Lecture 8:

Manifold learning (2)

1. Laplacian Eigenmaps
2. Log map / Riemannian manifold learning

Dimensionality Reduction as Sample Embedding problem:

Given an input dataset $\mathbf{X}_n = \{X_1, X_2, \ldots, X_n\} \subset R^p$, find an '$n$-point' Embedding mapping

$$h_{(n)}: \mathbf{X}_n \to \mathbf{Y}_n = h_{(n)}(\mathbf{X}_n) = \{y_1, y_2, \ldots, y_n\} \subset R^q,$$

such that the resulting q-dimensional dataset $\mathbf{Y}_n$, $q < p$, ***faithfully represents*** the sample $\mathbf{X}_n$

Embedding: original high-dimensional inputs $\mathbf{X}_n$ $\to$ low-dimensional features $\mathbf{Y}_n = \{y_1, y_2, \ldots, y_n\}$

Laplacian Eigenmaps (LEM)

(Belkin, Niyogi: Laplacian Eigenmaps for Dimensionality Reduction and Data Representation, 2003)

Graph-based algorithms: have 3 basic steps.

1. Find sample points from small neighborhoods of all sample points (ε -ball)

2(a). Constructing the Adjacency graph $\Gamma(\mathbf{X}_n)$ describing local properties of the Data manifold:

- the sample points $\{X_j\}$ are nodes
- the edges $\mathbf{E} = \{E_{ij}\}$: E_{ij} connect the nodes X_i and X_j if and only if $|X_i - X_j| < \varepsilon$

2(b). Setting the weights to the Adjacency graph: edge $(X_i, X_j) \rightarrow$ weight $\exp\left\{-\frac{1}{t}|X_i - X_j|^2\right\}$, $t > 0$

$W = \|W_{ij}\|$ - $n\times n$ weighted adjacent matrix (uses 'heat-transfer' kernel):

- $W_{ij} = \exp\left\{-\frac{1}{t}\left|X_i - X_j\right|^2\right\}$, if the nodes X_i and X_j are connected ($W_{ij} = 1$ when $t = \infty$)
- $W_{ij} = 0$, otherwise

Weighted adjacency graph $\Gamma(\mathbf{X}_n) = (\mathbf{X}_n, \mathbf{E})$ / weighted matrix W reflect the intrinsic geometric structure of the Data manifold – 'Data manifold is approximated by the adjacency graph'

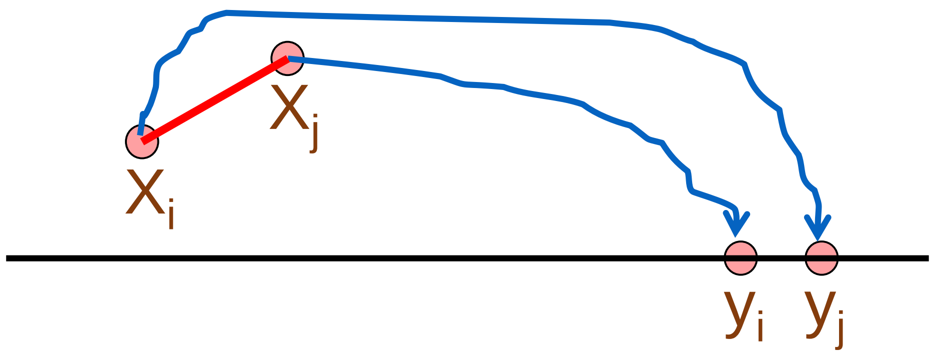
3. Find a global embedding that preserves the local properties:

   - choosing a cost function $L(\mathbf{Y}_n) = \sum_{i,j=1}^{n} W_{ij} \times \|y_i - y_j\|^2$
   - minimization of chosen cost function $L(\mathbf{Y}_n)$ over q-dimensional features $\{y_1, y_2, \dots, y_n\}$

Cost function $L(\mathbf{Y}_n) = \sum_{i,j=1}^{n} W_{ij} \times \|y_i - y_j\|^2$

Features $\{y_1, y_2, \dots, y_n\}$: the images (y_i, y_j) of connected nodes (X_i, X_j) in the weighted graph $\Gamma(\mathbf{X}_n)$ should stay as close together as possible

Example: $q = 1$



Constraints:

1) features $\{y_1, y_2, \dots, y_n\}$ may be determined up to a shift: require for definiteness: $\sum_{i=1}^{n} y_i = 0$

2) features $\{y_1, y_2, \dots, y_n\}$ may be determined up to a scale: to avoid trivial degenerate solution, require

$$\sum_{i=1}^{n} (y_i \times W_i \times y_i^T) = I_q$$

using the weights $W_i = \sum_{j=1}^{n} W_{ij}$, $i = 1, 2, \dots, n$

This constraint removes an arbitrary scaling factor in the features

Optimization problem: $L(\mathbf{Y}_n) = \sum_{i,j=1}^{n} W_{ij} \times \left\| y_i - y_j \right\|^2 \rightarrow \min$

under constraints: $\sum_{i=1}^{n} y_i = 0$ and $\sum_{i=1}^{n} \left(y_i \times W_i \times y_i^T \right) = I_q$

$$L(\mathbf{Y}_n) = \sum_{i,j=1}^{n} W_{ij} \times \left\| y_i - y_j \right\|^2 = \sum_{i,j=1}^{n} W_{ij} \times \left(y_i - y_j \right)^T \times \left(y_i - y_j \right)$$

$$= \sum_{i,j=1}^{n} W_{ij} \times \left\{ y_i^T \times y_i + y_j^T \times y_j - 2 y_i^T \times y_j \right\}$$

$$= 2 \times \left\{ \sum_{i=1}^{n} W_i \times \left(y_i^T \times y_i \right) - \sum_{i,j=1}^{n} W_{ij} \times \left(y_i^T \times y_j \right) \right\}$$

$\mathbf{Y}$ - $n \times q$ matrix with rows $y_1^T, y_2^T, \dots, y_n^T$

$W = \|W_{ij}\|$ - $n \times n$ **weighted adjacent matrix** $\rightarrow$ $D = \mathrm{Diag}(W_1, W_2, \dots, W_n)$ - $n \times n$ **degree matrix**

$$\sum_{i=1}^{n} W_i \times (y_i^T \times y_i) - \sum_{i,j=1}^{n} W_{ij} \times (y_i^T \times y_j) = \mathrm{Tr}(\mathbf{Y}^T \times D \times \mathbf{Y}) - \mathrm{Tr}(\mathbf{Y}^T \times W \times \mathbf{Y})$$

$$= \mathrm{Tr}(\mathbf{Y}^T \times L \times \mathbf{Y})$$

$L = D - W$ - $n \times n$ **Laplacian matrix**

$\sum_{i=1}^{n} y_i = 0 \quad \rightarrow \quad \mathbf{Y}^T \times \mathbf{1}_n = 0 \qquad \sum_{i=1}^{n} (y_i \times W_i \times y_i^T) = I_q \quad \rightarrow \quad \mathbf{Y}^T \times D \times \mathbf{Y} = I_q$

Optimization problem: $\mathrm{Tr}(\mathbf{Y}^T \times L \times \mathbf{Y}) \rightarrow \min$

under constraints: $\mathbf{Y}^T \times \mathbf{1}_n = 0$ and $\mathbf{Y}^T \times D \times \mathbf{Y} = I_q$

Optimization problem: $Tr(\mathbf{Y}^T \times L \times \mathbf{Y}) \to \min$ under constraint $\mathbf{Y}^T \times D \times \mathbf{Y} = I_q$

This problem reduces to finding the 'minimum eigenvalue solution' to generalized eigenvalue problem:

$$LY = \lambda DY$$

$(q+1)$ smallest eigenvalues: $0 = \lambda_0 \leq \lambda_1 \leq \ldots \leq \lambda_q$

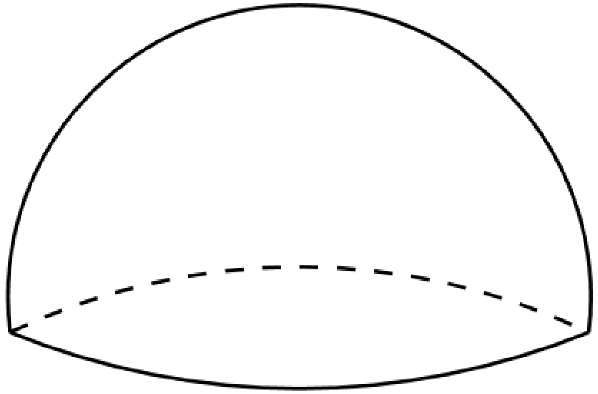
corresponding orthonormal eigenvectors: $Y_0, Y_1, Y_2, \ldots, Y_q \in R^n$

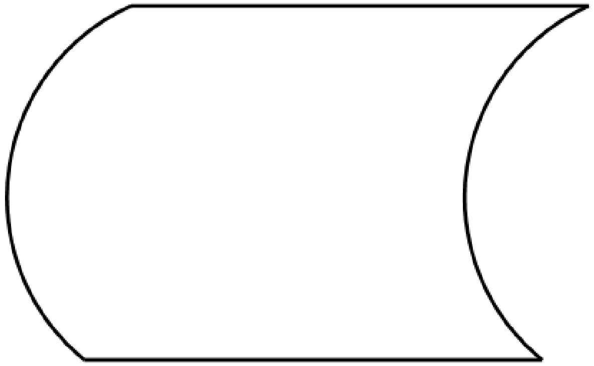
the eigenvalue $\lambda_0 = 0$ corresponds to the vector $\mathbf{1}_n$ and should be discarded due constraint $\mathbf{Y}^T \times \mathbf{1}_n = 0$

- the eigenvectors $Y_1, Y_2, \ldots, Y_q \in R^n$ are taken as columns of desired $n \times q$ matrix $\mathbf{Y}$
- the columns $y_1, y_2, \ldots, y_n \in R^q$ of $\mathbf{Y}^T$ are taken as desired low-dimensional features

ISOMAP provides a isometric embedding that preserves global geodesic distances

But it works only when **the surface can flatten**





cannot flatten —— can flatten

nonzero curvature —— zero curvature

No accurate map of Earth exists – Gauss's theorem.

Laplacian eigenmaps tries to preserve the geometric characteristics of the Data manifold

by preserving 'locality properties'

Math: Laplacian Eigenmaps justification – why Laplacian?

$\mathbf{M} \subset R^p$ – unknown Data manifold

$\mathbf{X}_n = \{X_1, X_2, \dots, X_n\} \subset M$ – training dataset (sample)

$h: \mathbf{M} \rightarrow R^q$ - desired Embedding mapping

$\mathbf{X}_n \rightarrow \mathbf{Y}_n = h(\mathbf{X}_n) = \{y_1, y_2, \dots, y_n\} \subset R^q$ – desired resulting q-dimensional dataset

$L(\mathbf{Y}_n) = \sum_{i,j=1}^{n} W_{ij} \times \|y_i - y_j\|^2 = \sum_{i,j=1}^{n} W_{ij} \times \|h(X_i) - h(X_j)\|^2 = F_n(h)$ - functional

Constraint: $D_n(h) = \sum_{i=1}^{n}\left(h(X_i) \times W_i \times h^T(X_i)\right) = I_q$

Let $q = 1$, $\nabla h(X) = \begin{pmatrix} \frac{\partial h}{\partial X_1} \\ \cdots \\ \frac{\partial h}{\partial X_p} \end{pmatrix} \in R^p$ – gradient $\nabla_{\mathbf{M}} h(X)$ – covariant differentiation is used

$X', X \in \mathbf{M}$ – near points: $h(X') - h(X) = (\nabla_{\mathbf{M}} h(X), X' - X) + o(X' - X)$ - Taylor expansion

$|h(X') - h(X)| \le |\nabla_{\mathbf{M}} h(X)| \times |X' - X| + o(|X' - X|)$ -

$|\nabla_{\mathbf{M}} h(X)|$ provides us with an estimate of how far apart h maps nearby points.

We look a mapping h that 'best preserves locality on average' – minimizes functional

$$F(h) = \int_{\mathbf{M}} |\nabla_{\mathbf{M}} h(X)|^2 \mathrm{mes}(dX)$$

under scale constraint $\|h\|^2 = \int_{\mathbf{M}} |h(X)|^2 \mathrm{mes}(dX) = 1$ to avoid trivial degenerate solution

$$F(h) = \int_{\mathbf{M}} |\nabla_{\mathbf{M}} h(X)|^2 mes(dX) \approx F_n(h) = \sum_{i,j=1}^{n} W_{ij} \times \left\| h(X_i) - h\left(X_j\right) \right\|^2$$

Minimizing $F(h)$ over h defined on the Data manifold **M** corresponds to minimizing $F_n(h)$ over $h_{(n)}$ defined on nodes of the graph $\Gamma(\mathbf{X}_n)$

Constraint: $\|h\|^2 = \int_{\mathbf{M}} |h(X)|^2 mes(dX) = 1 \sim D_n(h) = \sum_{i=1}^{n} \left(h(X_i) \times (\sum_{j=1}^{n} W_{ij}) \times h^T(X_i) \right) = 1$

- minimization is doing under similar constraints

Stokes theorem: $F(h) = \int_{\mathbf{M}} |\nabla_{\mathbf{M}} h(X)|^2 \mathrm{mes}(dX) = \int_{\mathbf{M}} (h \times \Delta_{\mathbf{M}} h)(X) \mathrm{mes}(dX)$

$\Delta_{\mathbf{M}}(h)$: $h(X) \to -\,\mathrm{div}(\nabla_{\mathbf{M}} h(X)) = -\sum_{k=1}^{p} \frac{\partial^2 h(X)}{\partial X_k^2}$

Laplace operator, if $X \in R^p$

Laplace-Beltrami operator L_{LB}, if $X \in \mathbf{M}$

$\int_{\mathbf{M}} (h \times \Delta_{\mathbf{M}} h)(X) \mathrm{mes}(dX) \approx \sum_{i,j=1}^{n} W_{ij} \times \left\| h(X_i) - h(X_j) \right\|^2 = (h(\mathbf{X}_n), L \times h(\mathbf{X}_n))$ $\mathbf{Y} = h(\mathbf{X}_n)$

Matrix L - sampling analogous of the Laplace-Beltrami operator – called Laplacian of a graph $\Gamma(\mathbf{X}_n)$

Laplace-Beltrami operator $\Delta_{\mathbf{M}}$: there exists eigensystem $\{f_i(X),\ i = 0, 1, \ldots\}$ consisting of orthonormal eigenfunctions:

$$\Delta_{\mathbf{M}} f_i = \lambda_i \times f_i, \qquad (f_i(X), f_j(X)) = \delta_{ij} \qquad \|f_i\|^2 = \int_{\mathbf{M}} |f_i(X)|^2 \mathrm{mes}(dX) = 1$$

corresponding to eigenvalues $0 = \lambda_0 \le \lambda_1 \le \lambda_2 \le \ldots$ $\qquad f_0$ - const

The eigenfunctions have properties desirable for embedding

The rows of best Embedding mapping

$$h(X) = \begin{pmatrix} h_1(X) \\ \cdots \\ h_q(X) \end{pmatrix} : X \in \mathbf{M} \to y = h(X) \in R^q$$

consists of the eigenvectors $\{f_1(X), f_2(X), \ldots, f_q(X)\}$ corresponding smallest eigenvalues $\lambda_1 \le \ldots \le \lambda_q$

Locally Linear Embedding (LLE): sampling analogous of continuous eigenfunction problem for the squared Laplace-Beltrami operator L_{LB}^2

- has the same eigenfunctions as L_{LB}
- finds the same 'best' embedding

Intrinsic dimension estimation:

$\lambda_1 \leq \lambda_2 \leq \dots$ - eigenvalues. Then exists the constants A and B:

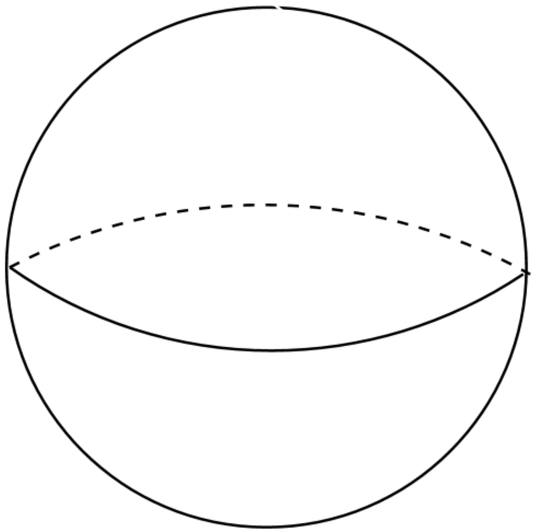
$$A + \frac{2}{q} \times \log(j) \leq \log(\lambda_j) \leq B + \frac{2}{q} \times \log(j + 1)$$

Riemannian manifold learning (RML) – Riemannian Normal Coordinates (RNC)

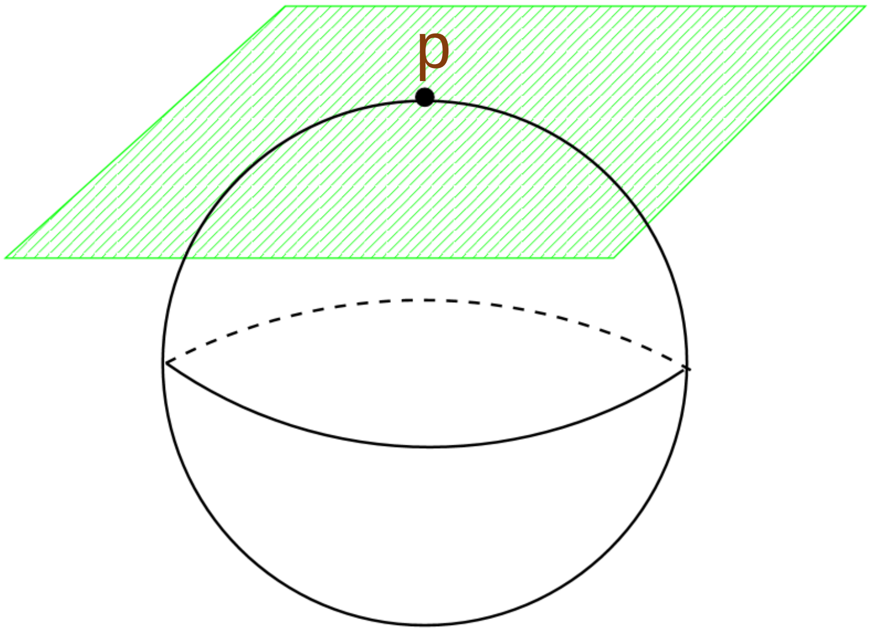
(Brun et al. Fast Manifold Learning Based on Riemannian Normal Coordinates, 2005;

Lin et al. Riemannian Manifold Learning for Nonlinear dimensionality reduction, 2006)

Riemannian Normal Coordinates: B. Riemann (**1854**)

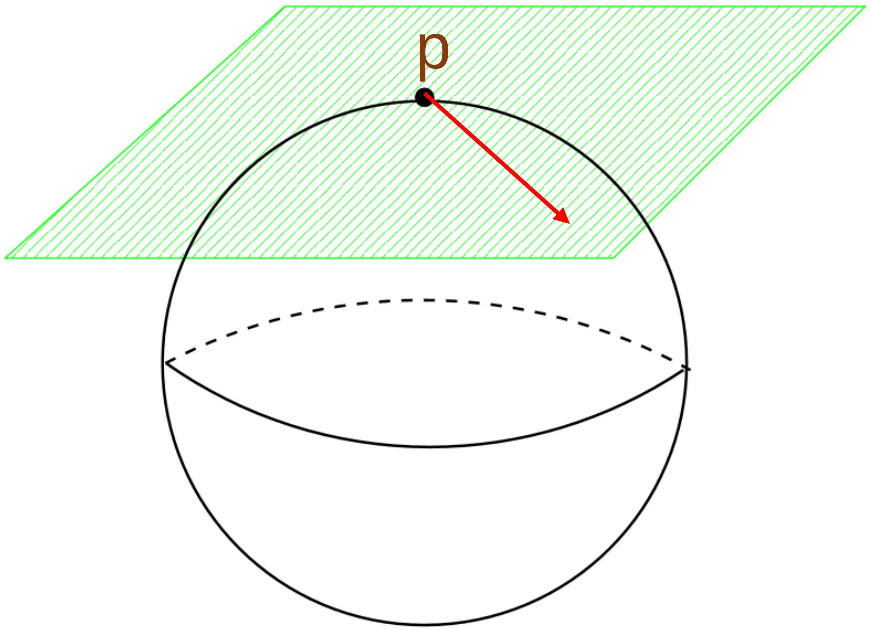


Data manifold **M**

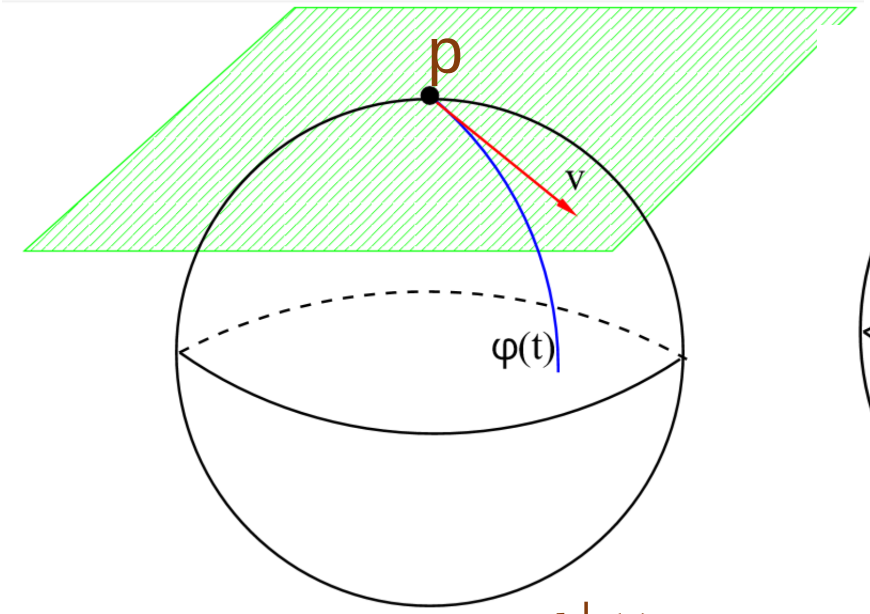


Tangent space $T_p\mathbf{M}$

$\mathrm{Dim}T_p\mathbf{M} = q$

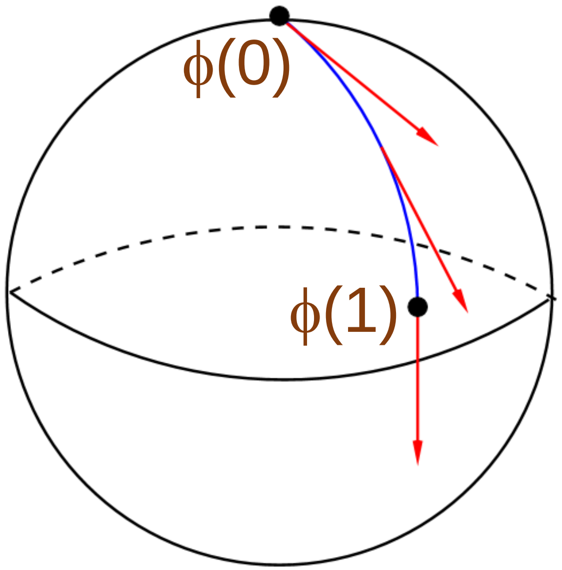


Tangent vector



$\phi(t)$ – curve on $\mathbf{M}$, $\dfrac{d\phi(t)}{dt} = v$,

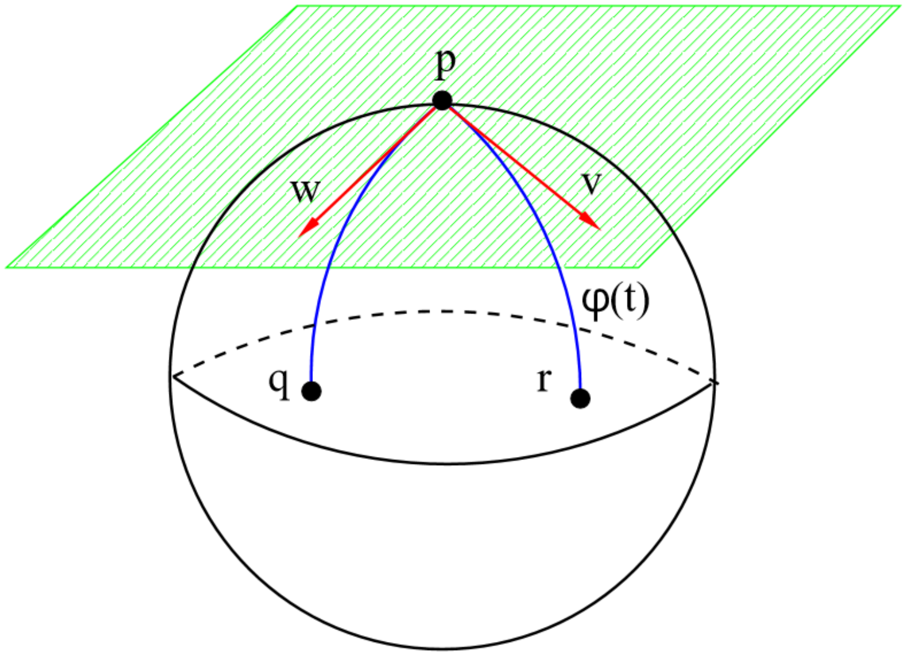
$v \in T_p\mathbf{M}$ – tangent vector



$\phi(t)$: $[0, 1] \rightarrow \mathbf{M}$,

$$L(\phi) = \int_0^1 \left|\frac{d\phi(t)}{dt}\right|^2 dt$$

Geodesic – shortest curve between two points



Geodesic $\phi(t)$

$$\phi(0) = p,\ \dot{\phi}(0) = \left.\frac{d\phi(t)}{dt}\right|_{t=0} = v$$

$$\phi(|v|) = r$$

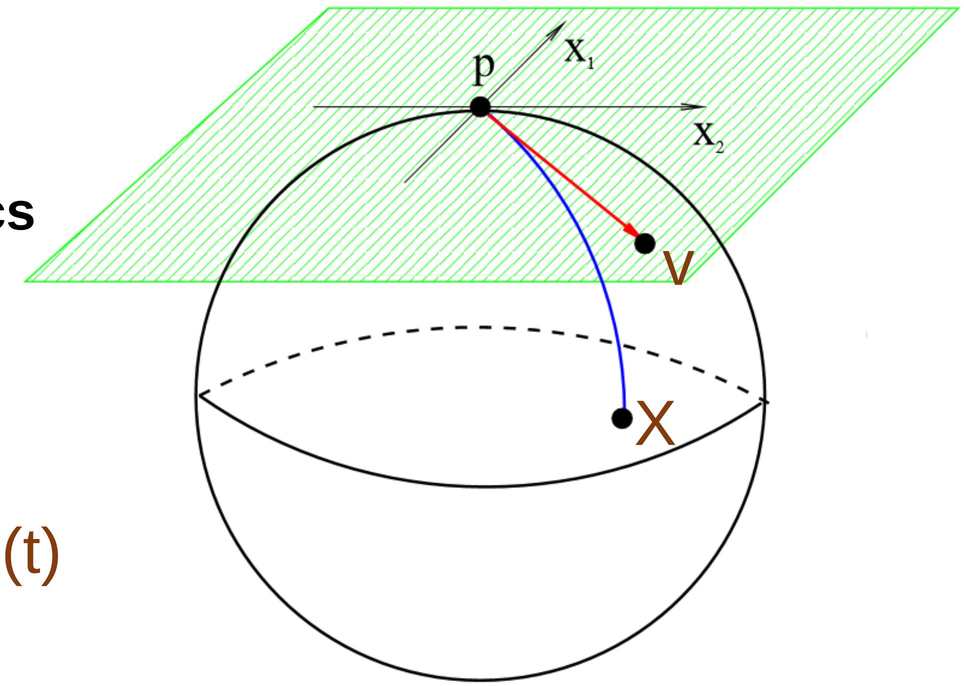
Exponential map $\exp_p$: $T_p\mathbf{M} \rightarrow \mathbf{M}$

$\exp_p(v) = r$, $\exp_p(w) = q$

$\mathrm{dist}_{\mathbf{M}}(p, X)$ – a length of shortest curve (geodesic) between points p and X

Math: For any point $p \in \mathbf{M}$ and any vector $v \in T_p\mathbf{M}$ there exists **a unique geodesic** $\phi(t)$ such that:

$$\phi(0) = p,\ \dot{\phi}(0) = v$$



All the geodesics passing through p are called **radial geodesics**

Exponential map:

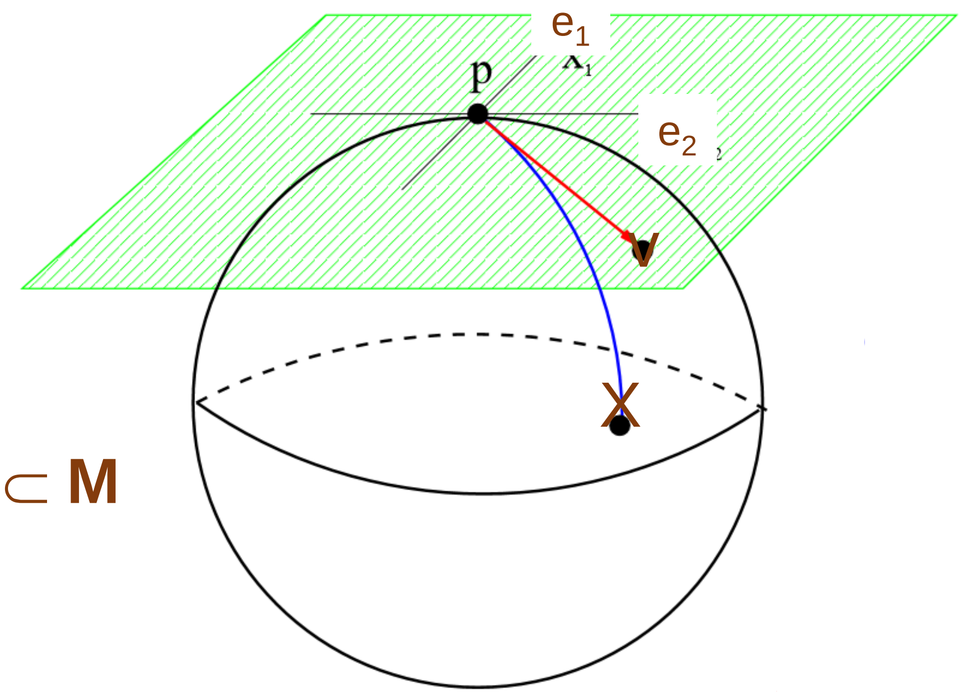
- tangent vector $v \in T_p\mathbf{M}$ determines the unique geodesic $\phi(t)$ traveling through p with the tangent vector v
- geodesic $\phi(t)$ determines a point $X = \phi(1) = \exp_p(v) \in \mathbf{M}$ such that $\mathrm{dist}_{\mathbf{M}}(p, X) = |v|$

Exponential map $v \in T_p\mathbf{M} \to X = \exp_p(v) \in \mathbf{M}$ is one-to-one in a neighborhood U of p

$v = \log_p(X)$ - inverse **logarithmic mapping** (**log map**) - parameterization of the neighborhood U

$$\log_p: X \in U \subset \mathbf{M} \to v \in T_p\mathbf{M}$$

Local coordinates defined by the chart $(U, \log_p)$ - **Riemannian Normal Coordinates** with center p



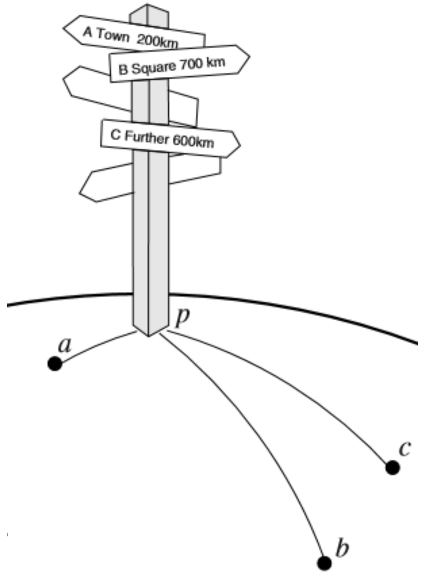
$\{e_1, e_2, \dots, e_q\}$ – basis in $T_p\mathbf{M}$, $v = \sum_{i=1}^{q} y_i \times e_i$

$\{y_1, y_2, \dots, y_q\}$ – Riemannian Normal Coordinates of $X \in U \subset \mathbf{M}$

$v = \sum_{i=1}^{q} y_i \times e_i \in T_p\mathbf{M}$ $\rightarrow$ polar coordinates: $v \rightarrow (e = \frac{v}{|v|}, |v|)$

Geodesic $\phi(t)$: $\phi(0) = p$, $\dot{\phi}(0) = e \in S_q(p)$ $S_q(p) = \{v \in T_p\mathbf{M}: |v| = 1\}$ – unit sphere

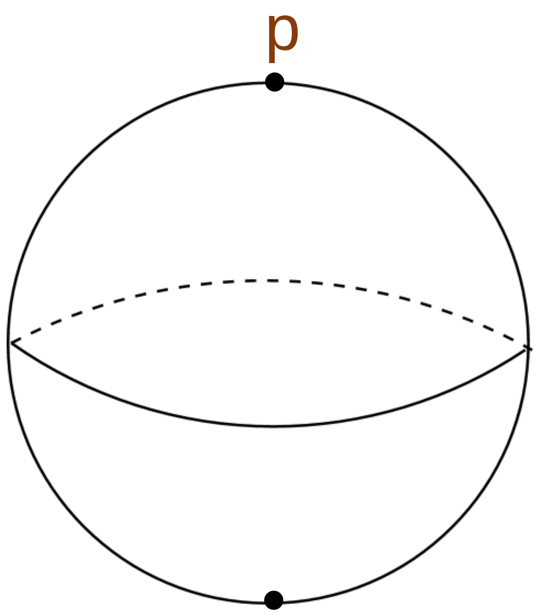
$X = \phi(|v|) = \exp_p(v) \in \mathbf{M}$



$(e, \mathrm{dist}_M(p, X))$ – Riemannian Normal Coordinates of the point $X \in \mathbf{M}$:

- the direction e, $|e| = 1$, in which you should move to reach the point X by the shortest way
- the length $\mathrm{dist}_M(p, X)$ which you should move to reach the point X by the shortest way

The set CL(p) of points on $\mathbf{M}$ for which there exists more than one shortest path from p - **cut locus** of p



$$\mathbf{M}(p) = \mathbf{M} \setminus CL(p)$$

$X \in \mathbf{M}(p) \to v = \log_p(X)$ - Riemannian Normal Coordinates with center p

Cut locus of p - antipodal point

Manifold embedding via Riemannian Normal Coordinates (RNC)

- input dataset $\mathbf{X}_n = \{X_1, X_2, \dots, X_n\} \subset R^p$
- desired 'n-point' dataset $\mathbf{Y}_n = \{y_1, y_2, \dots, y_n\} \subset R^q$

An approach:

- 'ISOMAP' part (start)
- Riemannian Normal Coordinates construction (main content)

ISOMAP part

- find sample points from small neighborhoods (ε -ball, K nearest neighbors) of the sample points

- construct a weighted undirected adjacency graph $\Gamma(\mathbf{X}_n)$ with sample points $\{X_i\}$ as nodes and edges connecting 'near' nodes X_i and X_j with Euclidean distances $|X_i - Xj|$ as its weight

- compute the shortest-distance paths between all the graph vertices (Dijkstra's algorithm) – $\{D(X_i, X_j)\}$ the lengths of the shortest "geodesic paths" between the points

Main algorithm - LOGMAP (Brun et al., 2005)

- choose a base point $p \in \mathbf{M}$

(1) $p = \bar{X}$ - mean vector

(2) p – a point with minimal geodesic radius (minimal eccentricity): 'one single chart at this point can represent the entire manifold'

$E(X) = \max_j D(X, X_j)$ – geodesic radius (eccentricity) of a point $X \in \mathbf{X}_n$

$p = \arg\min E(X)$ – the point with minimal geodesic radius (eccentricity)

- for selected point $X \in \mathbf{X}_n$, find the RNC of X:

- direction $e \in S_q(p)$ of the geodesic curve connecting the points p and X

- the length $dist_M(p, X)$ of this geodesic curve – can be estimated by $D(p, X)$

Estimating the direction $e \in S_q(p)$

- $f(Z) = (dist_M(Z, X))^2$ - length of geodesic curve connecting points X and $Z \in U$ (neighborhood of p)

- $f(X') \approx D^2(X', X)$ – approximate known values $f(Z)$ at the points $Z = X' \in U \cap \mathbf{X}_n$

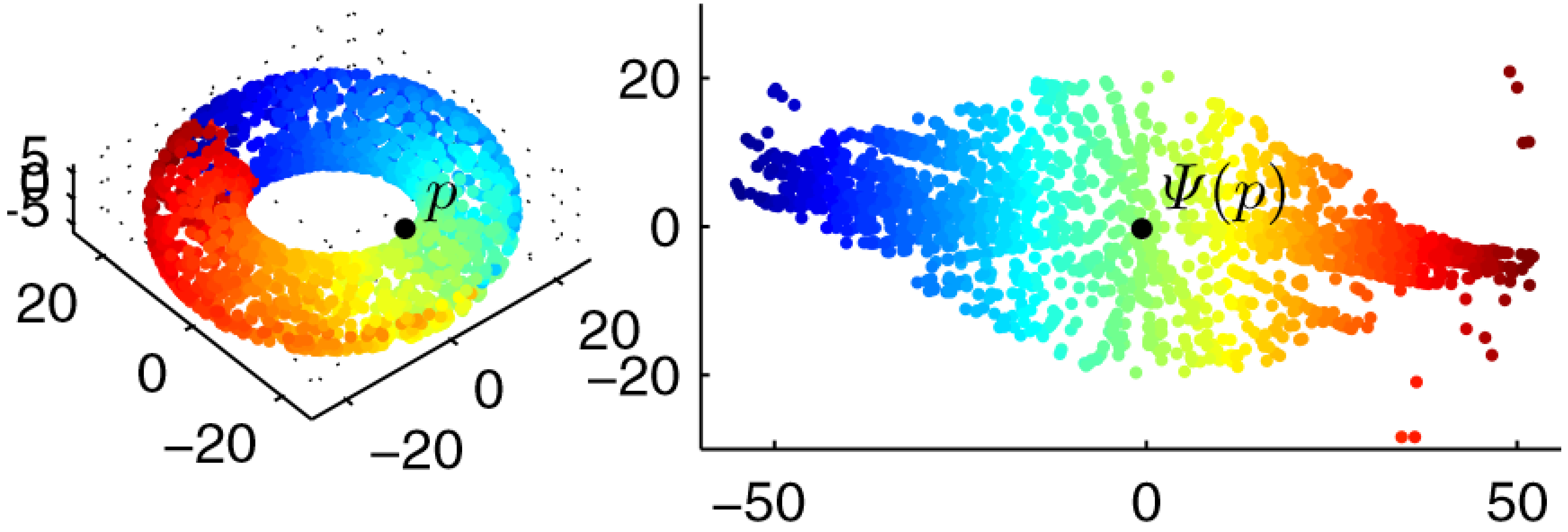
- $\varphi(Z)$ - a second order polynomial which approximates the function $f(Z)$ from known values $\{f(X')\}$

- $g = \nabla\varphi(Z)|_{Z=p} \in R^q$ – gradient at point p

- $e = g / |g| \in e \in S_q(p)$ – normalization $\rightarrow$ desired direction

- $y = dist_M(p, X) \times e \in R^q$ – Riemannian Normal Coordinates of point $X \in \mathbf{M}(p)$

Example (Torus embedded in 3D)



- training dataset consisting of n = 2000 points
- torus embedded in 3D is not flat, the manifold is not mapped to a perfect rectangle
- some outliers are present, due to incorrect estimation of the gradient for points near the cut locus

Main algorithm - Riemannian Manifold Learning (RML)

(Lin, Zha, Lee: Riemannian Manifold Learning for Nonlinear Dimensionality Reduction, 2006)

- choose a base point $p \in \mathbf{M}$
- estimation of the Tangent space $T_p\mathbf{M}$

- sample points $X_0, X_1, \dots, X_q$ from the neighborhood U of base point p lying in a general position are selected

- $L(p) = X_0 + \mathrm{Span}(X_1 - X_0, X_2 - X_0, \dots, X_q - X_0)$ - estimator of the Tangent space $T_p\mathbf{M}$

- $\{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_q\}$ – orthonormal basis in $L(p)$

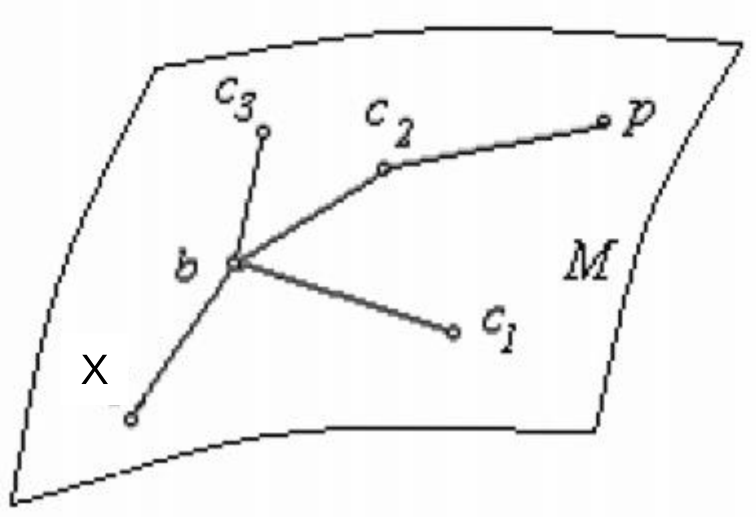
- constructing the Riemannian Normal Coordinates $\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n$ for sample points $X_1, X_2, \dots, X_n$

1) $X \in U \cap \mathbf{X}_n$: $\mathbf{z} = (z_1, z_2, \dots, z_q) = \mathrm{Pr}_{L(p)}X$ - projection into linear space $L(p)$

the solution to the Least Squares task: $\left\| X - (p + \sum_{i=1}^{q} z_i \times \mathbf{e}_i) \right\|_{(n)}^2$ $\|\cdot\|_{(k)}$ - Euclidean norm in R^k

$$\mathbf{y} = \frac{\|X - p\|_{(n)}}{\|X\|_{(q)}} \times \mathbf{z}$$ - Riemannian Normal Coordinates of X

2) $X \notin U \cap \mathbf{X}_n$:



b - the previous point on the shortest path from p to X

point b has $k \geq q$ edge points $c_1, c_2, \ldots, c_k$ whose normal coordinates

$\mathbf{y}_{(0)}$ and $\mathbf{y}_{(1)}, \mathbf{y}_{(2)}, \ldots, \mathbf{y}_{(k)}$ have been computed previously

α_i - an angle between vectors $(X - b)$ and $(c_i - b)$: $\cos(\alpha_i) = \dfrac{(X - b, c_i - b)}{\|X - b\| \times \|c_i - b\|}$, $i = 1, 2, \ldots, k$

β_i - an angle between their RNC - the vectors $(\mathbf{y} - \mathbf{y}_{(0)})$ and $(\mathbf{y}_{(i)} - \mathbf{y}_{(0)})$:

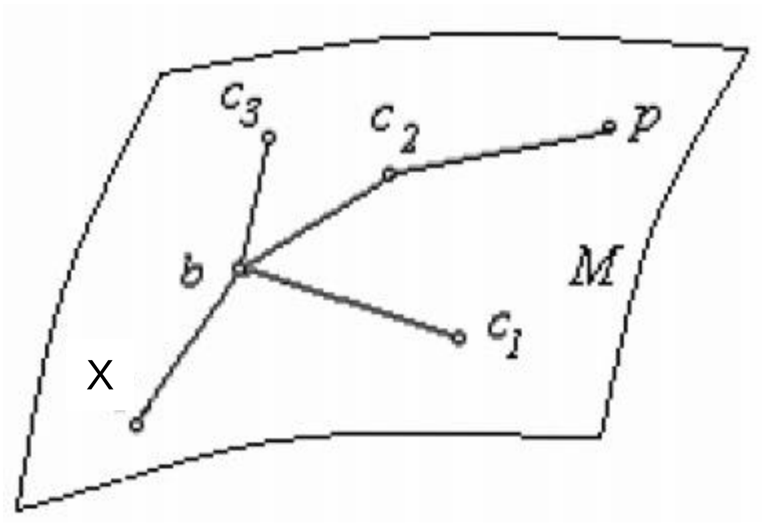
$$\cos(\beta_i) = \frac{\left(\mathbf{y} - \mathbf{y}_{(0)}, \mathbf{y}_{(i)} - \mathbf{y}_{(0)}\right)}{\left\|\mathbf{y} - \mathbf{y}_{(0)}\right\| \times \left\|\mathbf{y}_{(i)} - \mathbf{y}_{(0)}\right\|}, \quad i = 1, 2, \ldots, k$$

The desired RNC **y** is constructed

- to preserve the angles between the vectors $(X - b)$ and $(c_i - b)$ and their RNC: $\cos(\alpha_i) \approx \cos(\beta_i)$, $i = 1, 2, \ldots, k$

- to keep the distance between X and b unchanged:

$$\|X - b\| = \|\mathbf{y} - \mathbf{y}_{(0)}\|$$



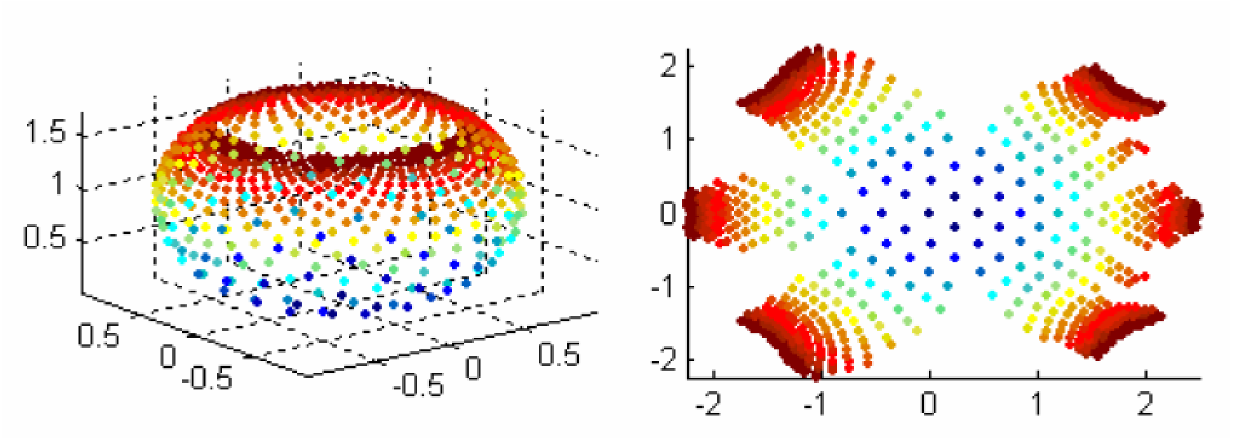
Optimization problem:

$$\left\|A \times \left(\mathbf{y} - \mathbf{y}_{(0)}\right) - h\right\|^2 \to \min$$ over **y** under the constraint $\|\mathbf{y} - \mathbf{y}_{(0)}\| = \|X - b\|$

$$h = \begin{pmatrix} \cos(\alpha_1) \\ \cdots \\ \cos(\alpha_k) \end{pmatrix} \in R^k$$

A - $k \times q$ matrix with i-th row $\dfrac{\left(\mathbf{y}_{(i)} - \mathbf{y}_{(0)}\right)^T}{\|X - b\| \times \left\|\mathbf{y}_{(i)} - \mathbf{y}_{(0)}\right\|}$, $i = 1, 2, \ldots, k$

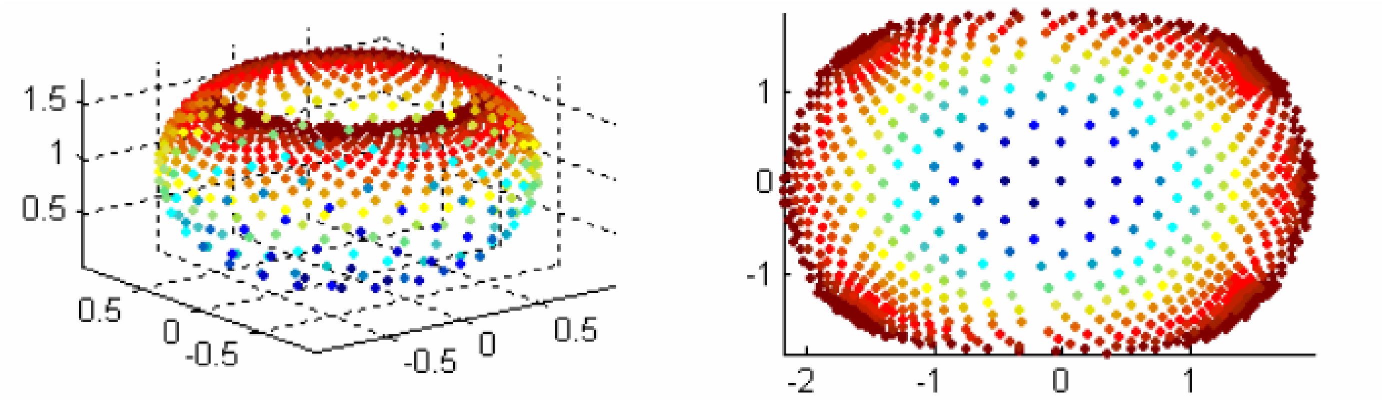
Linear least squares problem with a quadratic constraint: Lagrange multipliers + Newton's method



Improved algorithm - Riemannian Manifold Learning (RML)

(Lin, Zha: Riemannian Manifold Learning, 2008)

- improved procedure for computing the geodesics based on their local quadratic approximation



Locally Linear Embedding - 2000

ISOmetric MAPing (ISOMAP) - 2000

Laplacian Eigenmaps - 2003

Logmap - 2005

Riemannian Manifold Learning - 2006/2008

C-ISOMAP, L-ISOMAP - 2003

Hessian Eigenmaps - 2003

Local Tangent Space Alignment - 2004

Manifold charting - 2003

Semidefinite embedding - 2004

Diffusion maps (2005)/ Vector Diffusion maps (2012)

Grassmann&Stiefel Eigenmaps - 2012

…