Estimation of the Mean Function of Functional Data via Deep Neural Networks

Shuoyang Wang

AUBURN UNIVERSITY

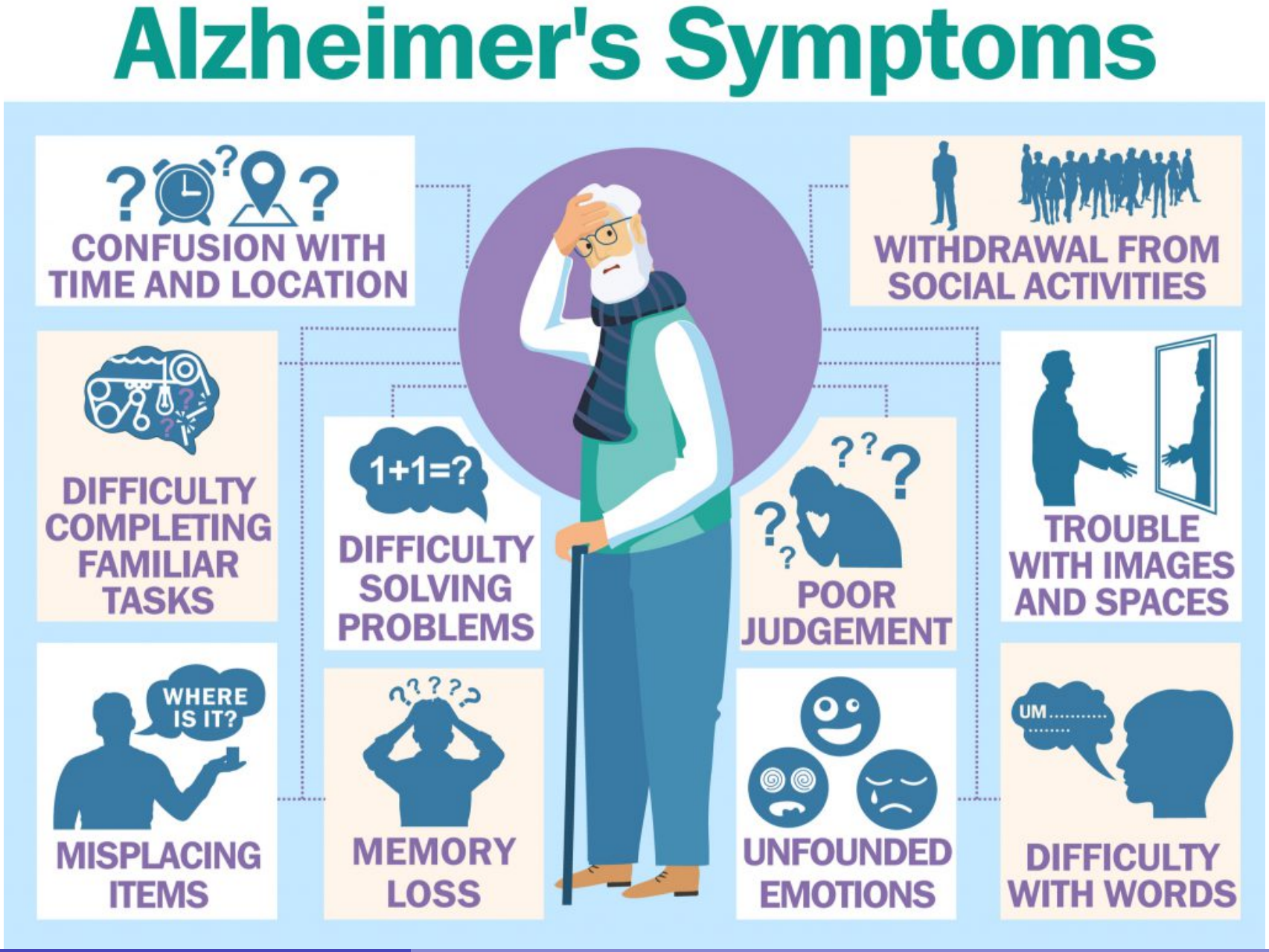
Symposium on Data Science and Statistics 2021

June 3, 2021

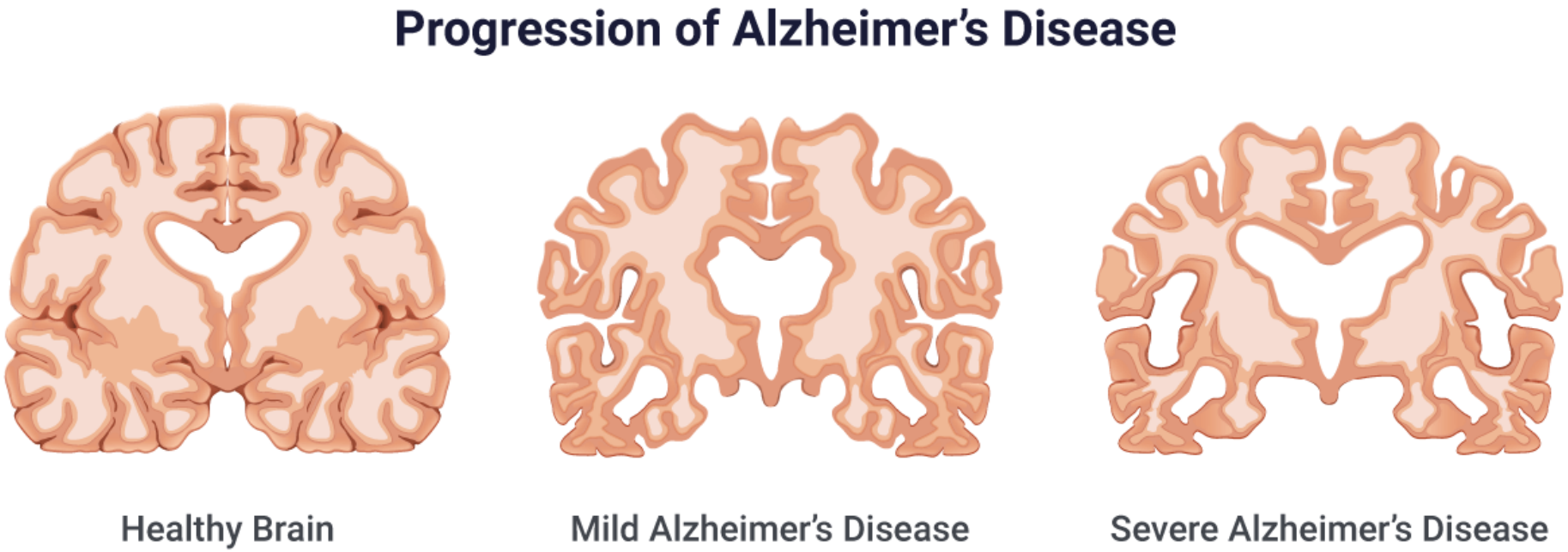
Acknowledgement

- Collaborators
  - Dr. Guanqun Cao, Auburn University
  - Dr. Zuofeng Shang, New Jersey Institute of Technology
- Partially funded by NSF award DMS-1736470

Alzheimer's Disease

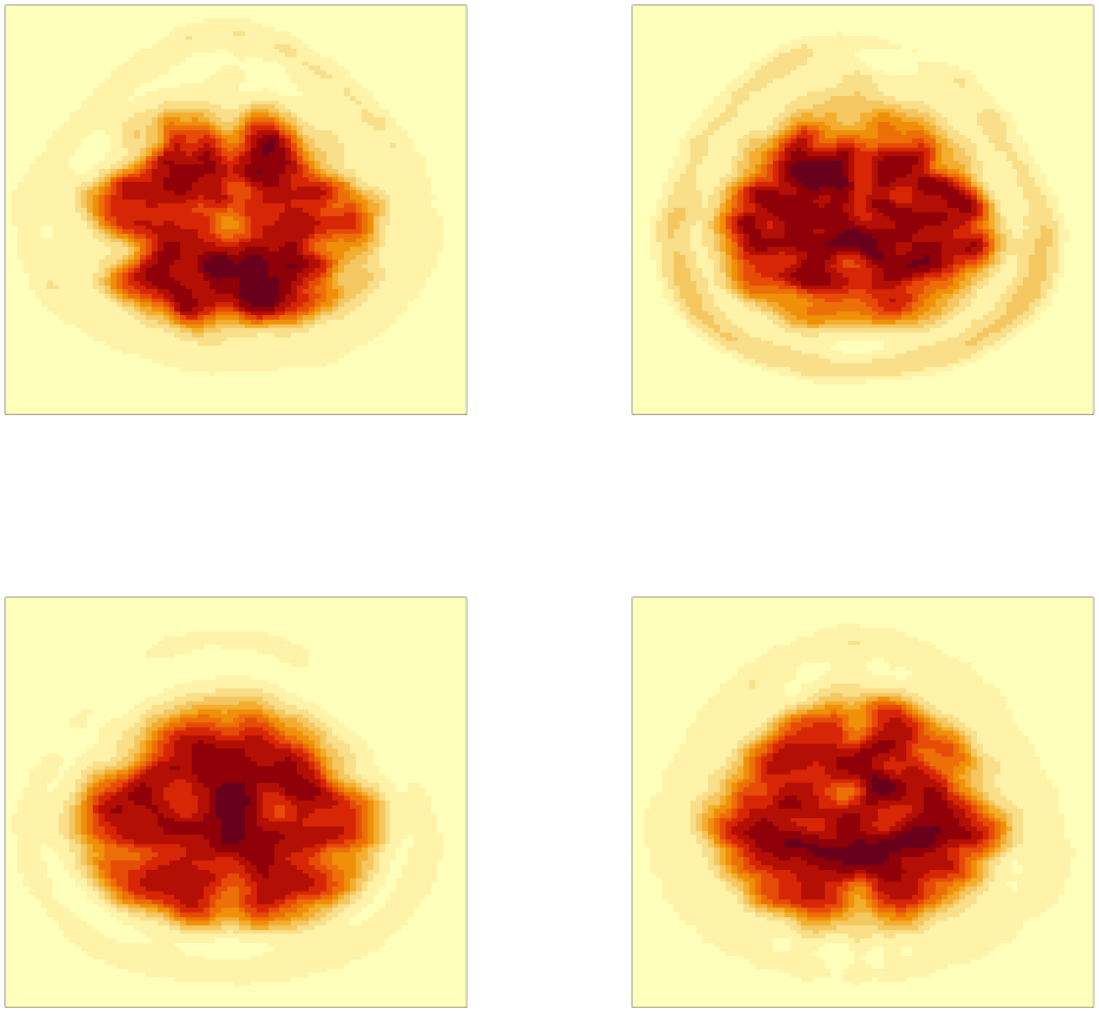


Brain Imaging



Source: https://www.caring.com/caregivers/alzheimers/

Positron Emission Tomography (PET) Images



Functional regression model

$$Y_{ij} = f_0\left(\mathbf{X}_j\right) + \eta\left(\mathbf{X}_j\right) + \epsilon_i\left(\mathbf{X}_j\right), \quad i = 1, 2, \ldots, n, j = 1, 2, \ldots, N,$$

- $f_0 : \mathbb{R}^d \to \mathbb{R}$, $E(Y_{ij}) = f_0\left(\mathbf{X}_j\right)$; $\mathbf{X}_j \in \mathbb{R}^d$; $d \geq 1$;
- $\eta\left(\cdot\right)$: individual curve variations; zero mean Gaussian process;
- $\epsilon_i\left(\cdot\right)$: zero mean measurement error;
- n: sample size;
- N: number of observations for each subject.

Covariance function

$$E(\eta(\mathbf{X}_j)) = 0, \quad G(\mathbf{X}_j, \mathbf{X}_{j'}) := \mathrm{Cov}(Y_{ij}, Y_{ij'}) = \mathrm{Cov}(\eta(\mathbf{X}_j), \eta(\mathbf{X}_{j'}))$$

Mercer's decomposition:

$$G(\mathbf{x}_j, \mathbf{x}_{j'}) = \sum_{k=1}^{\infty} \lambda_k \psi_k(\mathbf{x}_j) \psi_k(\mathbf{x}_{j'}),$$

Covariance function

$$E(\eta(\mathbf{X}_j)) = 0, \quad G(\mathbf{X}_j, \mathbf{X}_{j'}) := \mathrm{Cov}(Y_{ij}, Y_{ij'}) = \mathrm{Cov}(\eta(\mathbf{X}_j), \eta(\mathbf{X}_{j'}))$$

Mercer's decomposition:

$$G(\mathbf{x}_j, \mathbf{x}_{j'}) = \sum_{k=1}^{\infty} \lambda_k \psi_k(\mathbf{x}_j)\psi_k(\mathbf{x}_{j'}),$$

- $\{\lambda_k\}_{k=1}^{\infty}$ are eigenvalues with $\lambda_1 \geq \lambda_2 \geq \ldots \geq 0$, $\sum_{k=1}^{\infty} \lambda_k < \infty$;

Covariance function

$$E(\eta(\mathbf{X}_j)) = 0, \quad G(\mathbf{X}_j, \mathbf{X}_{j'}) := \mathrm{Cov}(Y_{ij}, Y_{ij'}) = \mathrm{Cov}(\eta(\mathbf{X}_j), \eta(\mathbf{X}_{j'}))$$

Mercer's decomposition:

$$G(\mathbf{x}_j, \mathbf{x}_{j'}) = \sum_{k=1}^{\infty} \lambda_k \psi_k(\mathbf{x}_j)\psi_k(\mathbf{x}_{j'}),$$

- $\{\lambda_k\}_{k=1}^{\infty}$ are eigenvalues with $\lambda_1 \geq \lambda_2 \geq \ldots \geq 0$, $\sum_{k=1}^{\infty} \lambda_k < \infty$;
- $\{\psi_k(\mathbf{x})\}_{k=1}^{\infty}$ are eigenfunctions with $\int \psi_k^2(\mathbf{x}) = 1$ and $\int \psi_k(\mathbf{x})\psi_{k'}(\mathbf{x})d\mathbf{x} = 0$

How to estimate mean function $f_0(\cdot)$?

Literature reviews on functional regression

- Nonparametric regression:
  - Most approaches still focus on 1D functional data ($\mathbf{X} \in \mathbb{R}$)
  - *bivariate splines*: Wang et al. (2019) handled an irregular 2D domain of the images.
  - *Haar wavelet*: Wang et al. (2014) considered functional linear regression for 3D brain image data.
  - *Unified estimator? $d \geq 3$?*
- Functional principal component analysis (FPCA):
  - Lila et al. (2016) proposed a FPCA model that can handle real functions observable on a 2D manifold.
  - Chen and Jiang (2017) analyzed functional/longitudinal data observed on a general d-dimensional domain.
  - *number of eigenfunctions?*

Literature reviews on DNN for functional data

- From the statistical perspective its application and theoretical research is still in its infancy stage.
- Rossi, et al.(2005) extended Radial-Basis function networks and multi-layer perceptron models to functional data inputs
- Thind, et al. (2020a,2020b) proposed for deep learning algorithms for functional linear regression from statistical point of view.

Deep neural networks

Definition

$$f(\mathbf{x}) = \mathbf{W}_L \sigma(W_{L-1} \ldots \sigma(\mathbf{W}_2 \sigma(\mathbf{W}_1 \mathbf{x} + v_1) + v_2) \ldots + v_{L-1}),$$

- $d = d_0 \to d_1 \to \ldots, \ldots \to d_L \to d_{L+1} = 1$;
- $\sigma(x) = \max(x, 0)$: ReLU activation function;
- $\mathbf{W}_\ell : p_\ell \times p_{\ell+1}$ weight matrix;

Sparse network space:

$$\mathcal{F}_{DNN}(L, \mathbf{p}, s) = \left\{ f : \max_{\ell=0,\ldots,L} \|\mathbf{W}_\ell\|_\infty + |\mathbf{v}_\ell|_\infty \leq 1, \sum_{\ell=0}^{L} \|\mathbf{W}_\ell\|_0 + |\mathbf{v}_\ell|_0 \leq s \right\}$$

Structured compositions of Hölder Functions

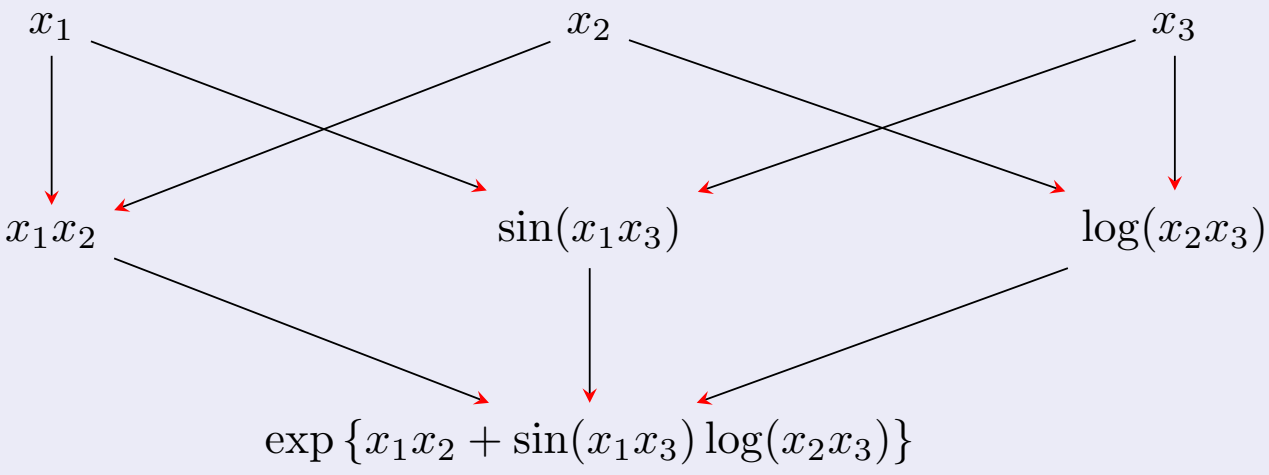
- $g_i : [a_i, b_i]^{d_i} \to [a_{i+1}, b_{i+1}]^{d_{i+1}}$, $g_i = \left(g_{ij}\right)_{j=1,\ldots,d_{i+1}}^{\top}$, ambient
- Each component g_{ij} is β_i-Hölder function with at most t_i-variate: $\left\{g_{ij} \in \mathcal{C}_{t_i}^{\beta_i}\left([a_i, b_i]^{t_i}, K_i\right), |a_i|, |b_i| \leq K_i\right\}$ intrinsic
- True underlying function space: $\mathcal{G}\left(q, \{d_i, t_i, \beta_i, K_i\}_{i\in[q]}\right)$ consists of $f_0 = g_q \circ g_{q-1} \circ \ldots \circ g_1 \circ g_0$
- Smoothness of $f_i = g_q \circ g_{q-1} \circ \ldots \circ g_i$
  $\beta_i^* := \beta_i \prod_{k=i+1}^{q} (\beta_k \wedge 1)$

Composition function example

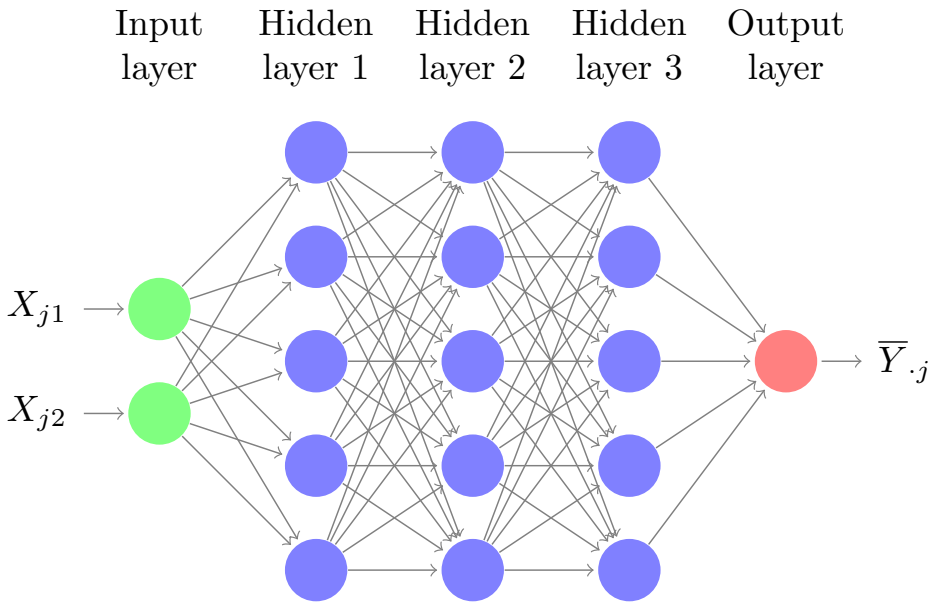
$$f_0(x_1, x_2, x_3) = \exp\{x_1 x_2 + \sin(x_1 x_3)\log(x_2 x_3)\}$$

Composition function example

$$f_0(x_1, x_2, x_3) = \exp\{x_1 x_2 + \sin(x_1 x_3)\log(x_2 x_3)\}$$



Functional regression via Deep Neural Networks



Empirical risk minimization

$$\widehat{f} = \arg\min_{f \in \mathcal{F}_{DNN}} \frac{1}{N} \sum_{j=1}^{N} \left\{ \overline{Y}_{\cdot j} - f\left(\mathbf{X}_j\right) \right\}^2,$$

where $\overline{Y}_{\cdot j} = n^{-1} \sum_{i=1}^{n} Y_{ij}$, $\mathbf{X}_j = (X_{j1}, \ldots, X_{jd})$

Non-asymptotic convergency rate

Theorem 1

Under mild assumptions, with probability greater than $(1 - \frac{2}{nN^{\varrho}})^{\log(nN^{\varrho})+1} \to 1$, we have

$$\|\widehat{f} - f_0\|_N^2 \leq c(nN^{\varrho})^{-\frac{\theta}{\theta+1}} \log^6(nN^{\varrho}),$$

where $\varrho \geq 0$, $\theta = \min_{i=0,\ldots,q} \frac{2\beta_i^*}{t_i}$ and c depend on true function class of f_0.

- $(nN^{\varrho})^{-\frac{\theta}{\theta+1}} = (nN^{\varrho})^{-\alpha}$ and $\alpha = \min_{i=0,\ldots,q} \frac{2\beta_i^*}{2\beta_i^*+t_i}$
- If $\varrho = 0$, $\|\widehat{f} - f_0\|_N^2 \leq cn^{-\frac{\theta}{\theta+1}} \log^6(n)$

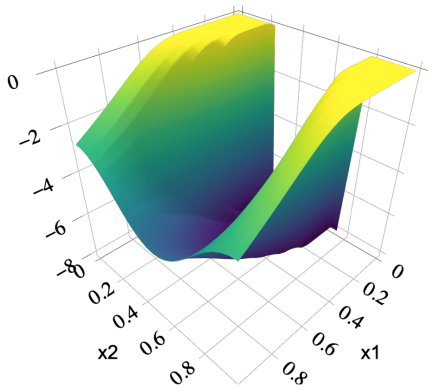
Implementation

R Package

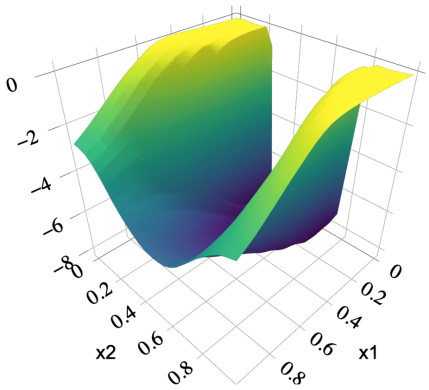
https://github.com/FDASTATAUBURN/FDADNN.

Case I (2D)

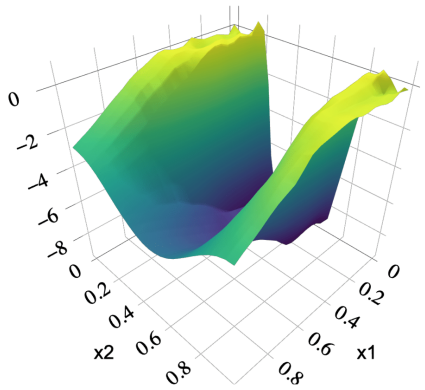
f_0



$\widehat{f}_{DNN}$



$\widehat{f}_{BS}$



$$f_0(x_{1j}, x_{2j}) = \frac{-8}{1+\exp\left(\cot(x_{1j}^2)\cos(2\pi x_{2j})\right)}$$

- $\widehat{f}_{BS}$: Bivariate spline smoothing (Wang et al., 2019)

Average empirical L_2 risk and their standard deviations (Case I)

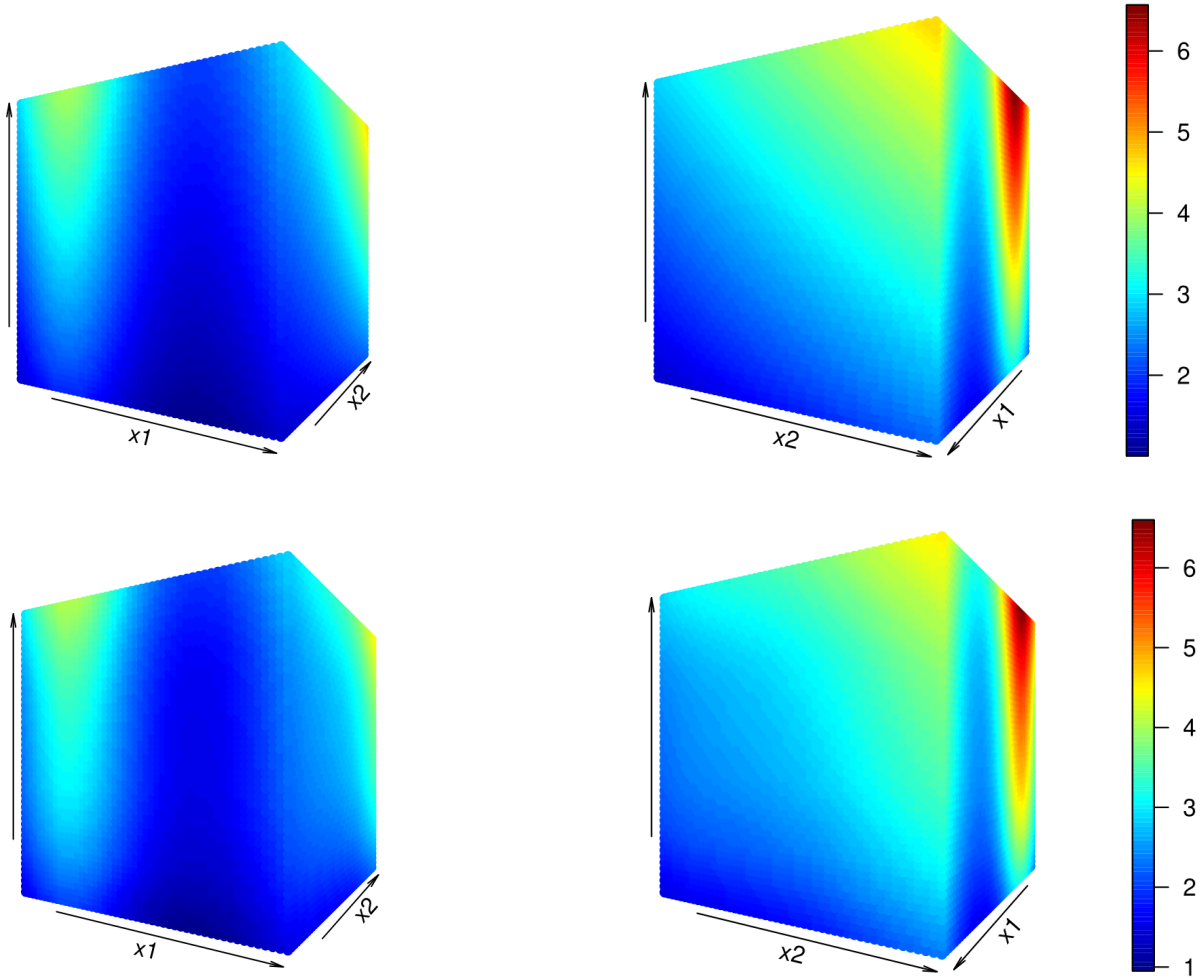
$$f_0(x_{1j}, x_{2j}) = \frac{-8}{1+\exp\left(\cot(x_{1j}^2)\cos(2\pi x_{2j})\right)}$$

σ	N	n	DNN		bivariate spline	
			L_2 risk	SD	L_2 risk	SD
1	225	50	0.1327	0.1905	0.6030	0.0418
		100	0.0797	0.1244	0.5757	0.0249
		200	0.0432	0.0574	0.5584	0.0120
	625	50	0.0770	0.0497	0.1497	0.0462
		100	0.0535	0.0368	0.1136	0.0214
		200	0.0352	0.0295	0.0987	0.0098
2	225	50	0.1880	0.1521	0.6564	0.1009
		100	0.0918	0.0793	0.6035	0.0619
		200	0.0593	0.0529	0.5765	0.0316
	625	50	0.1594	0.1555	0.2241	0.1218
		100	0.0862	0.0755	0.1430	0.0557
		200	0.0420	0.0412	0.1098	0.0232

Case II (3D)

f_0

$\widehat{f}_{DNN}$



$$f_0(x_{1j}, x_{2j}, x_{3j}) = \exp\left(\frac{1}{3}x_{1j} + \frac{1}{3}x_{2j} + \sqrt{x_{3j} + 0.1}\right)$$

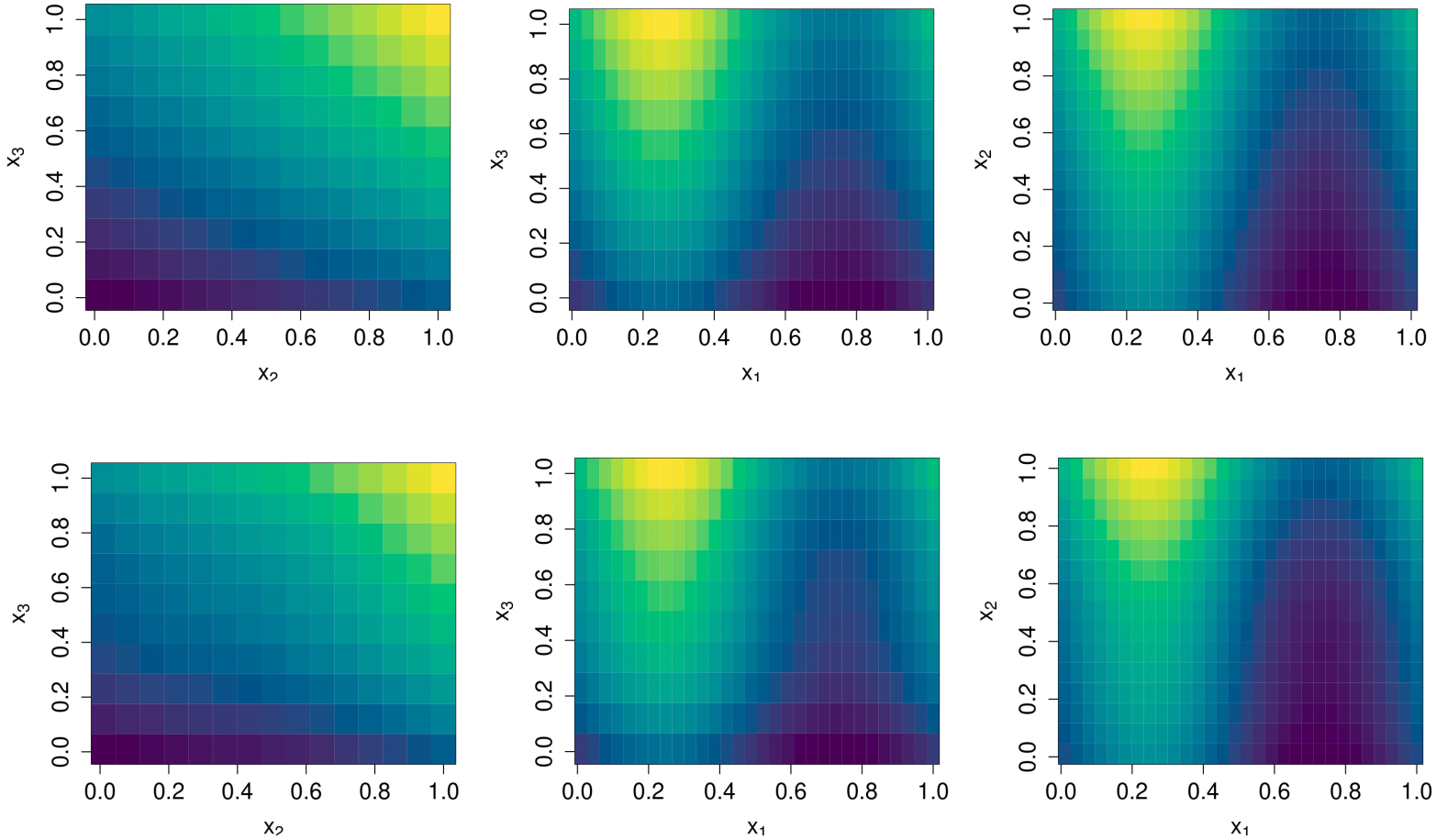
Average empirical L_2 risk and their standard deviations (Case III)

σ	N	n	L_2 risk	SD
1	3000	50	0.0028	0.0020
		100	0.0011	0.0006
		200	0.0006	0.0004
	4500	50	0.0007	0.0007
		100	0.0005	0.0007
		200	0.0003	0.0004
2	3000	50	0.0030	0.0024
		100	0.0012	0.0007
		200	0.0007	0.0005
	4500	50	0.0009	0.0007
		100	0.0005	0.0008
		200	0.0003	0.0005

2D slices

f_0

$\widehat{f}_{DNN}$



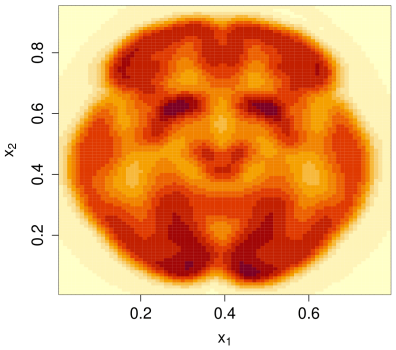
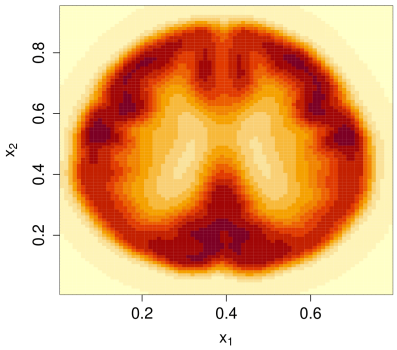
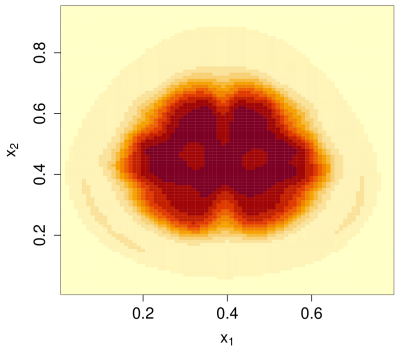
ADNI

- 79 patients from the AD group.
  - 33 females
  - 46 males

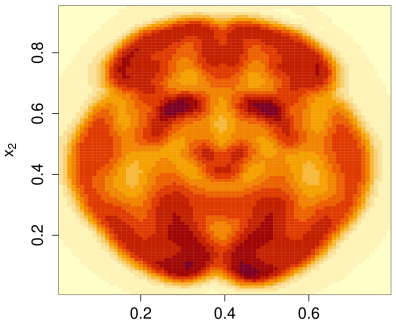
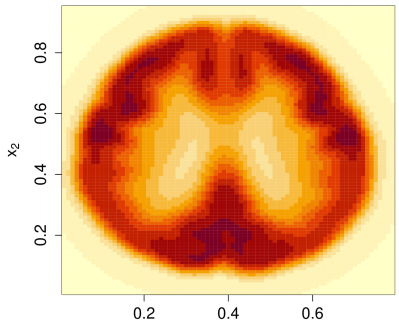
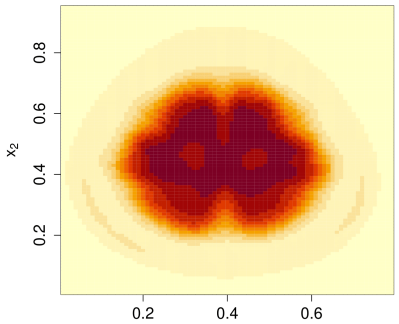
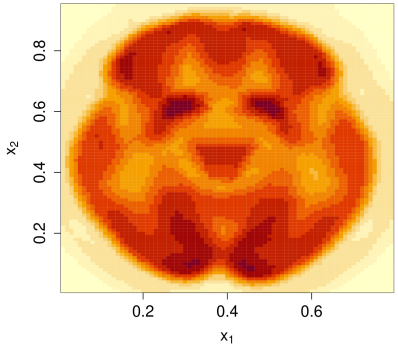
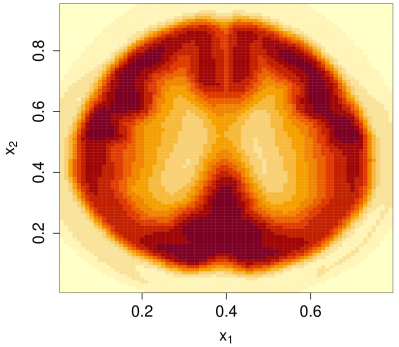
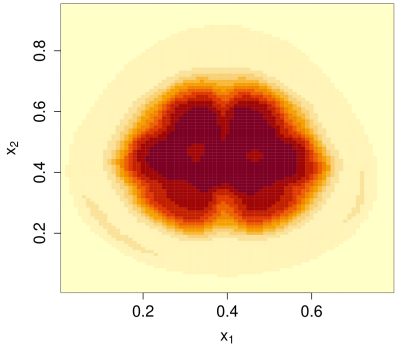
ADNI

- 79 patients from the AD group.
  - 33 females
  - 46 males
- reoriented into $79 \times 95 \times 69$ voxels.
- each patient has 69 sliced 2D images with 79×95.

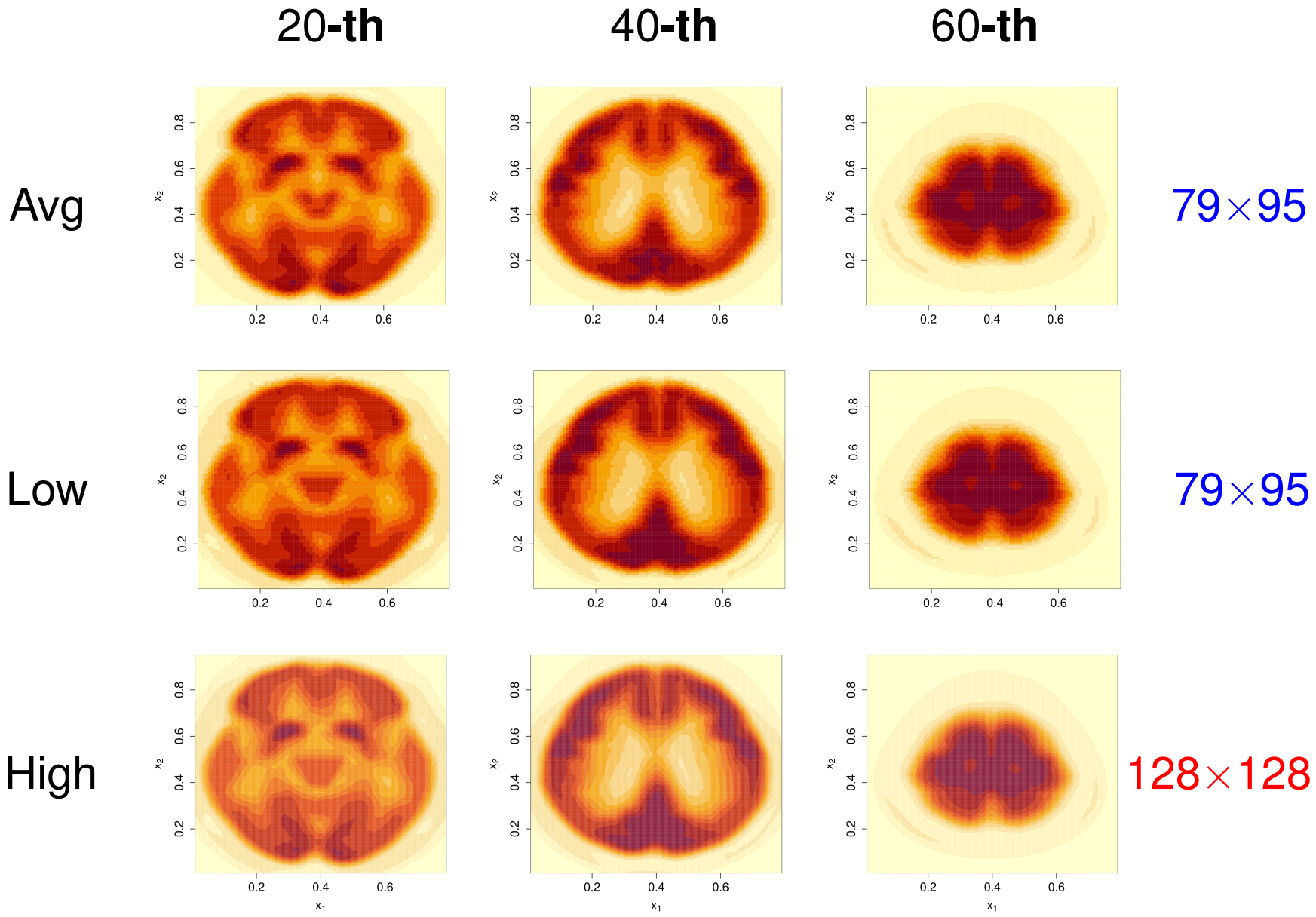
Recovery from 2D scans

	20-th	40-th	60-th	
Avg				79×95

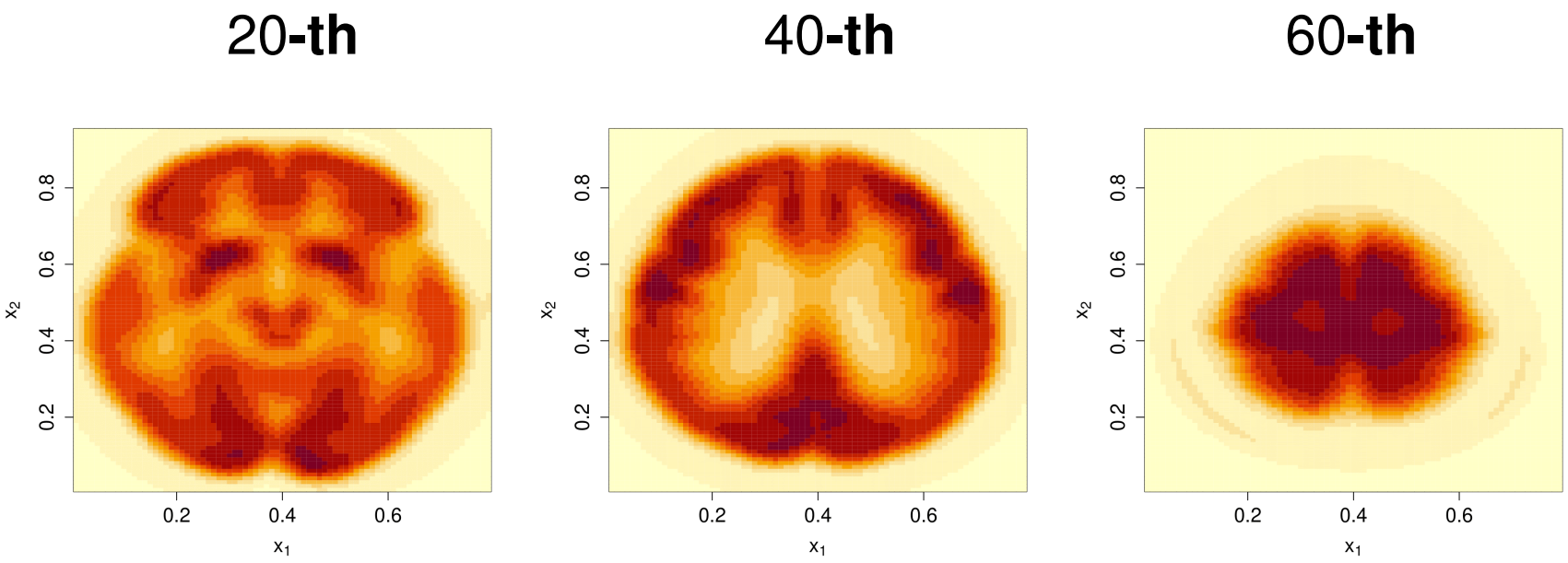
Recovery from 2D scans

	20-th	40-th	60-th	
Avg				79×95
Low				79×95

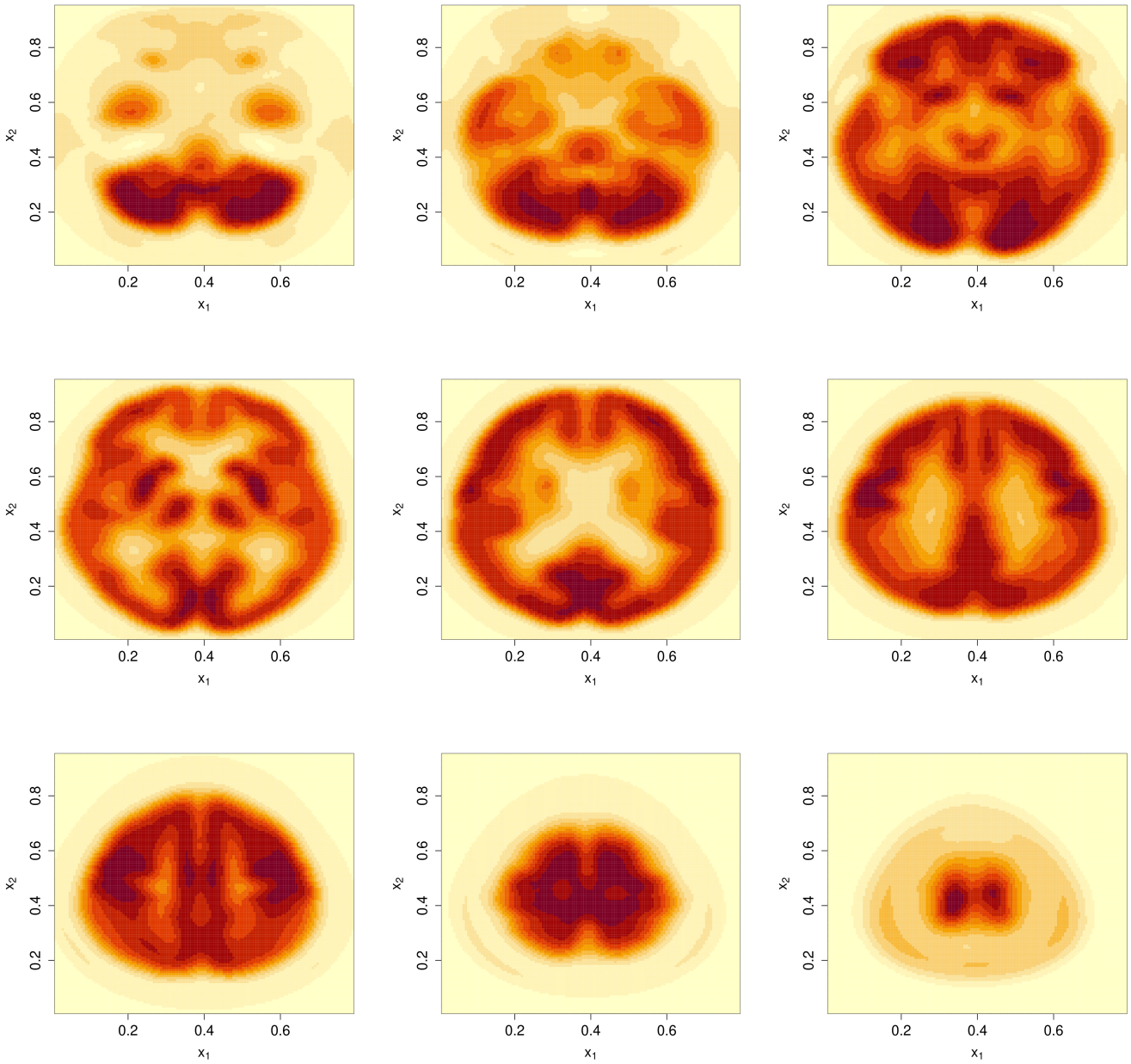
Recovery from 2D scans



Recovery (79×95) from 3D scans



Resolution from $79 \times 95 \times 69$ to $128 \times 128 \times 128$



Summary

- The proposed DNN estimator
  - achieves attractive empirical convergence rate, which is free from dimension d.
  - can recover the signal for multi-dimensional functional data (imaging data).
  - is unified for any dimensional functional data, which has broader and more flexible applications.
- We do not assume additional or complex structure for the true mean function.

- Wang, S., Cao, G. and Shang, Z. (2021) Estimation of the Mean Function of Functional Data via Deep Neural Networks. *Stat*

R Package

https://github.com/FDASTATAUBURN/FDADNN.

Assumptions

(A1) The true regression function f_0 has a composition structure.

Deep and wide neural networks

(A2) $\widehat{f} \in \mathcal{F}(L, \mathbf{p}, s)$, s.t.

- Depth: $L \asymp \log(nN^{\varrho})$, $\varrho \geq 0$;

Assumptions

(A1) The true regression function f_0 has a composition structure.

Deep and wide neural networks

(A2) $\widehat{f} \in \mathcal{F}(L, \mathbf{p}, s)$, s.t.

- Depth: $L \asymp \log(nN^{\varrho})$, $\varrho \geq 0$;
- Width: $\min_{l=1,\ldots,L} p_\ell \asymp (nN^{\varrho})^{\frac{1}{\theta+1}}$, where $\theta = \min_{i=0,\ldots,q} \frac{2\beta_i^*}{t_i}$

Assumptions

(A1) The true regression function f_0 has a composition structure.

Deep and wide neural networks

(A2) $\widehat{f} \in \mathcal{F}(L, \mathbf{p}, s)$, s.t.

- Depth: $L \asymp \log(nN^{\varrho})$, $\varrho \geq 0$;
- Width: $\min_{l=1,\ldots,L} p_\ell \asymp (nN^{\varrho})^{\frac{1}{\theta+1}}$, where $\theta = \min_{i=0,\ldots,q} \frac{2\beta_i^*}{t_i}$
- sparsity: $s \asymp (nN^{\varrho})^{\frac{1}{\theta+1}}$;

Assumptions

(A1) The true regression function f_0 has a composition structure.

Deep and wide neural networks

(A2) $\widehat{f} \in \mathcal{F}(L, \mathbf{p}, s)$, s.t.

- Depth: $L \asymp \log(nN^{\varrho})$, $\varrho \geq 0$;
- Width: $\min_{l=1,\ldots,L} p_\ell \asymp (nN^{\varrho})^{\frac{1}{\theta+1}}$, where $\theta = \min_{i=0,\ldots,q} \frac{2\beta_i^*}{t_i}$
- sparsity: $s \asymp (nN^{\varrho})^{\frac{1}{\theta+1}}$;

Assumptions

(A3) The maximal eigenvalue of the kernel matrix is $O(N^{-\varrho})$ for some constant $\varrho \geq 0$.