LINEAR-TIME TRAINING OF NONLINEAR LOW-DIMENSIONAL EMBEDDINGS

Max Vladymyrov and **Miguel Á. Carreira-Perpiñán**, UC Merced, USA

1 Abstract

Nonlinear embeddings such as stochastic neighbor embedding or the elastic embedding achieve better results than spectral methods but require an expensive, nonconvex optimization, where the objective function and gradient are quadratic on the sample size. We address this bottleneck by formulating the optimization as an N-body problem and using fast multipole methods (FMMs) to approximate the gradient in linear time. We study the effect, in theory and experiment, of approximating gradients in the optimization and show that the expected error is related to the mean curvature of the objective function, and that gradually increasing the accuracy level in the FMM over iterations leads to a faster training. When combined with standard optimizers, such as gradient descent or L-BFGS, the resulting algorithm beats the $\mathcal{O}(N \log N)$ Barnes-Hut method and achieves reasonable embeddings for one million points in three hours.

2 Nonlinear Embedding Methods

Given the symmetric nonnegative affinity matrix $\mathbf{W}$ defined for a high-d data set $\mathbf{Y}_{D\times N} = (\mathbf{y}_1, \dots, \mathbf{y}_N)$ nonlinear embeddings (NLE) find low-d projection $\mathbf{X}_{d\times N} = (\mathbf{x}_1, \dots, \mathbf{x}_N)$ by minimizing: $E(\mathbf{X}; \lambda) = E^+(\mathbf{X}) + \lambda E^-(\mathbf{X})$, with $\lambda \geq 0$. For example, in the Elastic Embedding algorithm the objective function and the gradient are given by:

$$E_{\text{EE}}(\mathbf{X}) = \sum_{n,m=1}^{N} w_{nm} \left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2 + \lambda \sum_{n=1}^{N} S(\mathbf{x}_n),$$

$$\mathbf{G}_{\text{EE}}(\mathbf{X}) = 4\mathbf{X}\mathbf{L} - 4\lambda\Big(\mathbf{X}\operatorname{diag}\left(S(\mathbf{X})\right) + S^x(\mathbf{X})\Big).$$

where

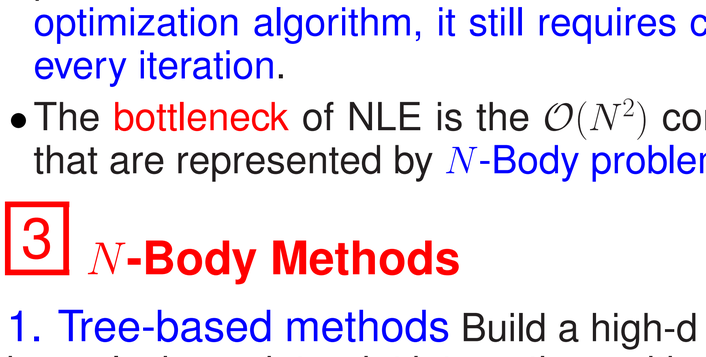
$$S(\mathbf{x}_n) = \sum_{m=1}^{N} \exp(-\left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2) \text{ and } S^x(\mathbf{x}_n) = \sum_{m=1}^{N} \mathbf{x}_m \exp(-\left\|\mathbf{x}_n - \mathbf{x}_m\right\|^2).$$

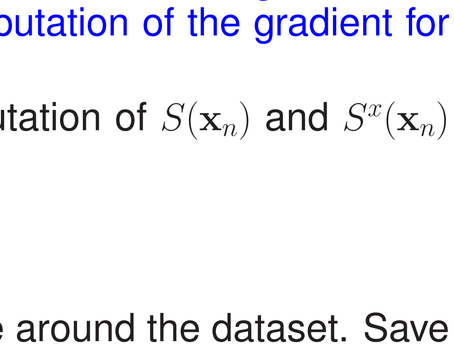
- Vladymyrov and Carreira-Perpiñán (2012) achieve the best descent per iteration for NLE methods. However, no matter how good is the optimization algorithm, it still requires computation of the gradient for every iteration.
- The bottleneck of NLE is the $\mathcal{O}(N^2)$ computation of $S(\mathbf{x}_n)$ and $S^x(\mathbf{x}_n)$ that are represented by N-Body problem.

3 N-Body Methods

1. Tree-based methods Build a high-d tree around the dataset. Save by replacing point-point interactions with node-point or node-node ones. Complexity $\mathcal{O}(N \log N)$. Focus on Barnes-Hut algorithm:

- *Preprocess:* build a quadtree around the dataset saving the center of mass $\mathbf{c}$ for every cell.
- *Query:* for every point $\mathbf{x}$ traverse down the tree computing the size of the current cell l and the distance to its centroid D. If $l/D < \theta \Rightarrow$ use an interaction between $\mathbf{x}$ and $\mathbf{c}$ for all the points in that cell.
- *Accuracy:* controlled by θ. Bigger values $\Rightarrow$ larger speed-up, also larger error.





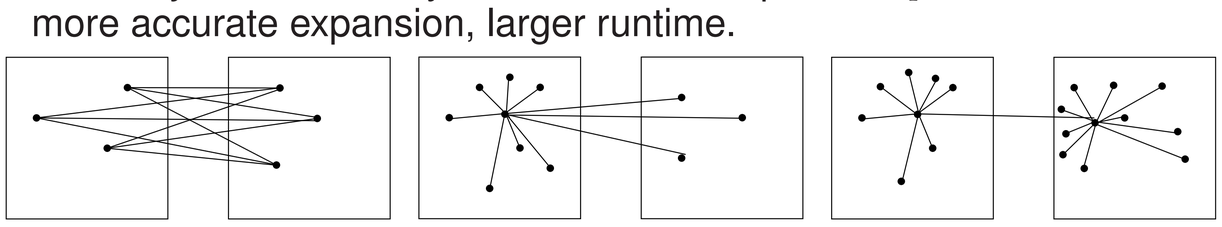
2. Fast Multipole Methods Decouple $\mathbf{x}_n$ and $\mathbf{x}_m$ using a series expansion:

$$Q(\mathbf{x}_n) = \sum_{m=1}^{N} \mathbf{q}_m K(\left\|(\mathbf{x}_n - \mathbf{x}_m)/\sigma\right\|^2) \approx \sum_{\boldsymbol{\alpha}\geq 0} \left(f_{\boldsymbol{\alpha}}(\mathbf{x}_n) \sum_{m=1}^{N} g_{\boldsymbol{\alpha}}(\mathbf{x}_m)\right)$$

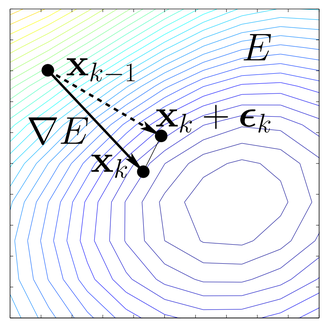
f and g are some functions and $\boldsymbol{\alpha}$ is a multi-index notation $\boldsymbol{\alpha} \geq 0 \Rightarrow \alpha_1, \dots, \alpha_d \geq 0$.

Computation complexity is $\mathcal{O}(N)$.

- *Preprocess:* divide $\mathbf{X}$ into boxes. Many points $\Rightarrow$ expand around the center of mass.
- *Query:* (1) ignore interactions between distant boxes; (2) many points per box $\Rightarrow$ use center of mass, otherwise – compute exactly.
- *Accuracy:* controlled by the number of expansion p. More terms $\Rightarrow$ more accurate expansion, larger runtime.



4 Optimization with inexact gradient



- Each iteration k incurs a small error $\boldsymbol{\epsilon}_k$.
- Model $\boldsymbol{\epsilon}_k$ as zero-mean Gaussian noise $\mathcal{N}(\mathbf{0}, \xi^2\mathbf{I})$ (assume non systematic error).
- Variance ξ^2 is a model parameter that represents the accuracy of the approximation.
- Mean of the error $\langle E(\mathbf{X} + \boldsymbol{\epsilon}) - E(\mathbf{X})\rangle = \frac{1}{2}\xi^2 \operatorname{tr}\left(\nabla^2 E(\mathbf{X})\right) + \mathcal{O}(\xi^4)$.

We can make the following qualitative prediction:

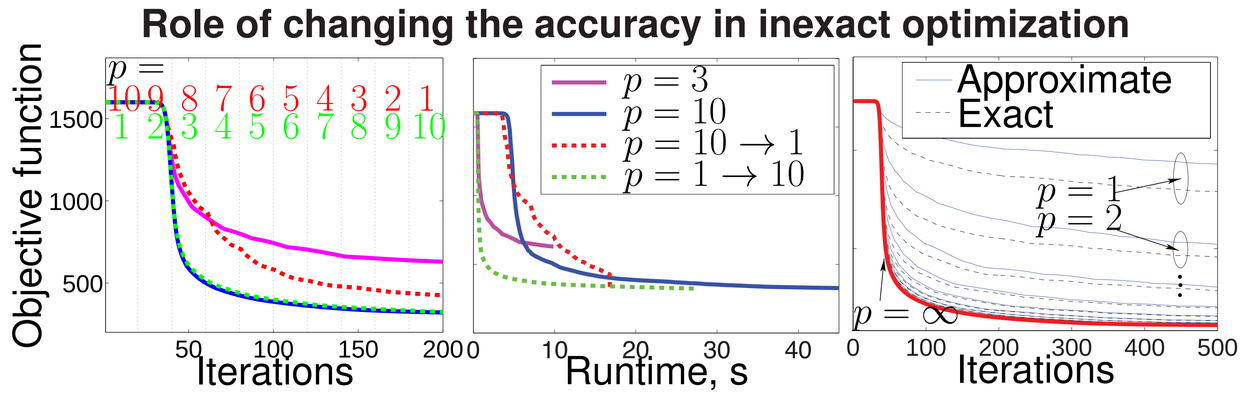
- Adding noise is beneficial if the mean curvature $\operatorname{tr}\left(\nabla^2 E(\mathbf{X})\right)$ is negative.
- Near the minimizer the mean curvature is positive $\Rightarrow$ no gains from the approximation.

We suggest starting with relatively low accuracy and increasing it progressively:

- Cheap initial iterations.
- Far from the minimizer $\Rightarrow$ benefit from the noise whenever the mean curvature is negative.
- Analogous to simulated annealing $\Rightarrow$ increasing the accuracy avoids wandering behavior.

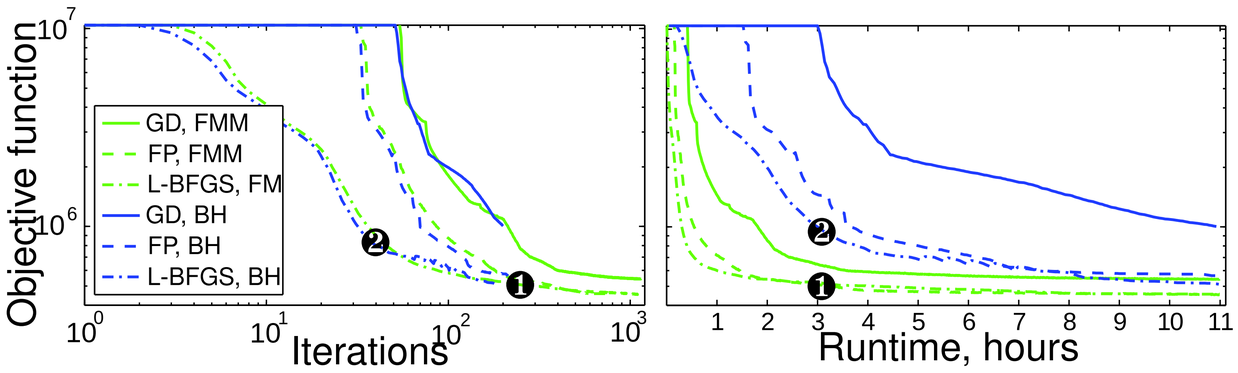
Input: (1) initial $\mathbf{X}_0$,
(2) sparse affinities $\mathbf{W}$,
(3) non-decreasing accuracy $(p_0, p_1 \dots)$,
(4) step size η.
for $k = 0$ **to** `maxit` **do**
  eval. approx. gradient $\mathbf{G}_k$
  eval. direction $\mathbf{P}_k$.
  $\mathbf{X}_{k+1} = \mathbf{X}_k + \eta\mathbf{P}_k$.
end for

Role of changing the accuracy in inexact optimization



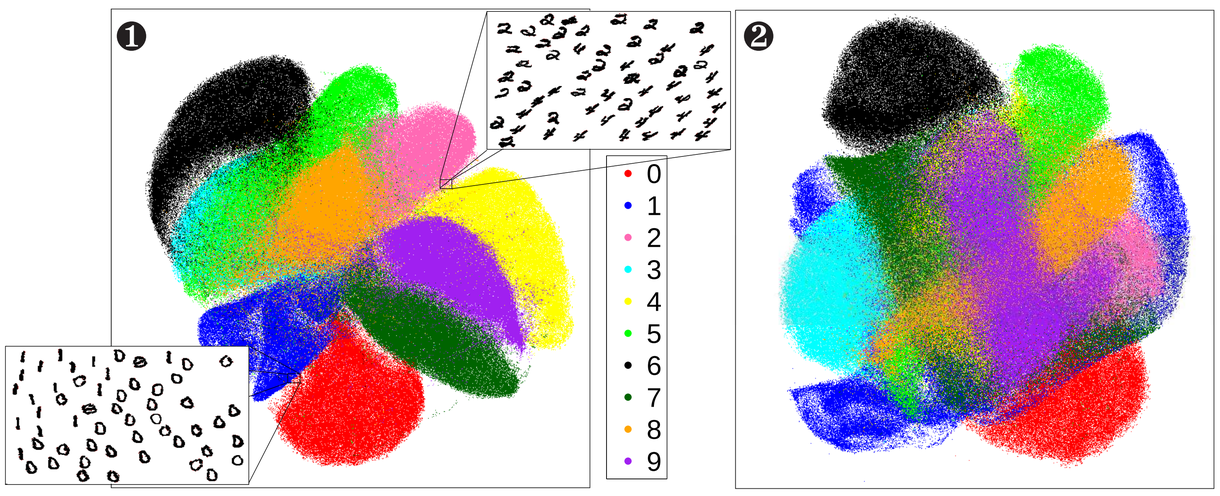
5 Experiments

- $1\,020\,000$ points from infiniteMNIST.
- Elastic Embedding algorithm ($\lambda = 10^{-4}$) optimized with gradient descent (GD), fixed point iterations (FP) and L-BFGS.
- Fixed step size. The accuracy grows from $p = 1$ to 10 for the first 100 iterations.



FGT, L-BFGS, 3 hours
$E = 521\,666,\ 221$ iter.

BH, L-BFGS, 3 hours
$E = 1\,079\,357{,}32$ iter.



6 Conclusions

- N-Body methods address the main bottleneck of nonlinear embedding methods: quadratic cost of the objective function and the gradient.
- Fast Multipole Methods are more beneficial than Barnes-Hut both theoretically and empirically ($4 - 7\times$ speedup for million size dataset).
- Gradual increase of the accuracy parameter is advisable.
- MATLAB code: `http://eecs.ucmerced.edu`.