NONLINEAR LOW-DIMENSIONAL REGRESSION USING AUXILIARY COORDINATES.

Weiran Wang and Miguel Á. Carreira-Perpiñán.
EECS, University of California, Merced.

1 Abstract

When doing regression with inputs and outputs that are high-dimensional, it often makes sense to reduce the dimensionality of the inputs before mapping to the outputs. We propose a method where both the dimensionality reduction and the regression mapping can be nonlinear and are estimated jointly. Our key idea is to define an objective function where the low-dimensional coordinates are free parameters, in addition to the dimensionality reduction and the regression mapping. This has the effect of decoupling many groups of parameters from each other, affording a more effective optimization, and to use a good initialization from other methods.

Work funded by NSF CAREER award IIS–0754089.

2 Low-dimensional regression using auxiliary coordinates

Given a training set $\mathbf{X}_{D_x \times N}$ and $\mathbf{Y}_{D_y \times N}$, instead of directly optimizing

$$E_1(\mathbf{F}, \mathbf{g}) = \sum_{n=1}^{N} \|\mathbf{y}_n - \mathbf{g}(\mathbf{F}(\mathbf{x}_n))\|^2 + \lambda_{\mathbf{g}} R(\mathbf{g}) + \lambda_{\mathbf{F}} R(\mathbf{F})$$

with $\lambda_{\mathbf{F}}, \lambda_{\mathbf{g}} \geq 0$ for dimension reduction mapping $\mathbf{F}$ and regression mapping $\mathbf{g}$, we let the low-dimensional coordinates $\mathbf{Z}_{D_z \times N} = (\mathbf{z}_1, \dots, \mathbf{z}_N)$ be independent, auxiliary parameters to be optimized over, and unfold the squared error into two terms that decouple given $\mathbf{Z}$:

$$E_2(\mathbf{F}, \mathbf{g}, \mathbf{Z}) = \textstyle\sum_{n=1}^{N} \|\mathbf{y}_n - \mathbf{g}(\mathbf{z}_n)\|^2 + \lambda_{\mathbf{g}} R(\mathbf{g}) + \sum_{n=1}^{N} \|\mathbf{z}_n - \mathbf{F}(\mathbf{x}_n)\|^2 + \lambda_{\mathbf{F}} R(\mathbf{F}).$$

Now, every squared error involves only a shallow mapping, compared to deeper nesting in the function $\mathbf{g} \circ \mathbf{F}$ that leads to ill-conditioning. We apply the following alternating optimization procedure to solve the problem.

Given $\mathbf{X}_{D_x \times N}$, $\mathbf{Y}_{D_y \times N}$, and initialization $\mathbf{Z}_{D_z \times N}$

repeat

1. Optimize over $\mathbf{g}$: $\min_{\mathbf{g}} \sum_{n=1}^{N} \|\mathbf{y}_n - \mathbf{g}(\mathbf{z}_n)\|^2 + \lambda_{\mathbf{g}} R_{\mathbf{g}}(\mathbf{g})$
2. Optimize over $\mathbf{F}$: $\min_{\mathbf{F}} \sum_{n=1}^{N} \|\mathbf{z}_n - \mathbf{F}(\mathbf{x}_n)\|^2 + \lambda_{\mathbf{F}} R_{\mathbf{F}}(\mathbf{F})$
3. Optimize over $\mathbf{Z}$: $\min_{\mathbf{Z}} \sum_{n=1}^{N} \|\mathbf{y}_n - \mathbf{g}(\mathbf{z}_n)\|^2 + \sum_{n=1}^{N} \|\mathbf{z}_n - \mathbf{F}(\mathbf{x}_n)\|^2$

until stop

3 Optimization over $\mathbf{F}$ and $\mathbf{g}$

We use shallower functions–linear or RBFs–for $\mathbf{F}$ and $\mathbf{g}$.

- Linear $\mathbf{g}$: a direct regression that acts on a lower input dimension D_z, reduces to least squares problem.
- RBFs $\mathbf{g}$: $\mathbf{g}(\mathbf{z}) = \mathbf{W}\boldsymbol{\Phi}(\mathbf{z})$ with $M \leq N$ Gaussian RBFs $\phi_m(\mathbf{z}) = e^{\left(-\frac{\|\mathbf{z}-\boldsymbol{\mu}_m\|^2}{2\sigma^2}\right)}$, and $R(\mathbf{g}) = \|\mathbf{W}\|^2$ is a quadratic regularizer on the weights.
  - Centers $\boldsymbol{\mu}_m$ are chosen by k-means on $\mathbf{Z}$ (once every few iterations, initialized at previous centers)
  - Weights $\mathbf{W}$ have a unique solution given by a linear system.
  - Time complexity: $\mathcal{O}(NM(M + D_z))$, linear in training set size.
  - Space complexity: $\mathcal{O}(M(D_y + D_z))$.

4 Optimization over $\mathbf{Z}$

- For fixed $\mathbf{g}$ and $\mathbf{F}$, optimization of the objective function decouples over each $\mathbf{z}_n \in \mathbb{R}^{D_z}$.
- We have N independent nonlinear minimizations each on D_z parameters, of the form $\min_{\mathbf{z} \in \mathbb{R}^{D_z}} E(\mathbf{z}) = \|\mathbf{y} - \mathbf{g}(\mathbf{z})\|^2 + \|\mathbf{z} - \mathbf{F}(\mathbf{x})\|^2$.
- If $\mathbf{g}$ is linear, then $\mathbf{z}$ can be solved in closed form by solving a linear system of size D_z.
- If $\mathbf{g}$ is nonlinear, we use Gauss-Newton method with line search.
- Cost over all $\mathbf{Z}$: $\mathcal{O}(ND_z^2 D_y)$, linear in training set size.
- The distribution of the coordinates $\mathbf{Z}$ changes dramatically in the first few iterations, while the error decreases quickly, but after that $\mathbf{Z}$ changes little.

5 Initialization and Validation

Initialization for $\mathbf{Z}$

- Unsupervised dimensionality reduction on $\mathbf{X}$ only.
- Supervised dimension reduction: Reduced Rank Regression, Kernel Slice Inverse Regression, Kernel Dimension Reduction, etc.
- Spectral methods run on $(\mathbf{X}, \mathbf{Y})$ jointly.

Validation of hyper-parameters

- Parameters for function $\mathbf{F}$ and $\mathbf{g}$ (#RBFs, Gaussian Kernel width).
- Regularization coefficients $\lambda_{\mathbf{g}}$ and $\lambda_{\mathbf{F}}$.
- Dimensionality of $\mathbf{z}$.

They can be determined through evaluating performance of $\mathbf{g} \circ \mathbf{F}$ on a validation set.

6 Advantages of low-dimensional regression

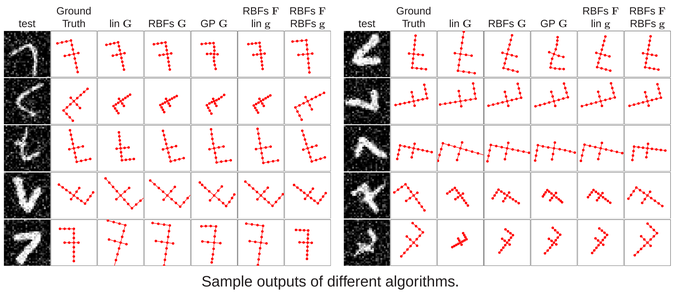
- We jointly optimize over both mappings $\mathbf{F}$ and $\mathbf{g}$, unlike one-shot methods.
- Our optimization is much more efficient than using a deep network with nested mappings (*pretty good model pretty fast*).
- The low-dimensional regressor has fewer parameters when D_z is small or #RBFs is small.
- The smooth functions $\mathbf{F}$ and $\mathbf{g}$ impose regularization on the regressor and may result in a better generalization performance.

7 Experimental evaluation

- We use $\mathbf{g} \circ \mathbf{F}$ as our regression function for testing, which is the natural "out-of-sample" extension for above optimization.
- Criteria: test error, and the quality of dimension reduction.
- Early stopping for training, usually happens within 100 iterations.

Rotated MNIST digits '7'

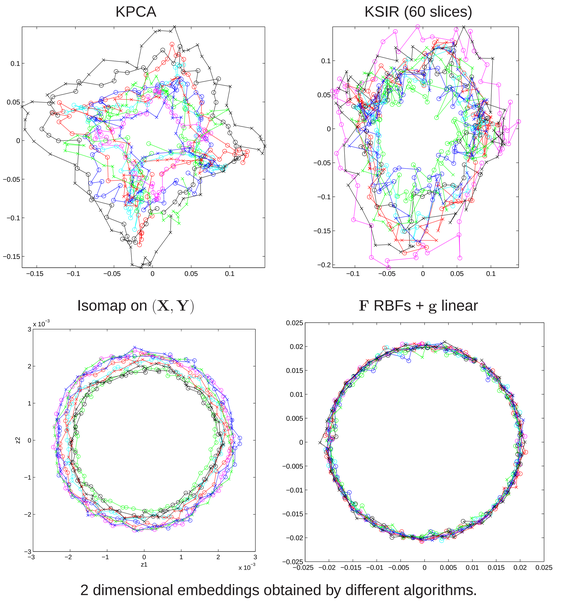
Input: images of digit 7, each has 60 different rotated versions.
Output: skeleton version of the digit.



Sample outputs of different algorithms.

Method	SSE
direct linear	51 710
direct RBFs $(1200, 10^{-2})$	32 495
direct RBFs $(2000, 10^{-2})$	29 525
Gaussian process	29 208
KPCA (3) + RBFs $(2000, 1)$	49 782
KSIR $(60, 26)$ + RBFs $(20, 10^{-5})$	39 421
$\mathbf{F}$ RBFs $(2000, 10^{-2})$ + g linear (10^{-3})	29 612
$\mathbf{F}$ RBFs $(1200, 10^{-2})$+g RBFs $(75, 10^{-2})$	27 346

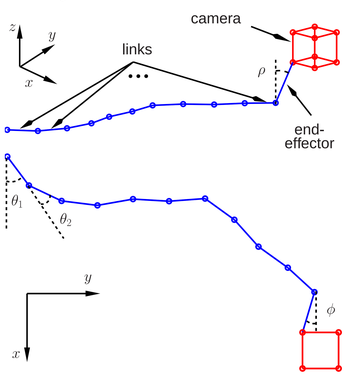
☞ Sum of squared errors (test set), with optimal parameters coded as RBFs (M, λ), KPCA (Gaussian kernel width), KSIR (number of slices, Gaussian kernel width). Number of iterations for our method: 16 (linear g), 7 (RBFs g).



2 dimensional embeddings obtained by different algorithms.

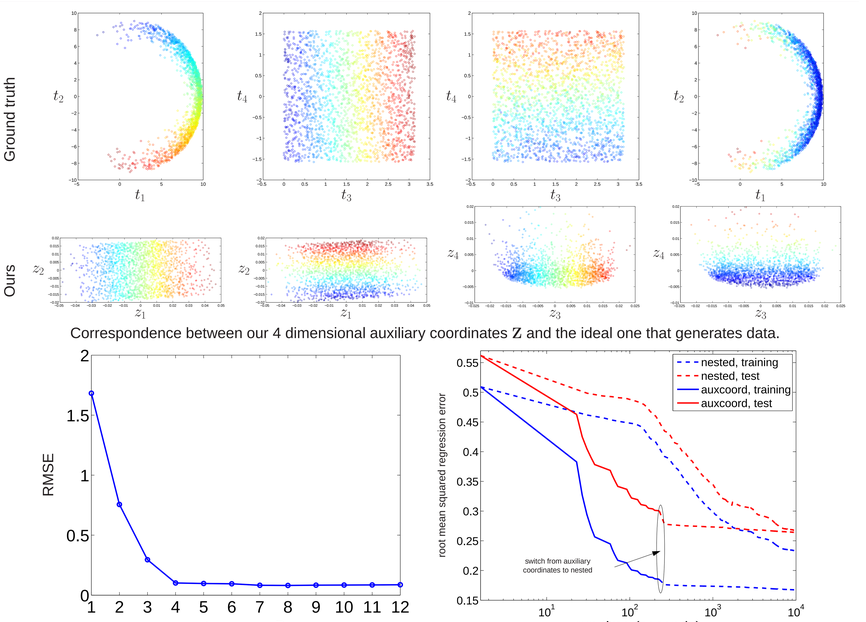
Serpentine robot forward kinematics

Forward kinematics mapping goes from 12 to 24 dimensions through 4 dimensions.



Method	RMSE
direct regression, linear (0)	2.2827
direct regression, RBF $(2000, 10^{-6})$	0.3356
direct regression, Gaussian process	0.7082
KPCA (2.5) + RBF $(400, 10^{-10})$	3.7455
KSIR $(400, 100)$ + RBF $(1000, 10^{-8})$	3.5533
$\mathbf{F}$ RBF $(2000, 10^{-6})$+g RBF $(100, 10^{-9}$	0.1006

Test error obtained by different algorithms.



Correspondence between our 4 dimensional auxiliary coordinates $\mathbf{Z}$ and the ideal one that generates data.

Validation of D_z by our algorithm.

Comparison of run time of our approach and optimizing the nested objective function.