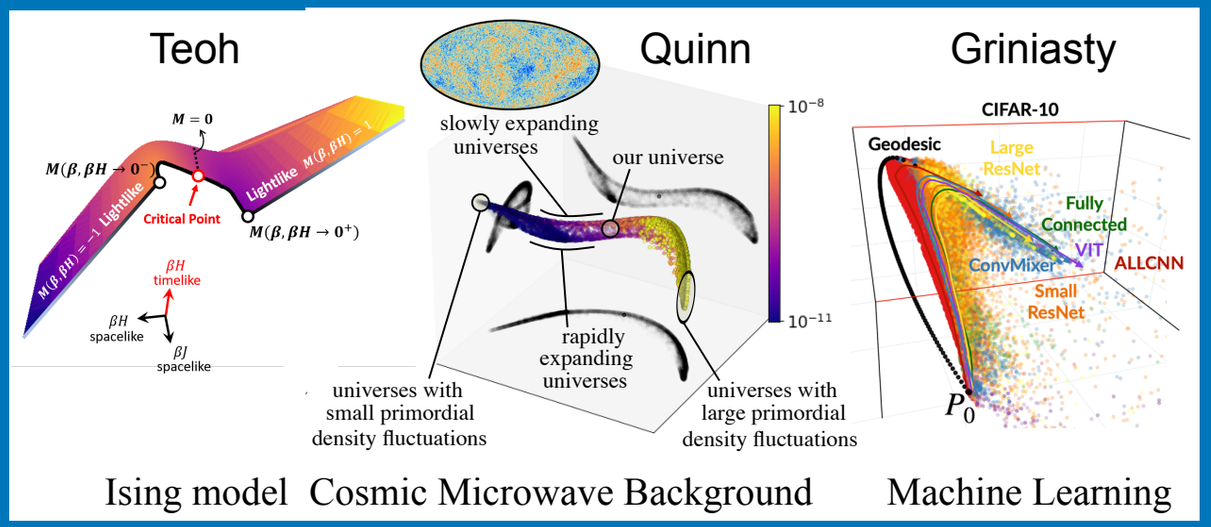
Multidimensional scaling is used when you can define a squared distance between the results of every pair of experiments, but the experiments can't be expressed in terms of a vector of measurements. By using the matrix of squared distances, the square roots of its eigenvalues are involved in the MDS coordinates. But when some of the eigenvalues are negative, this leads to some time-like coordinate axes (with pure imaginary coordinates, with negative contributions to the squared distance). This leads us to an isometric embedding in Minkowski space.

# *Bypassing the curse of dimensionality: Visualizing probabilistic models*

Let us now turn to visualizing probabilistic models. Here we shall discover a ‘curse of dimensionality’ — a common problem in big data and machine learning. We shall find two ways of dodging this curse.

# Visualizing the Model Manifold

Katherine Quinn, Itay Griniasty, Han Kheng Teoh




## Escaping the curse of dimensionality: Probabilistic models

$g_{\mu\nu}$ = Fisher Information Metric
Isometric embedding defined by model
Balances local and global

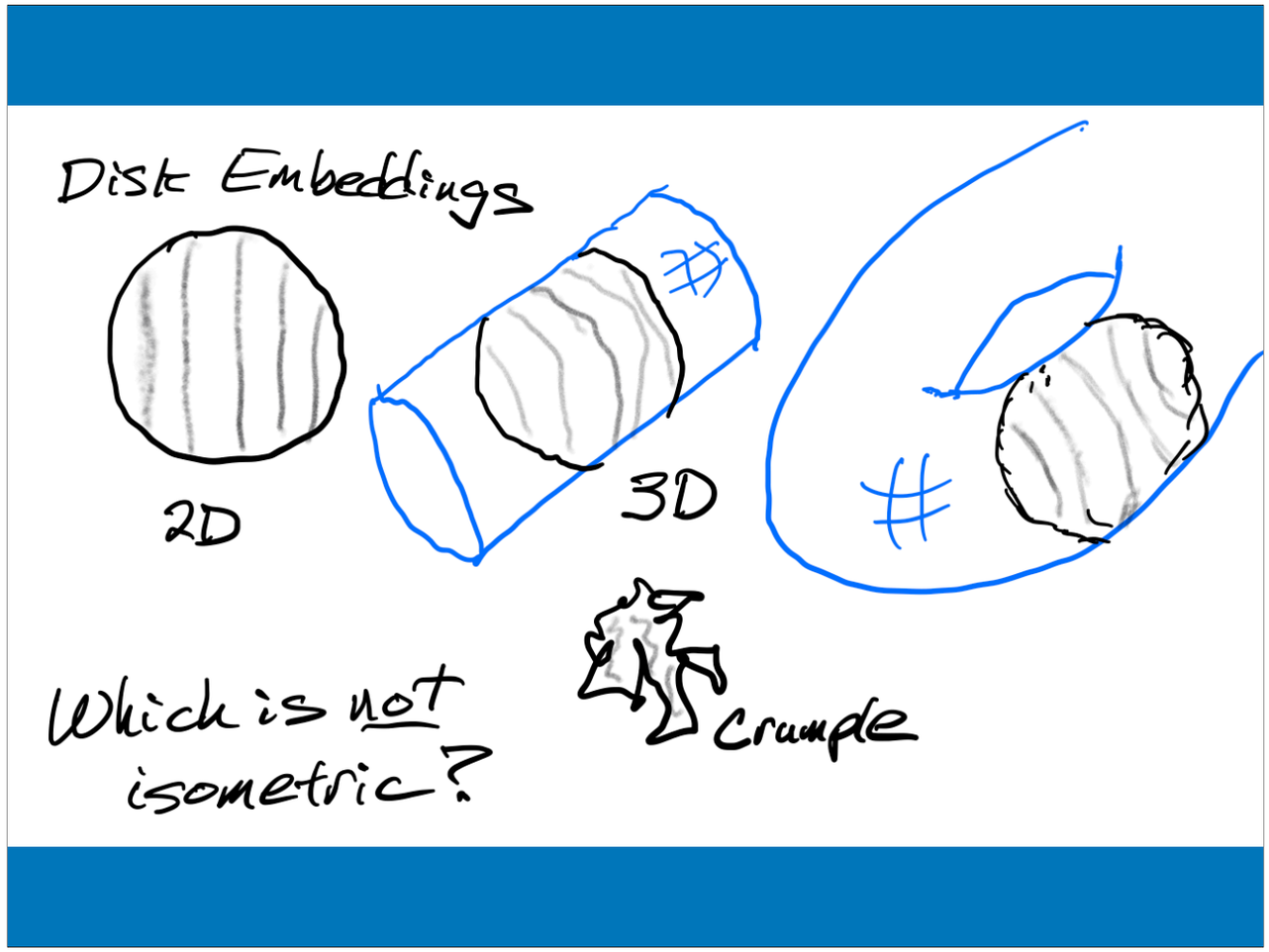
Now we turn to visualizing model manifolds for more general, probabilistic models.

But what about models which predict not data points, but probability distributions? We have discovered a family of ‘intensive’ embeddings — a method which finds a low-dimensional isometric embedding respecting the natural distance in the space of probability distributions (the Fisher Information metric).
Here we see how they can represent the predictions of a large Ising model, the ΛCDM model for the cosmic microwave background radiation, and the learning paths for deep neural networks.
We need to embed them in a space with a Minkowski-like metric to avoid the curse of dimensionality.

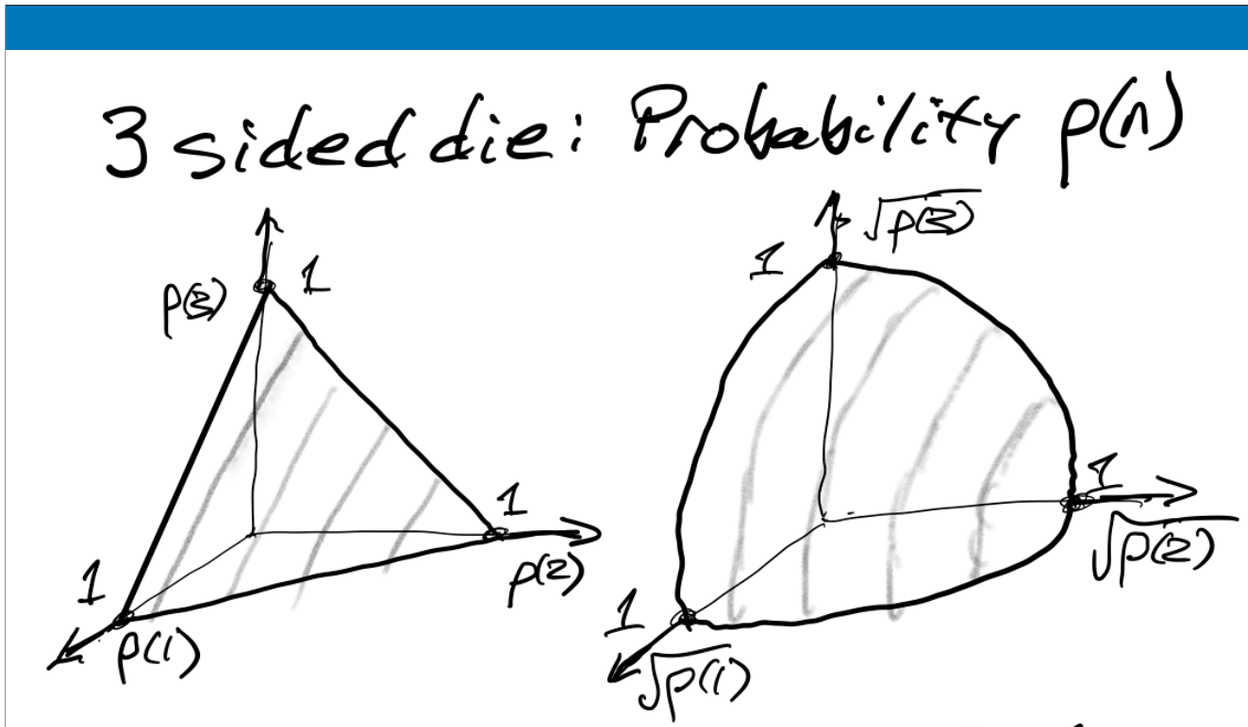
## *What is the shape of probability space?*

Our NLLS model manifolds are embedded in a Euclidean space, with each prediction given its own axis, and a metric which is a sum of squares (weighted by the error bars on the measurements, or on the predictions of the model.

Our probabilistic models sweep out a surface in a space of probability distributions. We are told its metric is the FIM. But what is its shape?




An isometric embedding preserves local distances. A disk can be wrapped around a cylinder, or crumpled, and still be isometric. But it can't be mapped in 3D onto a torus, because the torus is positively curved on the outside, and negatively curved in the doughnut hole.

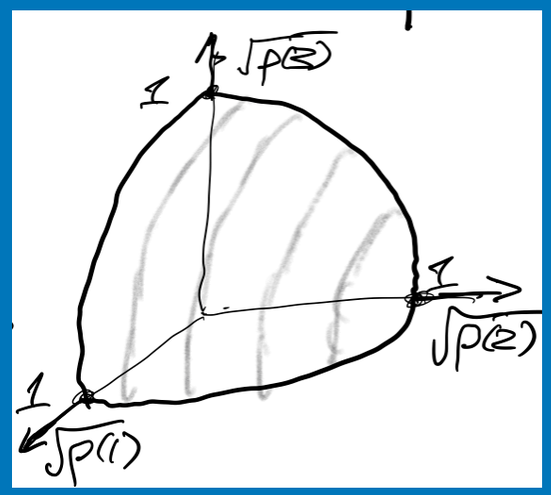



It turns out that the triangle of possible probabilities rho(3) = 1 - rho(1) - rho(2) does not have the correct FIM metric for probability distributions. Instead, treating the element-wise square root as a vector on the positive octant of the unit sphere works as an isometric embedding of the entire 3 sided die.

# Hellinger distance

The resulting distance on this sphere is due to Hellinger.

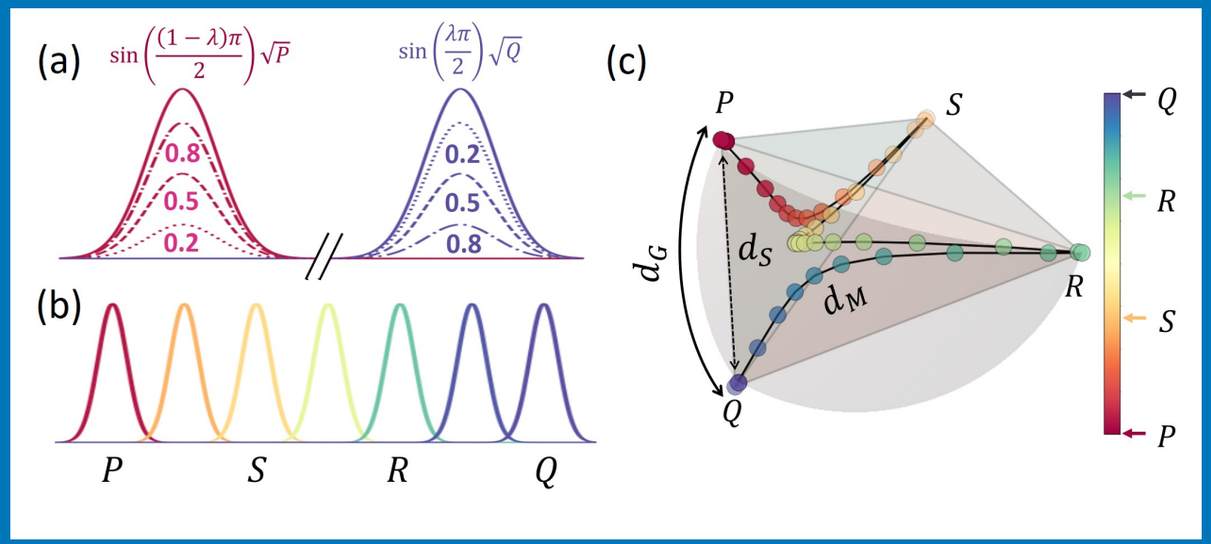
$$d^2_{\text{Hellinger}}(\rho_1, \rho_2) = \sum_{n=1}^{3} (\sqrt{\rho_1(n)} - \sqrt{\rho_2(n)})^2$$
$$= 2 - 2\sum_{n=1}^{3} \sqrt{\rho_1(n)}\sqrt{\rho_2(n)}$$
$$= 2 - 2\sqrt{\rho_1(n)} \cdot \sqrt{\rho_2(n)}$$




Any probabilistic model can be viewed as a submanifold on this sphere, and we could use PCA to draw low-dimensional projections of it. This fails spectacularly.

So, and N-spin Ising model manifold is a 2D surface lying on the sphere in 2^N dimensions. Why can't we use PCA to view it?

# Curse of dimensionality




Machine learning and big data are plagued by the ‘curse of dimensionality’. In our work, we faced this in trying to visualize probability distributions for large systems. We want our distance measure to be useful in describing probability distributions as they separate far away: we want P and Q to be three times farther apart than P and S. But in the space of probability distributions, all non-overlapping probability distributions are the same squared distance apart on the unit sphere, 2 - sqrt{P} . sqrt{Q) = 2. The geodesic is shown in (a). What happens is that the path (PSRQ) on the Hellinger sphere bends into a new dimension each time a new Gaussian becomes orthogonal to all the others, forming a hypertetrahedron.

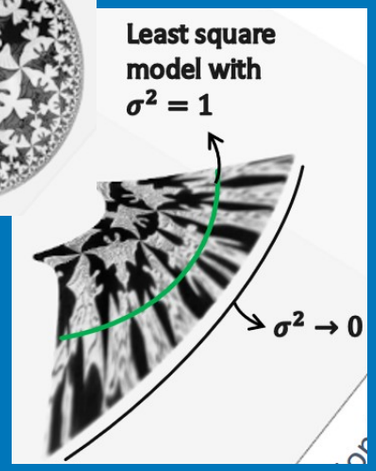
# Must crumple to fit into $\mathbb{R}^3$

Daina Taimina

Euclidean space

Minkowski space

Han Kheng Teoh: isKLe




A Gaussian fit with mean $\mu$ and rms $\sigma$ has a hyperbolic space as a model manifold: a surface with constant negative curvature.

So, adding a width to our Gaussians gives us a model with two parameters, the mean mu and standard deviation sigma. Just as the line of constant-mean Gaussians parameterized by mu must crumple up into many dimensions to fit onto the unit sphere, 2D hyperbolic space, although it can be squeezed into three dimensions, must get very wiggly and crumpled. We bypass this by embedding it into a (flat) Minkowski space. The Esher painting is not isometric, but — if the vertical axis displacement squared is counted with a minus sign, Han Kheng's isKLe embedding is.

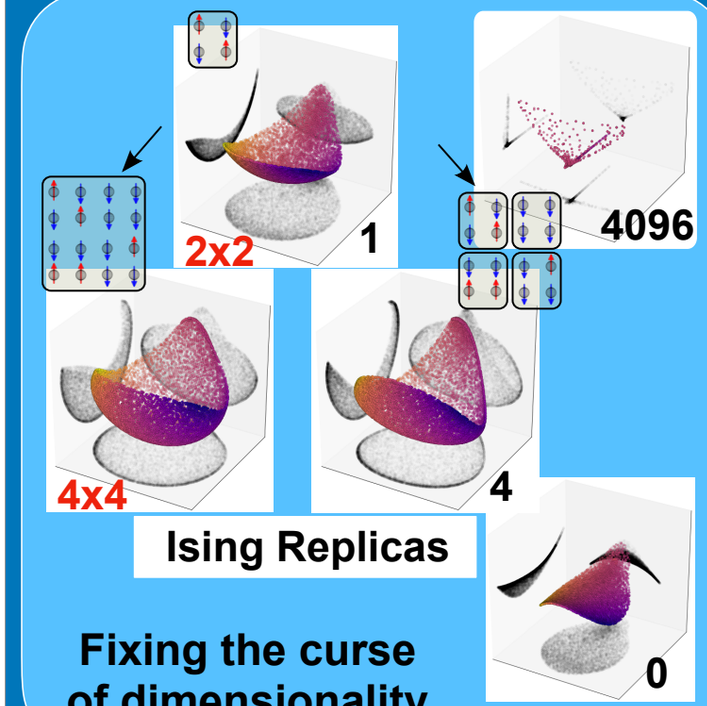
# Friday:
# Katherine Quinn: Replicas & inPCA
# Griniasty: Visualizing learning trajectories for deep neural networks using inPCA

# *inPCA: Taking the limit of zero data*

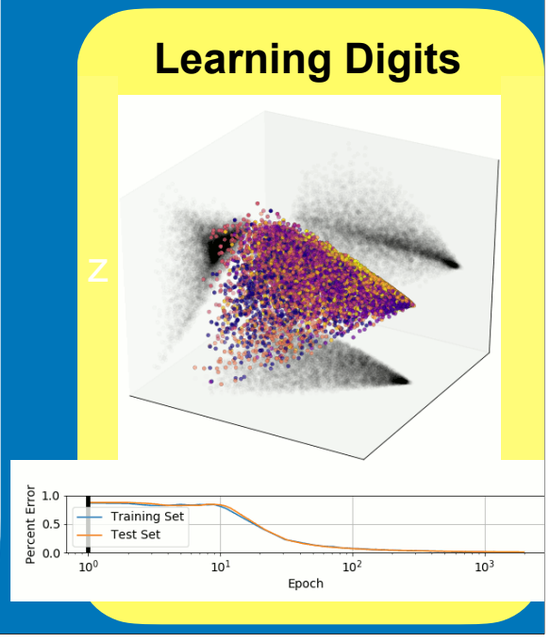
Machine learning and big data are plagued by the ‘curse of dimensionality’. In our work, we faced this in trying to visualize probability distributions for large systems.

# InPCA Intensive embeddings

Katherine Quinn, Colin Clement




Replica theory: limit of zero data
Fixes orthogonality catastrophe
Minkowski space.




The challenge we faced is to find a low-dimensional representation.
Big data applications routinely face a curse of dimensionality — most pairs of points are orthogonal, and often equally distant from one another. Big Ising models face the same problem. Once the parameters differ by enough to make their magnetizations and correlations clearly distinguishable, the geodesic distance in probability space saturates.
We see this in going from a 2x2 Ising model to a 4x4 Ising model using a traditional ‘Hellinger’ embedding. Going to a 128x128 Ising model shifts most pairs of points into orthogonal directions — projected on the upper right by the dark clump at the center. Here we mimic the 128x128 Ising model as 64^2 samples 2x2 Ising models.
We want a metric that is intensive — we want the limit of zero replicas of the 2x2 Ising model. This leads directly to our InPCA method.

# inPCA as limit of zero data

## Baby replica theory

Consider the Hellinger distance for an Ising model

$$d^2_{\text{Hellinger}}(\rho_1, \rho_2) = 2 - 2\sqrt{\rho_1} \cdot \sqrt{\rho_2} = -2(\sqrt{\rho_1} \cdot \sqrt{\rho_2} - 1)$$

The dot product goes to zero when one sample $\mathbf{s}$ is enough to tell $\rho_1$ from $\rho_2$. Consider $n$ samples of the Ising model:

$$\rho^{[n]}(\mathbf{s}_1, \ldots, \mathbf{s}_n) = \Pi_i \rho(\mathbf{s}_i).$$

The distance per replica is now

$$(1/n)(d^{[n]}_{\text{Hellinger}})^2(\rho_1, \rho_2) = -(2/n)(\sqrt{\rho_1^{[n]}} \cdot \sqrt{\rho_2^{[n]}} - 1) = -2((\sqrt{\rho_1} \cdot \sqrt{\rho_2})^n - 1)/n$$

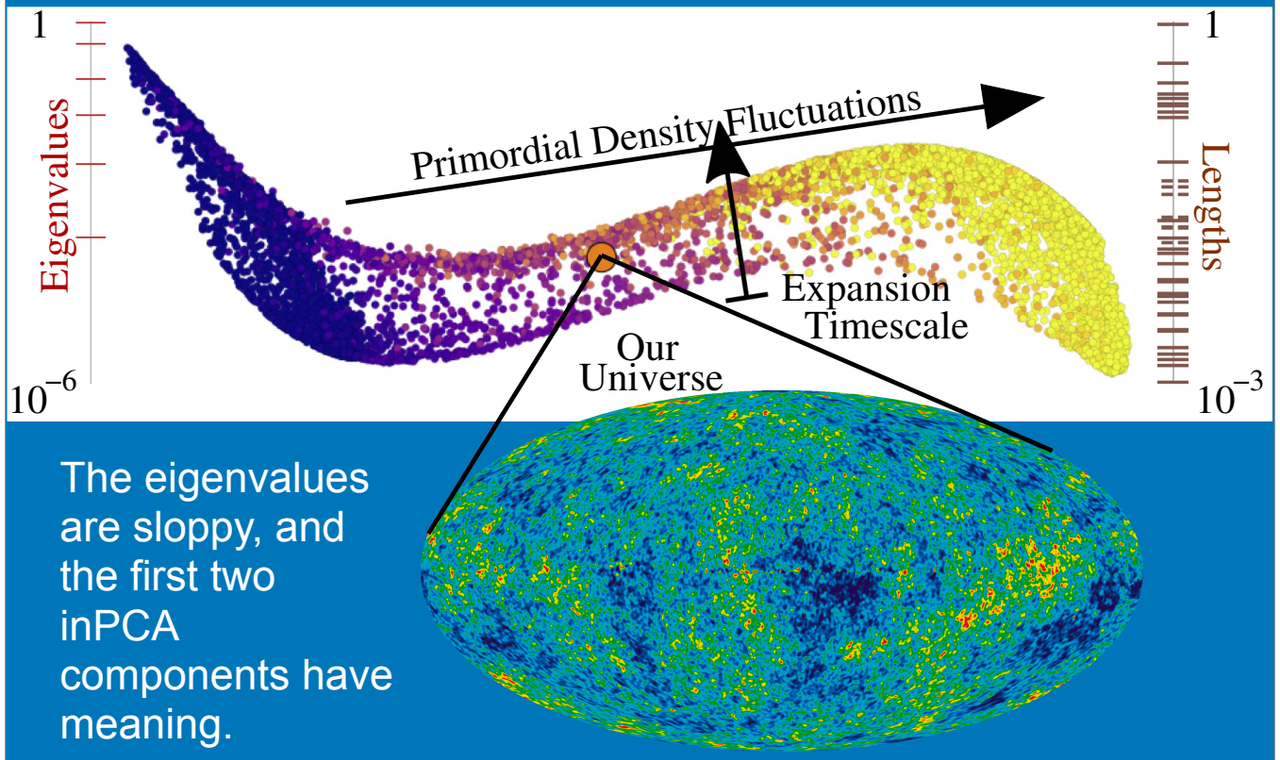
The replica trick tells us $\lim_{n->0} (x^n - 1)/n = \log(x)$, yielding a zero-replica divergence between the two distributions originally due to Bhattacharyya, $d^2_{\text{Bhatt}}(\rho_1, \rho_2) = -2\log(\sqrt{\rho_1} \cdot \sqrt{\rho_2})$.

The same limit applied to PCA rediscovers the MDS technique.

The dot product of two distributions after n measurements of the system is the nth power of the dot product after one measurement. Choosing to consider the squared distance per measurement, we use the replica trick to find that the squared distance is now minus two times the log of the dot product — the Bhattacharyya divergence. We also rediscover the multidimensional scaling method for using any distance measure to draw pictures.

# Manifold of possible Universes

Katherine Quinn, Mike Niemack

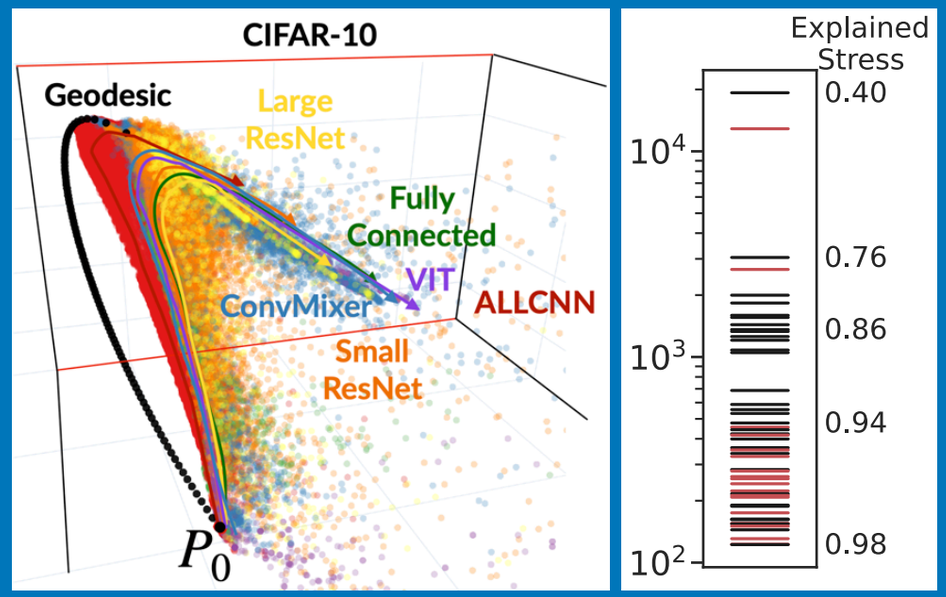



The eigenvalues are sloppy, and the first two inPCA components have meaning.

Taking the limit of zero data, we generalized PCA to use the Bhattacharyya divergence instead of the Hellinger distance, yielding the model manifold here shown projected onto the largest two inPCA components. On the left find the sloppy eigenvalues evaluated at the best fit to our Universe (dot near center). On the right, find the lengths of the model manifold as measured by the variance of the inPCA components (not as sloppy).

# Visualizing Deep Neural Networks

Mao, Chaudhari, Griniasty, Teoh







**Hyperribbon formed by neural net ensemble during training**

First three inPCA singular components capture 76% of the motion

We can watch how neural networks learn with our visualization methods. Here each dot is a snapshot of a single neural network as it learns 50000 images. They all start out ignorant at the beginning, and figure out which are frogs and which are cats by the end. The probability space has around half a million dimensions. Three inPCA components capture 76% of the information about their learning paths. What is amazing is not that we can tell the differences between different network architectures. It is that we can view them at all
for such a complex, nonlinear, high-dimensional learning process.

# *isKLe:* Explicit coordinates for exact intensive embedding in finite dimensions

Han Kheng Teoh
Katherine Quinn

# isKLe for Stat Mech

## Deep links in a global distance

Info Geometry: symmetrized KL as distance (Han-Kheng Tao)

$$d^2_{\mathrm{sKL}}(\rho_1,\rho_2) = \sum_{\mathbf{x}} (\rho_1(\mathbf{x}) - \rho_2(\mathbf{x}))\log(\rho_1/\rho_2) = \langle \log(\rho_1/\rho_2)\rangle_{\rho_1} + \langle \log(\rho_2/\rho_1)\rangle_{\rho_2}$$

Statistical Mechanics (Ising)

$$\langle \log \rho_i \rangle_{\rho_j} = -\log Z_i - \beta_i e_j + h_i m_j,$$

where $e = \Sigma s_i s_j$, $m = \Sigma s_i$, $\beta = J/k_B T$, $h = H/k_B T$

Explicit Coordinates for Model Manifold

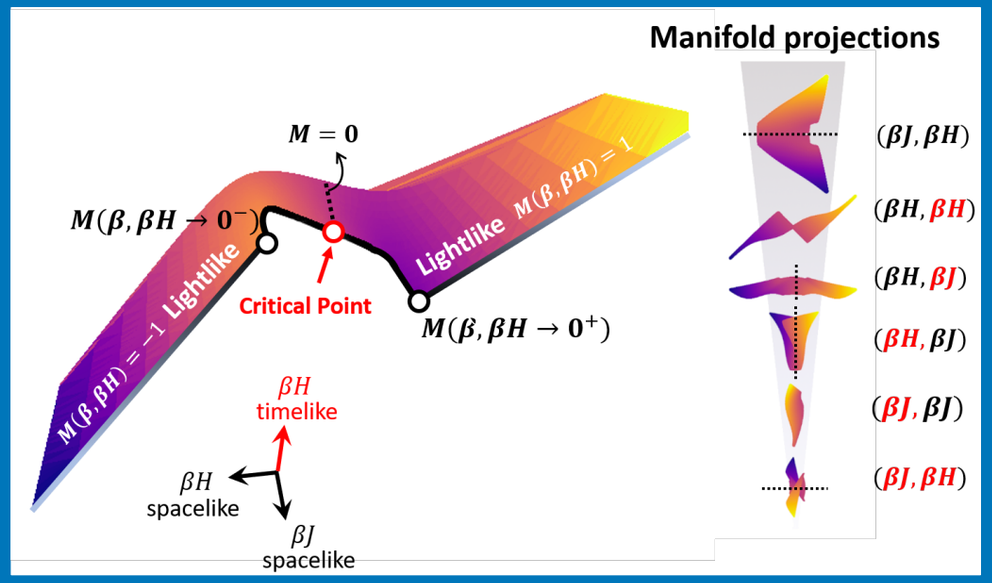
$$\begin{aligned} d^2_{\mathrm{sKL}}(\rho_1,\rho_2) &= -(\beta_1 - \beta_2)(e_1 - e_2) + (h_1 - h_2)(m_1 - m_2) \\ &= (1/4)[((e_1 - \beta_1) - (e_2 - \beta_2))^2 - ((e_1 + \beta_1) - (e_2 + \beta_2))^2 \\ &\quad + ((m_1 + h_1) - (m_2 + h_2))^2 - ((m_1 - h_1) - (m_2 - h_2))^2] \end{aligned}$$

Space-like / Time-like coordinates $(e \pm -\beta)/2$ and $(m \pm -h)/2$;
Squares contribute positively and negatively to $d^2_{\mathrm{sKL}}$

Exploring alternative intensive distances to the Bhattacharyya distance, Han Kheng Tao discovered that the symmetrized Kullback-Leibler divergence gave an intensive embedding for many problems that lived in a finite-dimensional Minkowski-like space. Probing further, we found that we could express the coordinates in terms of the ‘natural parameters’ and ‘sufficient statistics’ for any probability rho in an ‘exponential family’. For the Ising model, the coordinates are in terms of familiar quantities: temperature and energy, field and magnetization.

# 2D Ising Model: isKL Embedding

Han Kheng Teoh, Katherine Quinn, Colin Clement




isKLe is model graph, tilted by $\pi/4$

2+2 Minkowski space. Light-like direction allows distinction between fully magnetized states at high fields and various temperatures.

Here is the isKL embedding of the model manifold of the 2D Ising model in a field. We just ran Monte Carlo and plotted it! Note that we can think of the embedding as a 4D plot of the results (e, m) versus the parameters (beta, h), tilted by 45 degrees (which is not an allowed rotation in Minkowski space). Note the critical point. There is a weird fact, that the +-M(T) states at H=0 are at zero FIM distance from one another (because they have the same free energy). Note that isKLe also nicely distinguishes them using a light-like coordinate.
Is this a hyperribbon? It is infinitely thin in all but the first four cross-sections!

# Big Questions

## Consequences of Sloppiness

- PCA useful because high-dimensional data sets dominated by a few ‘component’ directions with large variation
- The best algorithms for least-squares fits approximate geodesics on the model manifold
- Biological evolution is moving in a high-dimensional sloppy genotype space whose phenotype space is a hyperribbon.
- Gross simplifications capture reality
  - Humans simplify images into ‘cartoons’
  - Experts distill experience into ‘intuition’
  - Big data draws low-dimensional representations from high-dimensional data sets
  - Macroeconomics, systems biology use grossly oversimplified models to predict system behavior

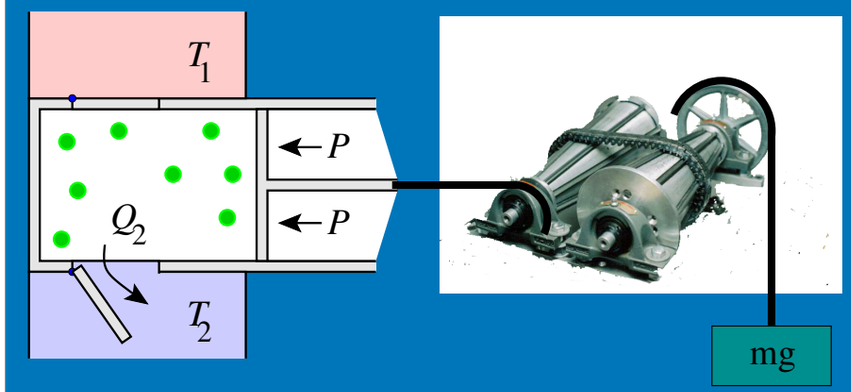
Underlying hyperribbons explain why cartoons work? Or cartoons work because we use hyperribbons to analyze reality?


- PCA useful because high-dimensional data sets dominated by a few ‘component’ directions with large variation
- The best algorithms for least-squares fits approximate geodesics on the model manifold
- Biological evolution is moving in a high-dimensional sloppy genotype space whose phenotype space is a hyperribbon.
- Gross simplifications capture reality
  - Humans simplify images into ‘cartoons’
  - Experts distill experience into ‘intuition’
  - Big data draws low-dimensional representations from high-dimensional data sets
  - Macroeconomics, systems biology use grossly oversimplified models to predict system behavior

Underlying hyperribbons explain why cartoons work? Or cartoons work because we use hyperribbons to analyze reality?

# Control and Carnot Corrections

**Ben Machta (not me)**



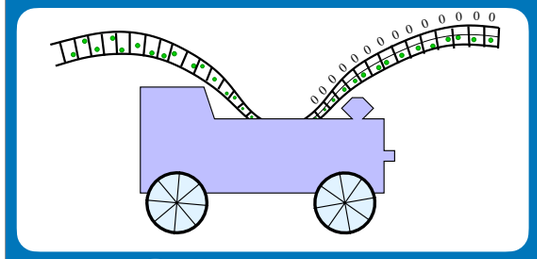

Variable Transmission
$\theta$ = Pressure

Mass falls to drive piston
(steady costs entropy)
Pressure fluctuations allow mass to fall
(slow costs entropy)

$$\langle \Delta S_{\text{control}} \rangle = 2 \int_{P_i}^{P_f} \sqrt{g_{PP}} dP$$

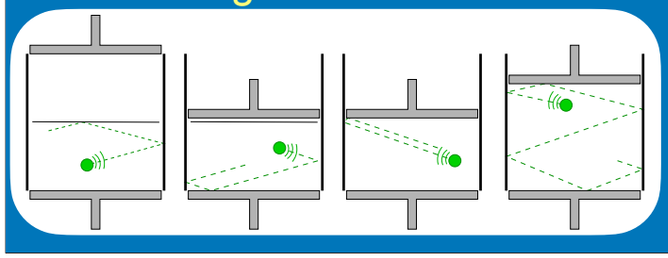
# Machta vs. fungibility of entropy

## Can information & thermodynamic entropy be exchanged?

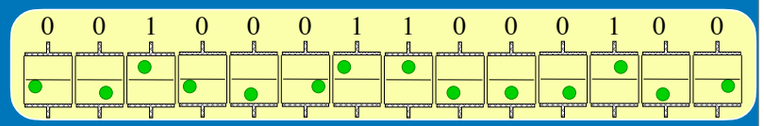


Szilard Engine
(Bennett, Feynman)

Work done by expanding piston is kT log 2 = T ΔS



Szilard argued that information can be exchanged for work. Feynman envisions a locomotive burning information from a data tape storing bits as atoms on one side or another of a partitioned piston.

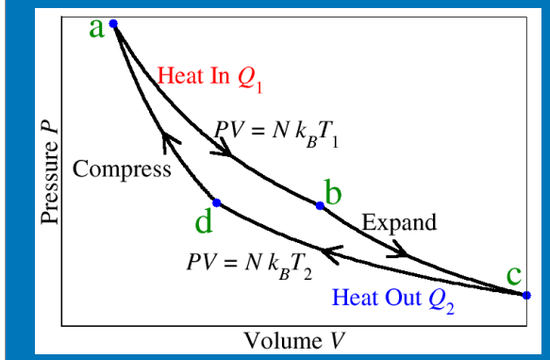


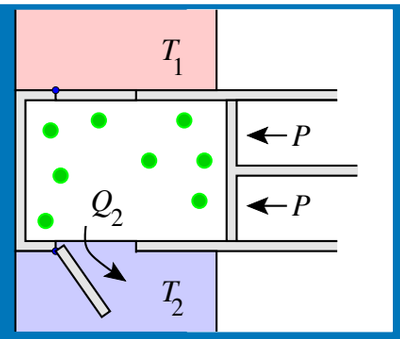

Machta argues that controlling the piston will cost entropy 4 k log 2, a net loss!

$$\langle \Delta S_{\text{control}} \rangle = 4\sqrt{N} \log(P_a/P_b) \cdot$$

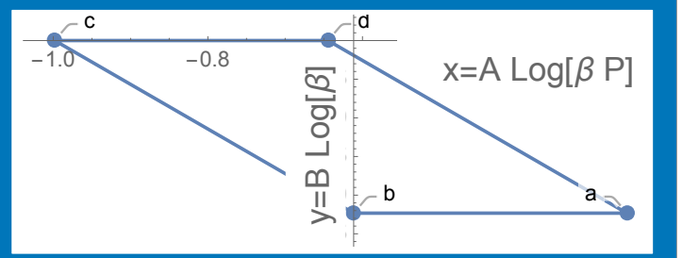
# Carnot Cycles on the Piston Manifold

$$\langle \Delta S_{\text{control}} \rangle = 4\sqrt{N}\log(P_a/P_b) + 2\sqrt{15N}\log(T_1/T_2)$$




**Carnot cycle**
Isothermal Expansion,
Adiabatic, Change Bath,
Isothermal, Adiabatic
Model Manifold is a *plane*,
Cartesian coordinate change




Machta's entropy cost sub-extensive:
Refrigerators OK *(√N)*