Locally Linear Landmarks for large-scale manifold learning

Max Vladymyrov and **Miguel Á. Carreira-Perpiñán**

Electrical Engineering and Computer Science

University of California, Merced

http://eecs.ucmerced.edu

Spectral dimensionality reduction methods

Given high-dim data points $\mathbf{Y}_{D\times N} = (\mathbf{y}_1, \ldots, \mathbf{y}_N)$, find low-dim points $\mathbf{X}_{d\times N} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$, with $d \ll D$, as the solution of the following optimization problem:

$$\min_{\mathbf{X}} \operatorname{tr}\left(\mathbf{X}\mathbf{A}\mathbf{X}^T\right) \text{ s.t. } \mathbf{X}\mathbf{B}\mathbf{X}^T = \mathbf{I}.$$

- $\mathbf{A}_{N\times N}$: symmetric psd, contains information about the similarity between pairs of data points $(\mathbf{y}_n, \mathbf{y}_m)$

  User parameters: number of neighbours k, bandwidth σ, etc.
- $\mathbf{B}_{N\times N}$: symmetric pd (usually diagonal), sets the scale of $\mathbf{X}$.

Examples:

- Laplacian eigenmaps: $\mathbf{A} =$ graph Laplacian
  Also: spectral clustering
- Isomap: $\mathbf{A} =$ shortest-path distances
- Kernel PCA, multidimensional scaling, LLE, etc.

Computational solution with large-scale problems

Solution: $\mathbf{X} = \mathbf{U}^T \mathbf{B}^{-\frac{1}{2}}$, where $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_d)$ are the d trailing eigenvectors of the $N \times N$ matrix $\mathbf{C} = \mathbf{B}^{-\frac{1}{2}} \mathbf{A} \mathbf{B}^{-\frac{1}{2}}$. With large N, solving this eigenproblem is infeasible even if $\mathbf{A}$ and $\mathbf{B}$ are sparse.

Goal of this work: a fast, approximate solution for the embedding $\mathbf{X}$.

Applications:

- When N is so large that the direct solution is infeasible.
- To select hyperparameters (k, $\sigma \ldots$) efficiently even if N is not large
  since a grid search over these requires solving the eigenproblem many times.
- As an out-of-sample extension to spectral methods.

Computational solution with large-scale problems (cont

The Nyström method is the standard way to approximate large-scale eigenproblems. Essentially, an out-of-sample formula:

1. Solve the eigenproblem for a subset of points (landmarks) $\widetilde{\mathbf{Y}} = \widetilde{\mathbf{y}}_1, \ldots, \widetilde{\mathbf{y}}_L$, where $L \ll N$.
2. Predict $\mathbf{x}$ for any other point $\mathbf{y}$ through an interpolation formula:

$$x_k = \frac{\sqrt{L}}{\lambda_k} \sum_{l=1}^{L} K(\mathbf{y}, \widetilde{\mathbf{y}}_l)\, u_{lk} \qquad k = 1, \ldots, d$$

Problems:

- Needs to know the interpolation kernel $K(\mathbf{y}, \mathbf{y}')$ (sometimes tricky).
- It only uses the information in $\mathbf{A}$ about the landmarks, ignoring the non-landmarks. This requires using many landmarks to represent the data manifold well. If too few landmarks are used:
  - Bad solution for the landmarks $\widetilde{\mathbf{X}} = \widetilde{\mathbf{x}}_1 \ldots, \widetilde{\mathbf{x}}_L$
  - . . . and bad prediction for the non-landmarks.

Locally Linear Landmarks (LLL)

Assume each projection is a locally linear function of the landmarks:

$$\mathbf{x}_n = \sum_{l=1}^{L} z_{ln}\widetilde{\mathbf{x}}_l,\ n = 1, \dots, N \quad \Rightarrow \quad \mathbf{X} = \widetilde{\mathbf{X}}\mathbf{Z}$$

Solving the original eigenproblem of $N \times N$ with this constraint results in a reduced eigenproblem of the same form but of $L \times L$ on $\widetilde{\mathbf{X}}$:

$$\min_{\widetilde{\mathbf{X}}} \operatorname{tr}\left(\widetilde{\mathbf{X}}\widetilde{\mathbf{A}}\widetilde{\mathbf{X}}^T\right) \text{ s.t. } \widetilde{\mathbf{X}}\widetilde{\mathbf{B}}\widetilde{\mathbf{X}}^T = \mathbf{I}$$

with reduced affinities $\widetilde{\mathbf{A}} = \mathbf{Z}\mathbf{A}\mathbf{Z}^T$, $\widetilde{\mathbf{B}} = \mathbf{Z}\mathbf{B}\mathbf{Z}^T$. After $\widetilde{\mathbf{X}}$ is found, the non-landmarks are predicted as $\mathbf{X} = \widetilde{\mathbf{X}}\mathbf{Z}$ (out-of-sample mapping).

Advantages over Nyström's method:

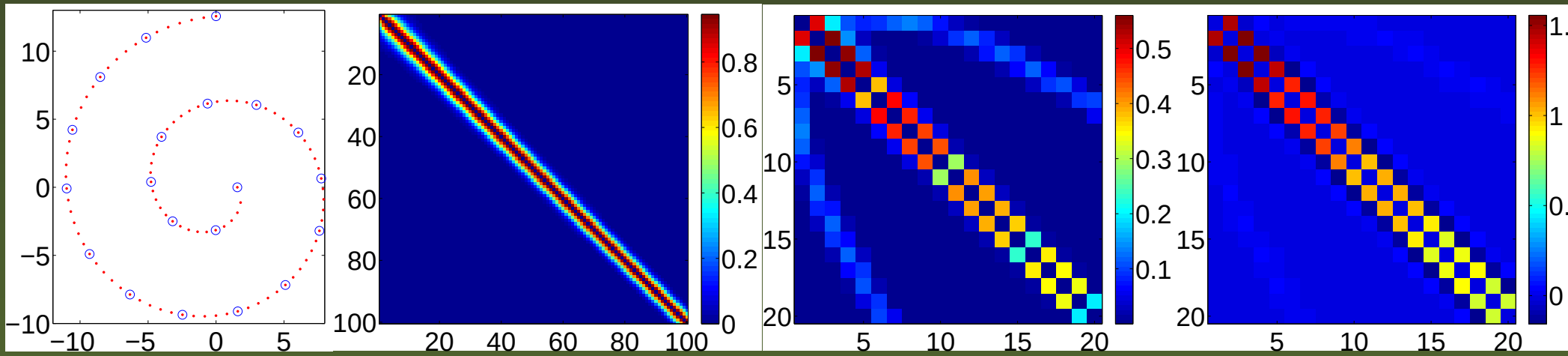
- The reduced affinities $\widetilde{\mathbf{A}} = \mathbf{Z}\mathbf{A}\mathbf{Z}^T$ involve the entire dataset and contain much more information about the manifold that the landmark–landmark affinities, so fewer landmarks are needed.
- Solving this smaller eigenproblem is faster.
- The out-of-sample mapping requires less memory and is faster.

LLL: reduced affinities

Affinities between landmarks:

- Nyström (original affinities): $\mathbf{A} \Rightarrow a_{ij} \Rightarrow$ path i—j.
- LLL (reduced affinities): $\widetilde{\mathbf{A}} = \mathbf{Z}\mathbf{A}\mathbf{Z}^T \Rightarrow \widetilde{a}_{ij} = \sum_{n,m=1}^{N} z_{in} a_{nm} z_{jm} \Rightarrow$ paths i—n—m—j $\forall n, m$.
  So landmarks i and j can be farther apart and still be connected along the manifold.

Dataset	Affinities between… All points	Landmarks (Nyström)	Landmarks (LLL)



LLL: construction of the weight matrix $\mathbf{Z}$

- Most embedding methods seek to preserve local neighbourhoods between the high- and low-dim spaces.
- Hence, if we assume that a point may be approximately linearly reconstructed from its nearest landmarks in high-dim space:

$$\mathbf{y}_n \approx \sum_{l=1}^{L} z_{ln}\widetilde{\mathbf{y}}_l,\ n = 1, \ldots, N \quad \Rightarrow \quad \mathbf{Y} \approx \widetilde{\mathbf{Y}}\mathbf{Z}$$

  the same will happen in low-dim space: $\mathbf{X} \approx \widetilde{\mathbf{X}}\mathbf{Z}$.
- We consider only the K_Z nearest landmarks, $d+1 \leq K_Z \leq L$. So:
  1. Find the K_Z nearest landmarks of each data point.
  2. Find their weights as $\mathbf{Z} = \arg\min_{\mathbf{Z}} \|\mathbf{Y} - \widetilde{\mathbf{Y}}\mathbf{Z}\|^2$ s.t. $\mathbf{Z}^T\mathbf{1} = \mathbf{1}$.

  These are the same weights used by Locally Linear Embedding (LLE) (Roweis & Saul 2000).
- This implies the out-of-sample mapping (projection for a test point) is globally nonlinear but locally linear: $\mathbf{x} = \mathbf{M}(\mathbf{y})\,\mathbf{y}$ where matrix $\mathbf{M}(\mathbf{y})$ of $d \times D$ depends only on the set of nearest landmarks of $\mathbf{y}$.

LLL: computational complexity

- We assume the affinity matrix is given.

  If not, use approximate nearest neighbours to compute it.

- Time: the exact runtimes depend on the sparsity structure of the affinity matrix $\mathbf{A}$ and the weight matrix $\mathbf{Z}$, but in general the time is dominated by:
  - LLL: finding the nearest landmarks for each data point
  - Nyström: computing the out-of-sample mapping for each data point

  and this is $\mathcal{O}(NLD)$ in both cases.
  Note LLL uses fewer landmarks to achieve the same error.

- Memory: LLL and Nyström are both $\mathcal{O}(Ld)$.

LLL: user parameters

- Location of landmarks: a random subset of the data works well.

  Refinements such as k-means improve a little with small L but add runtime.

- Total number of landmarks L: as large as possible.

  The more landmarks, the better the result.

- Number of neighbouring landmarks K_Z for the projection matrix $\mathbf{Z}$: $K_Z \gtrsim d+1$, where d is the dimension of the latent space.

  Each point should be a locally linear reconstruction of its K_Z nearest landmarks:

  - K_Z landmarks span a space of dimension $K_Z - 1 \Rightarrow K_Z \geq d+1$.
  - Having more landmarks protects against occasional collinearities, but decreases the locality.

LLL: algorithm

Given spectral problem $\min_{\mathbf{X}} \operatorname{tr}\left(\mathbf{X}\mathbf{A}\mathbf{X}^T\right)$ s.t. $\mathbf{X}\mathbf{B}\mathbf{X}^T = \mathbf{I}$ for dataset $\mathbf{Y}$:

1. Choose the number of landmarks L, as high as your computer can support, and $K_Z \gtrsim d+1$.
2. Pick L landmarks $\widetilde{\mathbf{y}}_1, \ldots, \widetilde{\mathbf{y}}_L$ at random from the dataset.
3. Compute local reconstruction weights $\mathbf{Z}_{L\times N}$ for each data point wrt its nearest K_Z landmarks:

$$\mathbf{Z} = \arg\min_{\mathbf{Z}} \|\mathbf{Y} - \widetilde{\mathbf{Y}}\mathbf{Z}\|^2 \text{ s.t. } \mathbf{Z}^T\mathbf{1} = \mathbf{1}.$$

4. Solve reduced eigenproblem

$$\min_{\widetilde{\mathbf{X}}} \operatorname{tr}\left(\widetilde{\mathbf{X}}\widetilde{\mathbf{A}}\widetilde{\mathbf{X}}^T\right) \text{ s.t. } \widetilde{\mathbf{X}}\widetilde{\mathbf{B}}\widetilde{\mathbf{X}}^T = \mathbf{I} \text{ with } \widetilde{\mathbf{A}} = \mathbf{Z}\mathbf{A}\mathbf{Z}^T,\ \widetilde{\mathbf{B}} = \mathbf{Z}\mathbf{B}\mathbf{Z}^T$$

   for the landmark projections $\widetilde{\mathbf{X}}$.
5. Predict non-landmarks with out-of-sample mapping $\mathbf{X} = \widetilde{\mathbf{X}}\mathbf{Z}$.

Experiments: Laplacian eigenmaps

We apply LLL to Laplacian eigenmaps (LE) (Belkin & Niyogi, 2003):

- $\mathbf{A}$: graph Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{W}$ for a Gaussian affinity matrix $\mathbf{W} = \left(\exp\left(-\|(\mathbf{y}_n - \mathbf{y}_m)/\sigma\|^2\right)\right)_{nm}$ on k-nearest-neighbour graph.
- $\mathbf{B}$: degree matrix $\mathbf{D} = \operatorname{diag}\left(\sum_{m=1}^{N} w_{nm}\right)$.

$$\min_{\mathbf{X}} \operatorname{tr}\left(\mathbf{X}\mathbf{L}\mathbf{X}^T\right) \text{ s.t. } \mathbf{X}\mathbf{D}\mathbf{X}^T = \mathbf{I},\ \mathbf{X}\mathbf{D}\mathbf{1} = \mathbf{0}.$$

LLL's reduced eigenproblem has $\widetilde{\mathbf{A}} = \mathbf{Z}\mathbf{L}\mathbf{Z}^T, \widetilde{\mathbf{B}} = \mathbf{Z}\mathbf{D}\mathbf{Z}^T$.

We compare LLL with 3 baselines:

1. Exact LE runs LE on the full dataset.
   Ground-truth embedding, but the runtime is large.

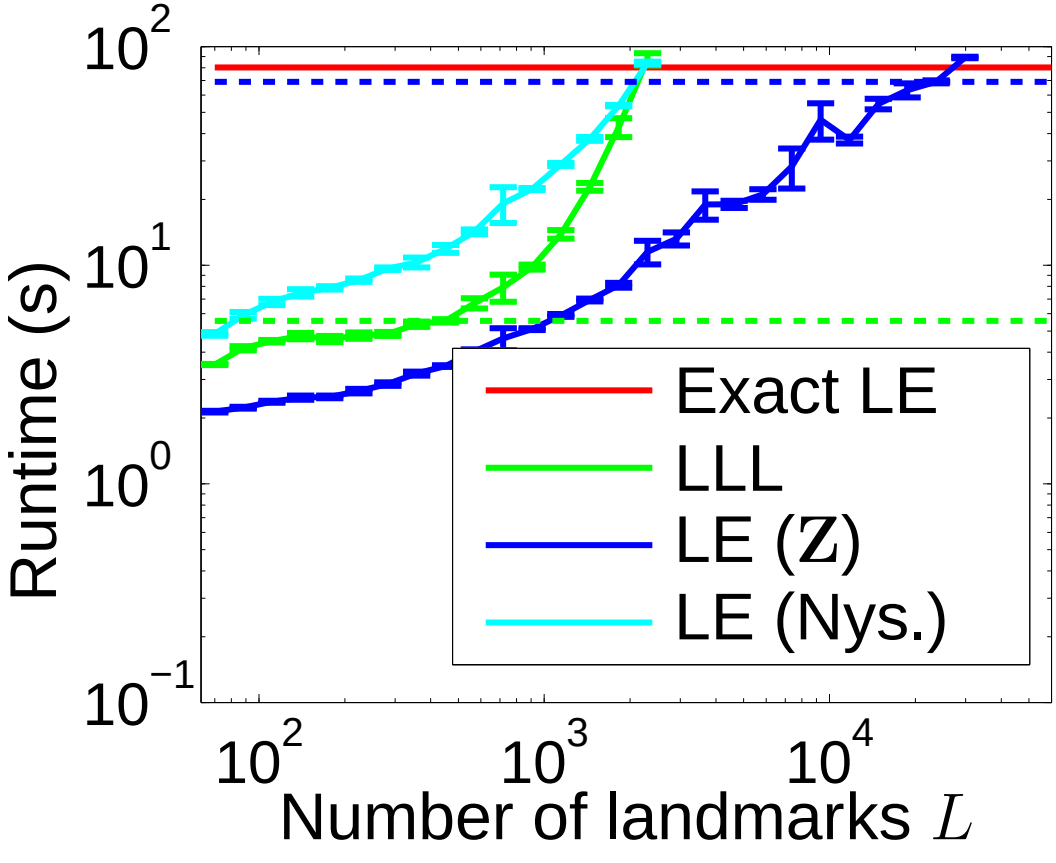
Landmark LE runs LE only on a set of landmark points. Once their projection is found, the rest of the points are embedded using:

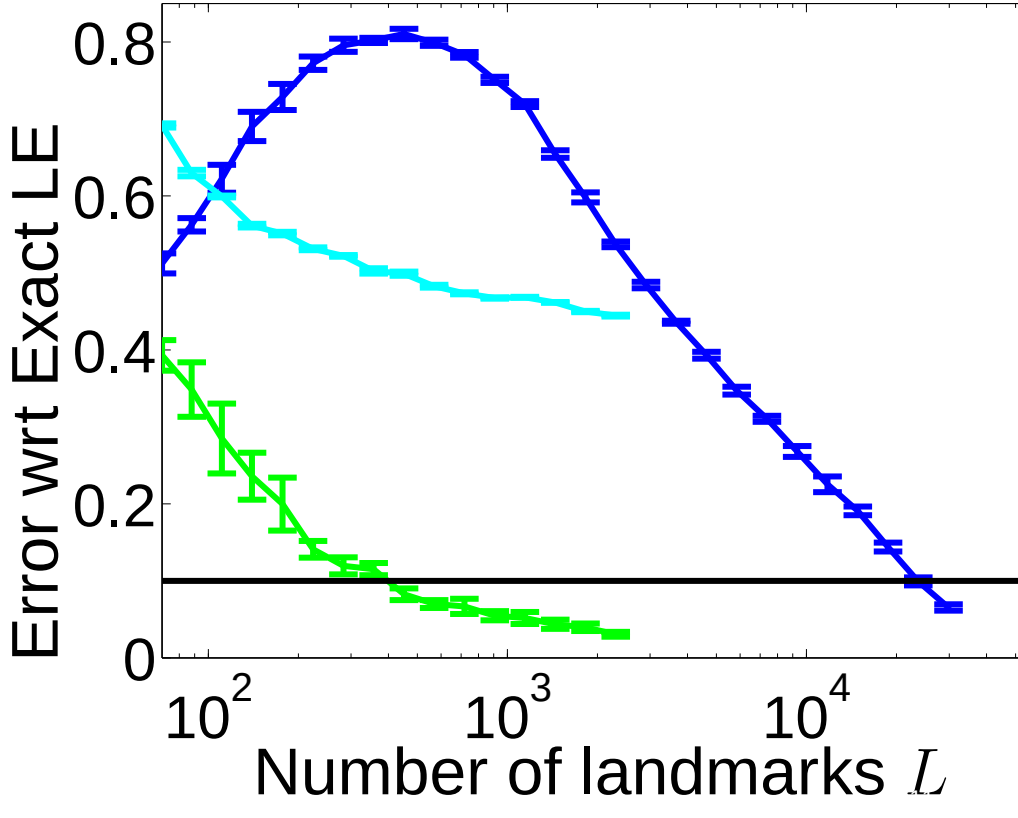
2. LE(Nys.): out-of-sample mapping using Nyström's method.
3. LE($\mathbf{Z}$): out-of-sample mapping using reconstruction weights.

Experiments: effect of the number of landmarks

- $N = 60\,000$ MNIST digits, project to $d = 50$, $K_Z = 50$ landmarks.
- Choose landmarks randomly, from $L = 50$ to $L = N$.

LLL produces an embedding with quite lower error than Nyström's method for the same number of landmarks L.

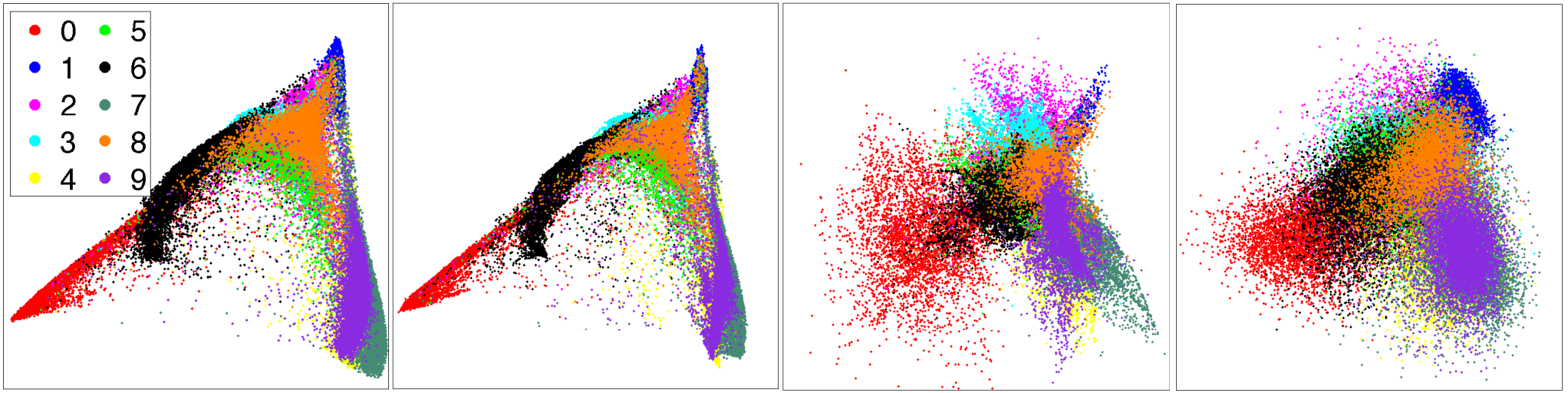




Experiments: effect of the number of landmarks (cont.)

Embeddings after 5 s runtime:

Exact LE, 80 s. LLL, 5 s. LE ($\mathbb{Z}$), 5 s. LE (Nys.), 5 s.

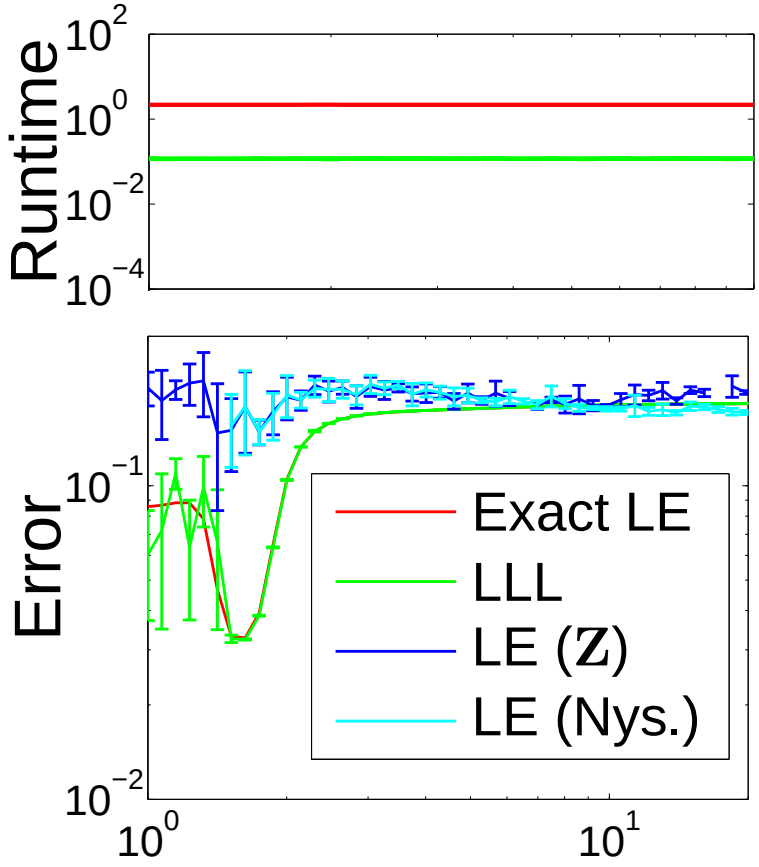


Experiments: model selection in Swiss roll dataset

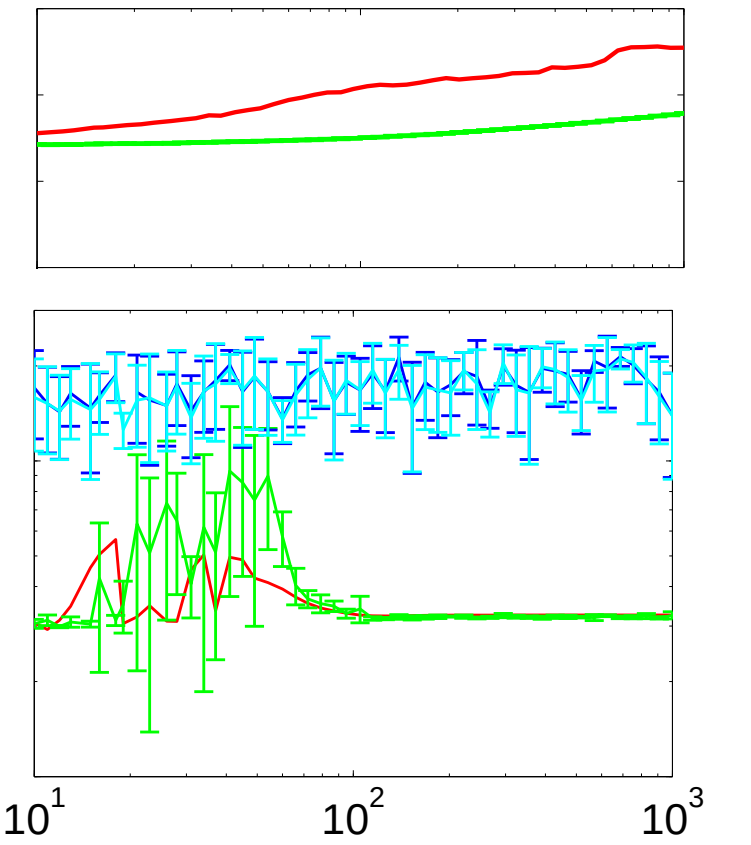
Vary the hyperparameters of Laplacian eigenmaps (affinity bandwidth σ, k-nearest-neighbour graph) and compute for each combination the relative error of the embedding $\mathbf{X}$ wrt the ground truth on $N = 4\,000$ points using $L = 300$ landmarks.

Matrix $\mathbf{Z}$ need only be computed once.

The minima of the model selection error curves for LLL and Exact LE align well.



Bandwidth σ (for $k = 150$)



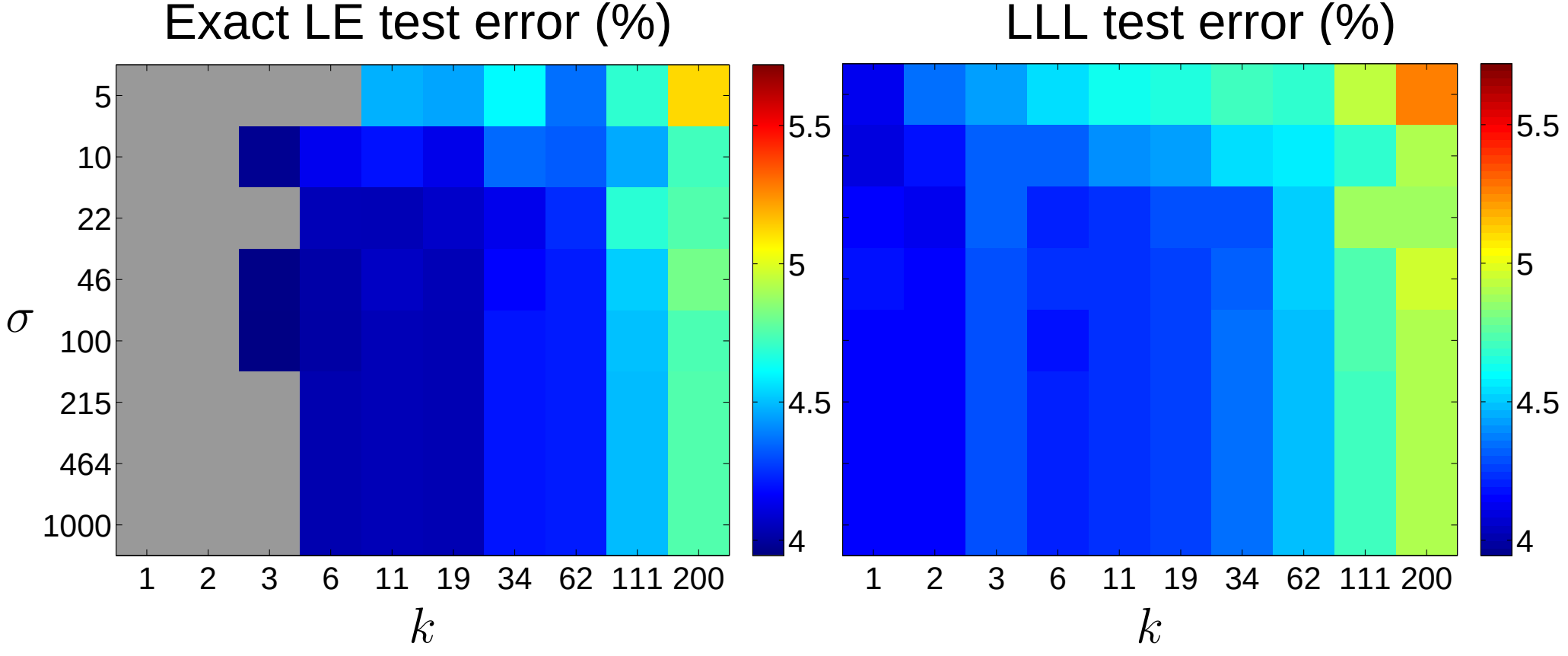
neighbours k (for $\sigma = 1.6$)

Experiments: model selection in classification task

Find hyperparameters to achieve low 1-nn classification error in MNIST.

- $50\,000$ points as training, $10\,000$ as test, $10\,000$ as out-of-sample.
- Project to $d = 500$ using LLL ($K_Z = 50$, $L = 1\,000$).

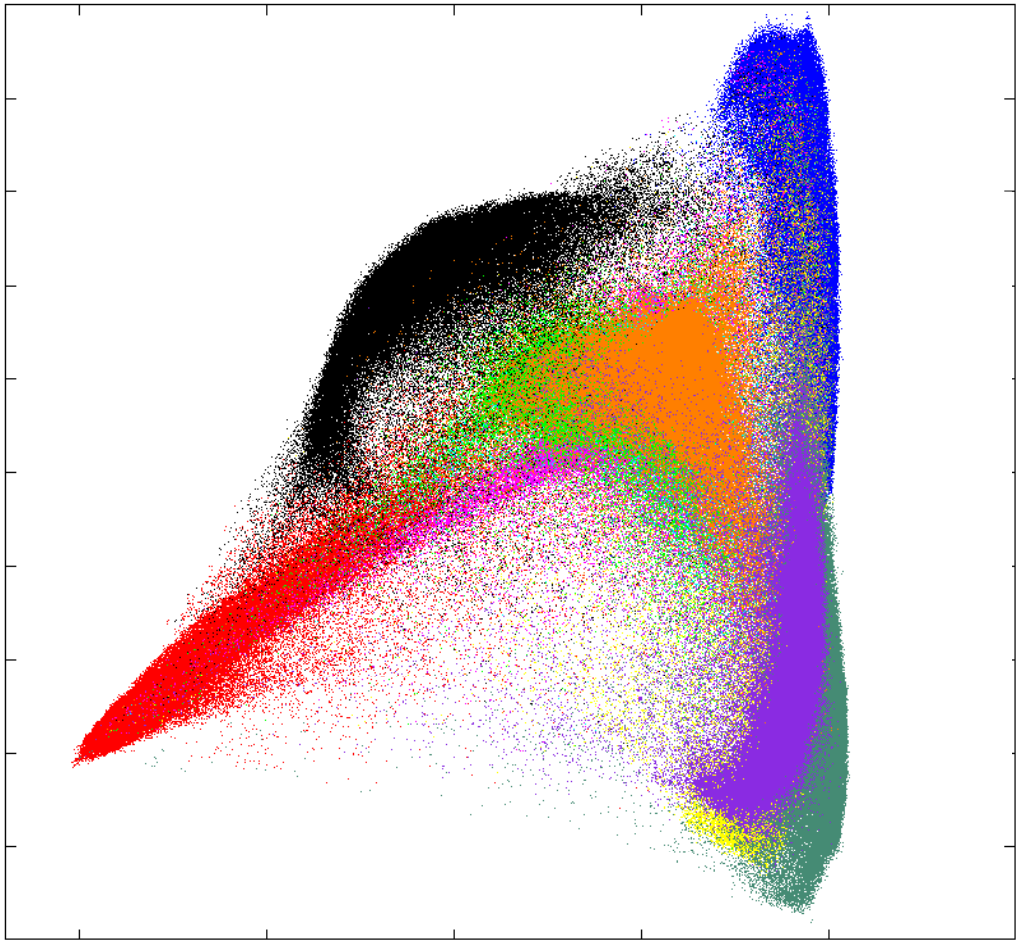
In runtime, LLL is 15–$40\times$ faster than Exact LE. The model selection curves align well, except `eigs` in Exact LE fails to converge for small k.



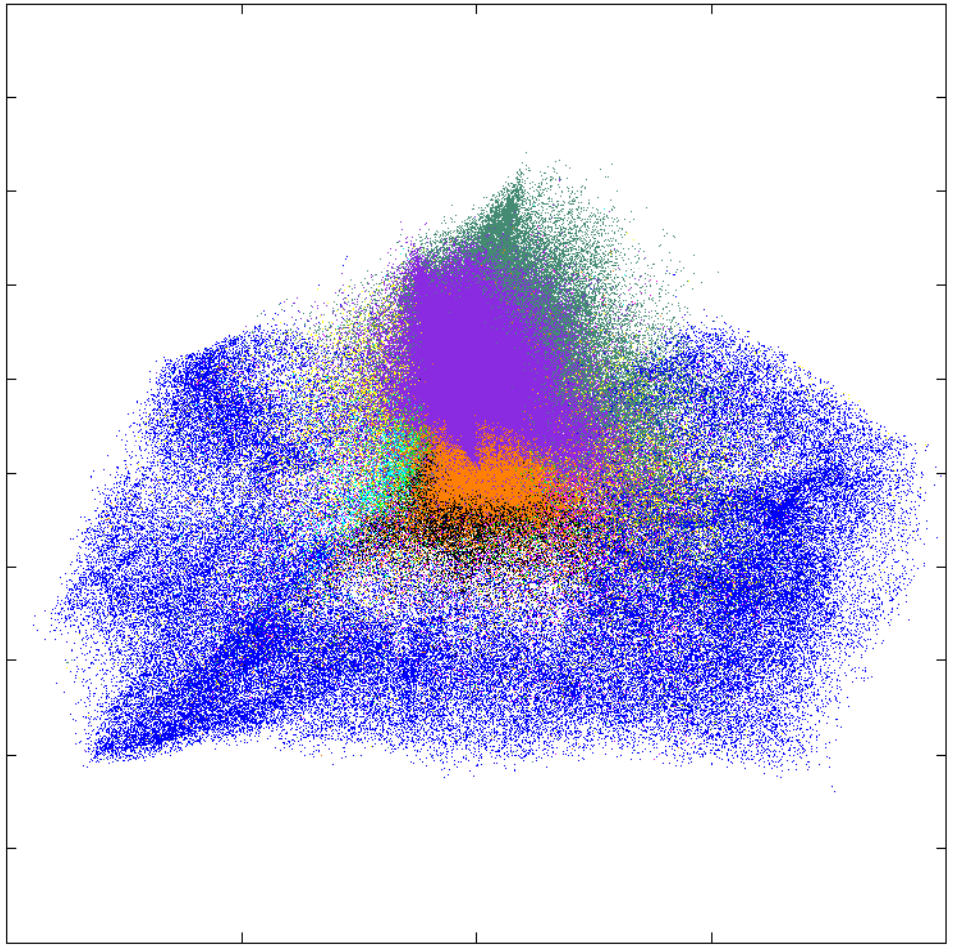
Experiments: large-scale dataset

- $N = 1\,020\,000$ points from infiniteMNIST.
- $L = 10^4$ random landmarks (1%), $K_Z = 5$.

LLL (18' runtime)



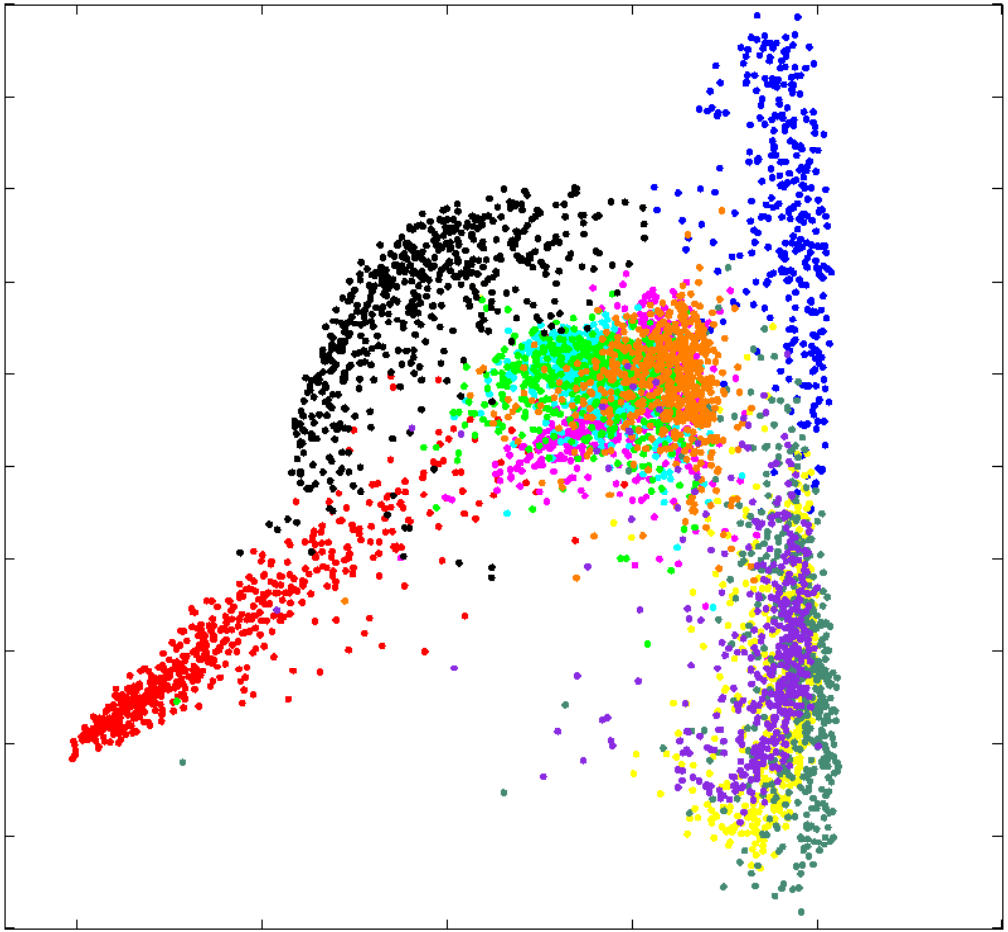
LE($\mathbf{Z}$)



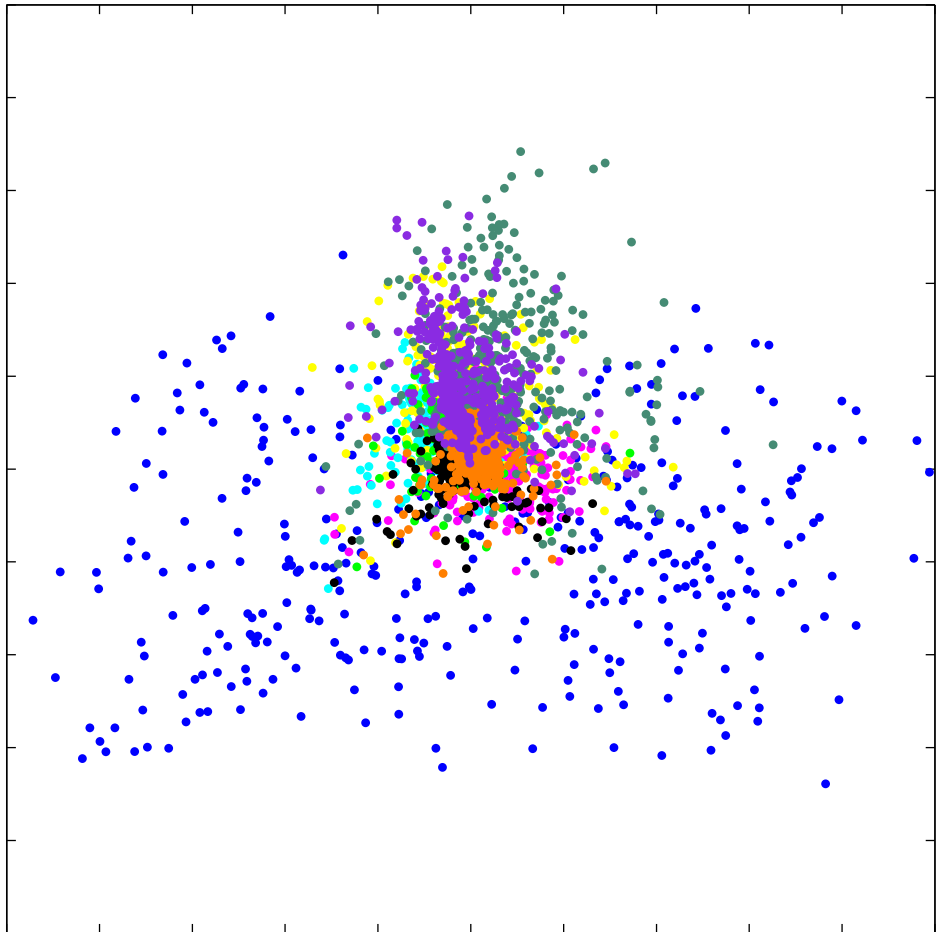
Experiments: large-scale dataset (cont.)

The reason for the improved result with LLL is that it uses better affinities, so the landmarks are better projected.

Landmarks with
LLL reduced affinities



Landmarks with
original affinities



Conclusions

- The basic reason why LLL improves over Nyström's method is that, by using the entire dataset, it constructs affinities that better represent the manifold for the same number of landmarks.
- Hence, it requires fewer landmarks, and is faster at training and test time.
- It applies to any spectral method.

  No need to work out a special kernel as in Nyström's method.
- LLL can be used:
  - to find a fast, approximate embedding of large dataset
  - to do fast model selection
  - as an out-of-sample extension to spectral methods.
- Matlab code: `http://eecs.ucmerced.edu`.

Partially supported by NSF CAREER award IIS–0754089.