Mmani Scalable Manifold Learning

Marina Meila with Jake Vanderplas

e-Science Institute Incubator Project 2014

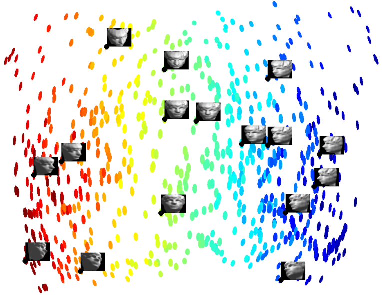
June 12, 2014

Overview

What is manifold learning?

Results from this Incubator project

Manifold Learning = non-linear dimension reduction



- Face images = high-dimensional data $p \in \mathbb{R}^D$ with $D = 64 \times 64 = 256 dimensions$
- can be described by a small number of continuous parameters = embedding in $\mathbb{R}^m$, $m << D$

Is Manifold Learning (ML) scalable?

- ML is data intensive
  - large amounts of data needed to reach accurate estimation of a manifold (e.g at least 10^{3-4})
- it is widely believed that ML is also computationally intensive
  - in particular, that it scales poorly with the sample size n

Is Manifold Learning (ML) scalable?

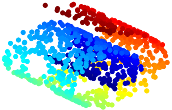
- ML is data intensive
  - large amounts of data needed to reach accurate estimation of a manifold (e.g at least 10^{3-4})
- it is widely believed that ML is also computationally intensive
  - in particular, that it scales poorly with the sample size n
- OUR PREMISE ML is no more expensive that PCA
  - i.e. non-linear dimension reduction is just as tractable as linear dimension reduction

Is Manifold Learning (ML) scalable?

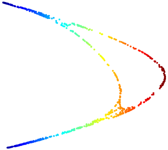
- ML is data intensive
  - large amounts of data needed to reach accurate estimation of a manifold (e.g at least 10^{3-4})
- it is widely believed that ML is also computationally intensive
  - in particular, that it scales poorly with the sample size n
- OUR PREMISE ML is no more expensive that PCA
  - i.e. non-linear dimension reduction is just as tractable as linear dimension reduction
- **This project:** implement ML in a scalable python package

Many manifold learning algorithms exist. . .

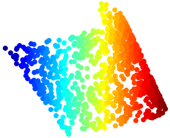
Original data
(Swiss Roll with hole)



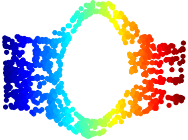
Laplacian Eigenmaps
(LE)



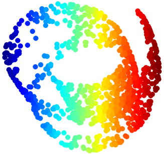
Hessian Eigenmaps
(HE)



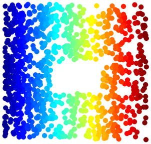
Isomap



Local Linear
Embedding (LLE)

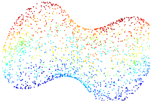


Local Tangent Space
Alignment (LTSA)

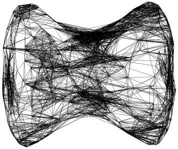


Typical manifold learning algorithm

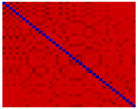
data $\{x_1, \ldots x_n\} \in \mathbb{R}^D$ find neighborhoods graph Laplacian L . . . . . .



$\longrightarrow$



$\longrightarrow$



Graph construction methods

- k-nearest neighbors
- ε-radius balls
- heat kernel with bandwidth ε (weighted neighborhoods)

$$W_{ij} = \exp\left(-\frac{||x_i - x_j||^2}{\varepsilon^2}\right) \quad \text{for } x_i, x_j \in \text{data}$$

From graph to embedding

- Preprocessing of the graph distance/adjacency matrix
- Eigendecomposition (with `arpack`) – find principal e-vectors
- Posprocess the e-vectors

Results from this Incubator project

Emb We reimplemented in python the core Diffusion Maps/Laplacian Eigenmaps family of algorithms.
In the process we changed some of the default choices in the algorithm to align them with the more recent advances in the field.

Metric Metric learning, an entirely new module for estimating the distortion in the embedding space was implemented.

Sim FLANN, an efficient approximate neighborhood graph package, representing the state of the art in this respect was incorporated in the software.

Astro we used the above module to analyze Galaxy Spectra from the SDSS

Python package Mmani Megamanifold

`Mmani.embedding.geometry.py` basic functionality for non-linear embedding algorithms

- `distance_matrix(), class DistanceMatrix` uses either `sklearn.neighbors` or `pyflann` – a state of the art Fast Approximate Nearest Neighbor search algorithm (default)
- `adjacency_matrix()` preprocesses the distance matrix
- `graph_laplacian` computes a variety of undirected Laplacians

`Mmani.embedding.rmetric.py` estimates the embedding distortion at each point

`Mmani.embedding.embed_with_rmetric.py` pipelines the above to produce a complete embedding from data

`Mmani.embedding.spectral_embedding.py` re-implementation of Laplacian Eigenmaps/Diffusion Maps

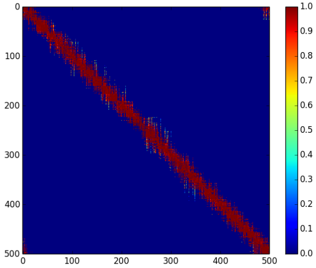
`Mmani.tests`, `Mmani.benchmarks`

Python package `Mmani` Megamanifold – a comparison with `sklearn.manifold`

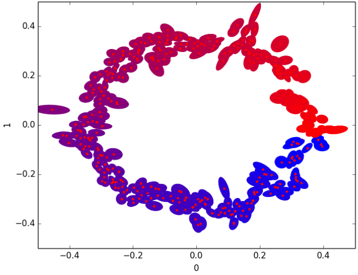
- Statistical/methodological novelty
  - implements recent advances in the statistical understanding of manifold learning (radius based neighborhoods , consistent graph Laplacians , Riemannian metric (stretch) estimation)
- Designed for performance
  - sparse representation as default
  - incorporates state of the art Fast Approximate Nearest Neighbor search algorithm (can handle Billions of data points)
  - exposes/caches intermediate states (e.g. data set index, distances, Laplacian, its eigenvectors)
  - lazy evaluation in postprocessing (specifically the `rmetric`/"stretch" evaluation
- Designed for extensions like neighborhood size estimation, dimension estimation

Does it work? Artificial data results

similarity matrix

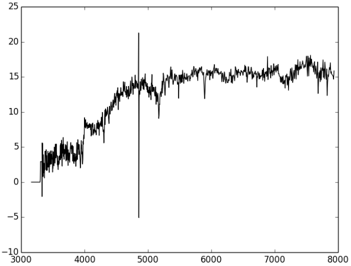


embedding with distortion



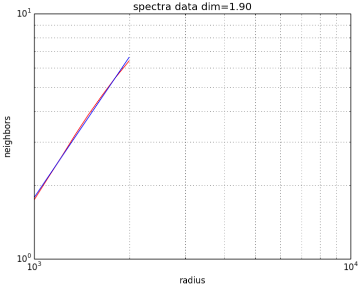
Manifold learning for SDSS Galaxy Spectra data
`astrodemo`

A spectrum

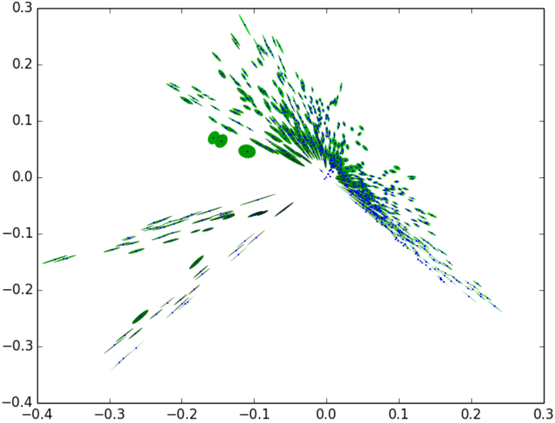


$n = 4000$ spectra $\times$ $D = 1000$ dimensions

Estimating the intrisic dimension

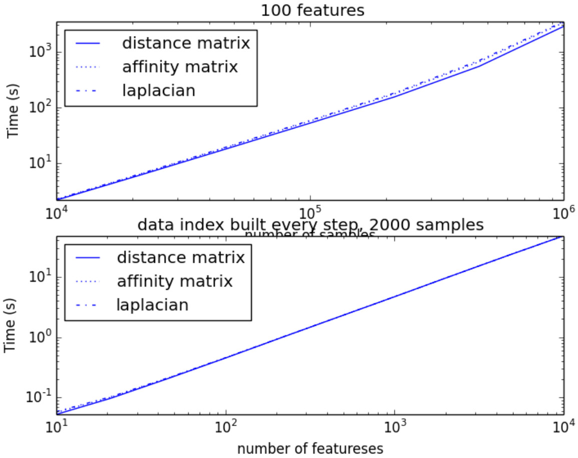


Embedding of SDSS Galaxy Spectra



(by Laplacian Eigenmaps)

Does it scale?



Conclusion/Future work

- Benchmarking in progress
- closer integration with `sklearn.manifold`
- Automatic estimation of dimension, neighborhood radius
- Directed embedding

- Incubator was a very productive, intense experience

Thank you, e-Science staff!