Neural Networks of ELM type for the numerical solution of PDEs with collocation

Prof. Francesco Calabrò

Dipartimento di Matematica e Applicazioni «Renato Caccioppoli»

Università degli studi di Napoli «Federico II», Italy

CALCOLO SCIENTIFICO E MODELLI MATEMATICI: alla ricerca delle cose nascoste attraverso le cose manifeste
Aula Convegni, CNR, Roma, 6-8 Aprile 2022

Outline

Mathematical notation for Data Analysis

Given <u>*data*</u> («individuals», «snapshots»), via collected features:

✓ Discrete output:

- Classificator - In Supervised Classification a set of «labelled» samples is available.
- Clustering - in Unsupervised Classification all samples are unlabled.

✓ The analogous regression/approximation problems are:

- Supervised case: Features are divided into input (or independent) variables and outcomes. The objective is to find a model: *predictive statistics, approximation function, regression modeling, parameters detection in inverse problems.*
- Unsupervised case: Find a way to describe the data in terms of some new variables so that the main behavior is preserved: *principal components, reduced models, projection techniques, feature selection, manifold learning, association rules*.

Interpolation properties: over-fitting?

- Belkin, M., Hsu, D., Ma, S., & Mandal, S. (2019). Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, *116*(32), 15849-15854.

Over-parametrized NN <--> Underdeterminated interpolation

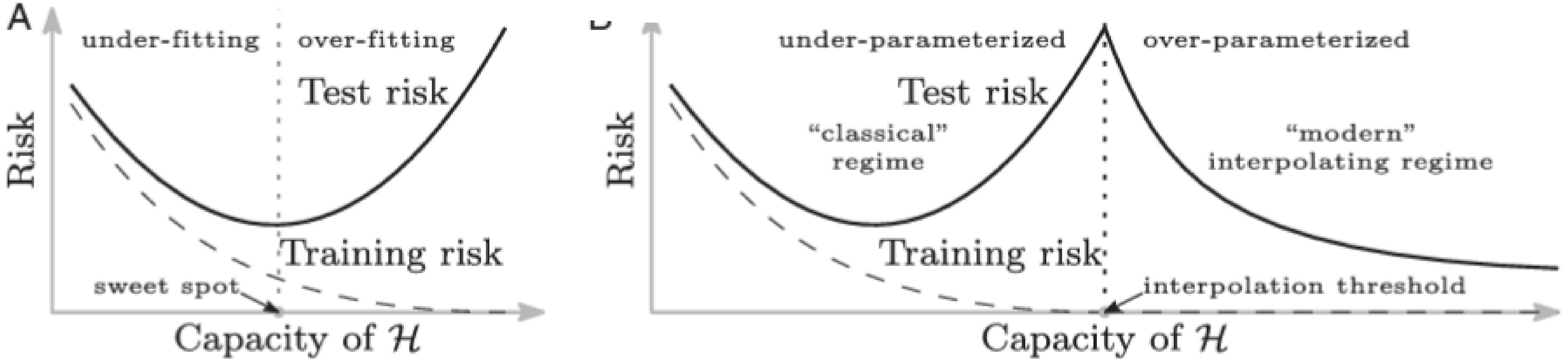
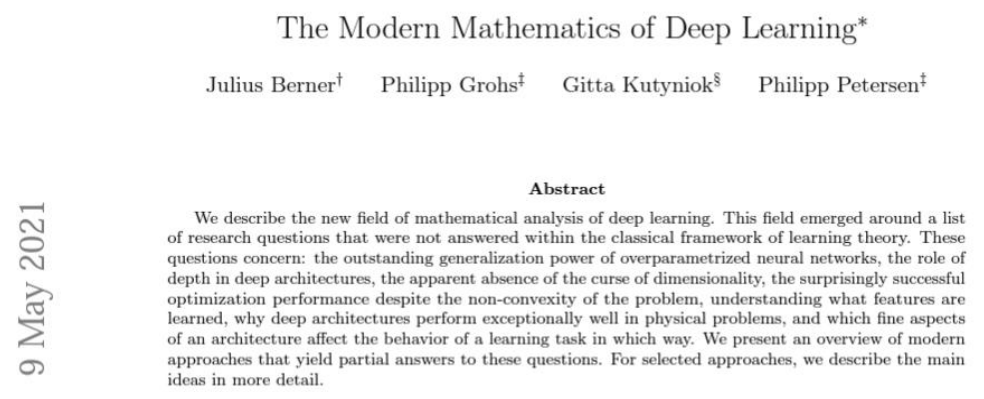


Fig. 1. Curves for training risk (dashed line) and test risk (solid line). (*A*) The classical U-shaped risk curve arising from the bias–variance trade-off. (*B*) The double-descent risk curve, which incorporates the U-shaped risk curve (i.e., the "classical" regime) together with the observed behavior from using high-capacity function classes (i.e., the "modern" interpolating regime), separated by the interpolation threshold. The predictors to the right of the interpolation threshold have zero training risk.

Some properties of Neural Networks



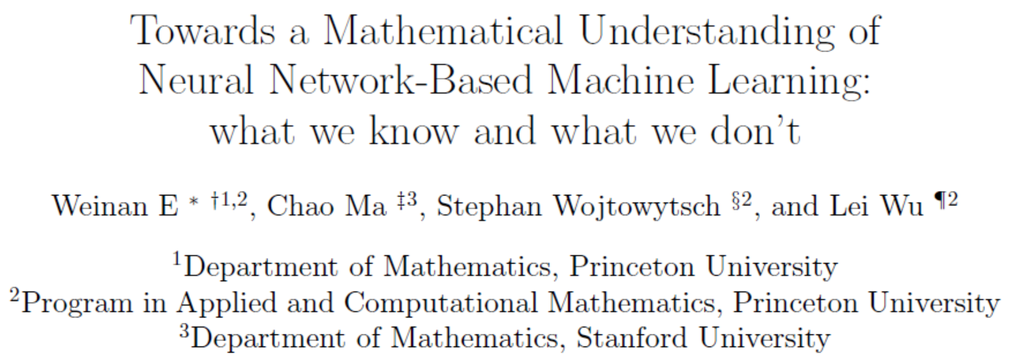
9 May 2021

The Modern Mathematics of Deep Learning*

Julius Berner† Philipp Grohs‡ Gitta Kutyniok§ Philipp Petersen‡

Abstract

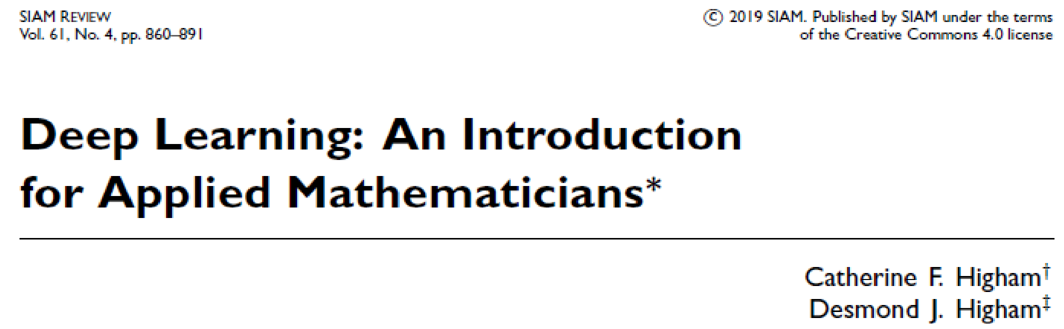
We describe the new field of mathematical analysis of deep learning. This field emerged around a list of research questions that were not answered within the classical framework of learning theory. These questions concern: the outstanding generalization power of overparametrized neural networks, the role of depth in deep architectures, the apparent absence of the curse of dimensionality, the surprisingly successful optimization performance despite the non-convexity of the problem, understanding what features are learned, why deep architectures perform exceptionally well in physical problems, and which fine aspects of an architecture affect the behavior of a learning task in which way. We present an overview of modern approaches that yield partial answers to these questions. For selected approaches, we describe the main ideas in more detail.



Towards a Mathematical Understanding of Neural Network-Based Machine Learning: what we know and what we don't

Weinan E * †1,2, Chao Ma ‡3, Stephan Wojtowytsch §2, and Lei Wu ¶2

1Department of Mathematics, Princeton University
2Program in Applied and Computational Mathematics, Princeton University
3Department of Mathematics, Stanford University

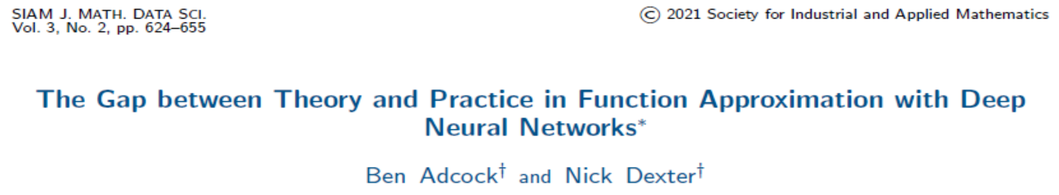


SIAM Review
Vol. 61, No. 4, pp. 860–891

© 2019 SIAM. Published by SIAM under the terms of the Creative Commons 4.0 license

Deep Learning: An Introduction for Applied Mathematicians*

Catherine F. Higham†
Desmond J. Higham‡

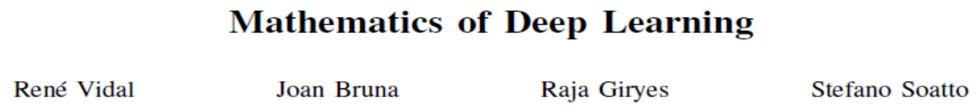


SIAM J. Math. Data Sci.
Vol. 3, No. 2, pp. 624–655

© 2021 Society for Industrial and Applied Mathematics

The Gap between Theory and Practice in Function Approximation with Deep Neural Networks*

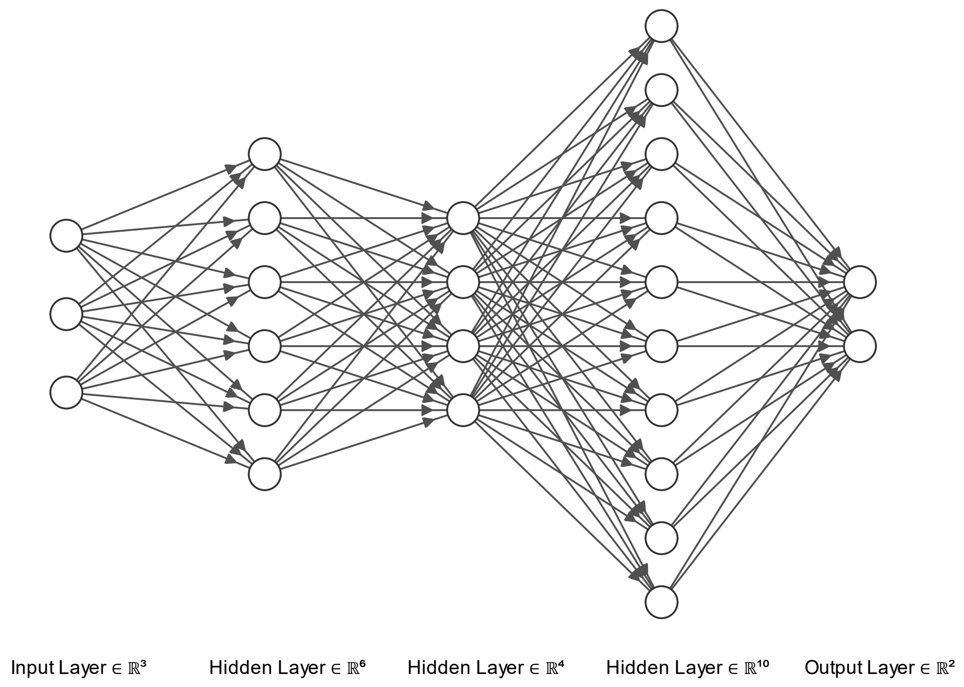
Ben Adcock† and Nick Dexter†



Mathematics of Deep Learning

René Vidal Joan Bruna Raja Giryes Stefano Soatto

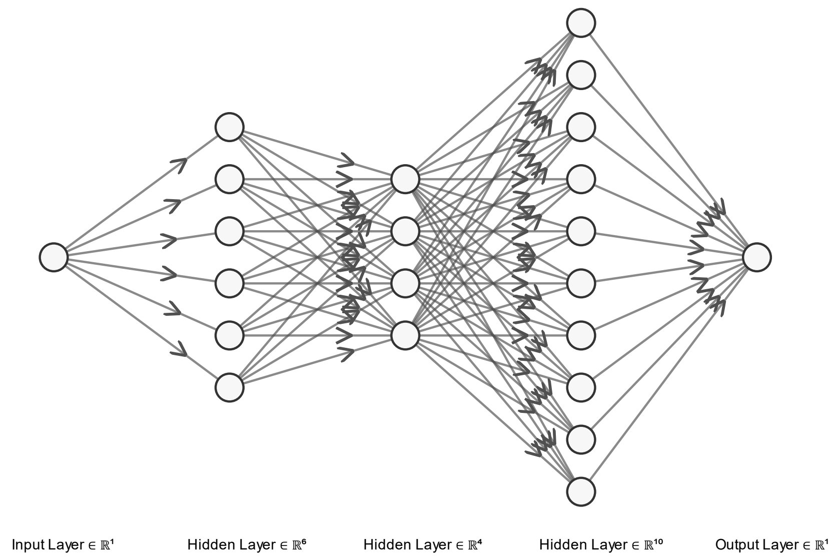
Feed–Forward Neural networks



Input Layer $\in \mathbb{R}^3$
Hidden Layer $\in \mathbb{R}^6$
Hidden Layer $\in \mathbb{R}^4$
Hidden Layer $\in \mathbb{R}^{10}$
Output Layer $\in \mathbb{R}^2$

FFNs: A set of connected units/nodes (neurons) that take from multiple input a real valued output by a *transfer* (activation) function.
Neurons are stored in layers: the *input layer*, *hidden layers* and the *output layer*.
Each layer consist of neurons, that are fully connected with nodes of the previous by weighted edges. *Weights* between layer and *vector of biases* describe the connection between layers.

Combining input data in the activation function: the case of additive nodes.



Each i-th neuron in the l-th (hidden) layer has $A_i^{(l)}$ (weights) and $\beta_i^{(l)}$ (bias) as *parameters*. It uses the input values and an activation function $\psi_i^{(l)}$ to compute it's output $x_i^{(l)}$: $x_i^{(l)} = \psi_i^{(l)}(z_i^{(l)})$ where $z_i^{(l)} = \sum_{j=1}^{N_{l-1}} A_{ij}^{(l)} \cdot x_j^{(l-1)} + \beta_i^{(l)}$.
Most common choices for activation functions are:

- Logistic Sigmoid (LS): $\psi(x) = \frac{1}{1+\exp(-x)}$
- Rectified Linear Unit (ReLU): $\psi(x) = x^+ = max\{x, 0\}$

The effect of deep structure: feature space vs function composition

All together, hidden layers encode the input in $N_{\mathcal{L}}$ new coordinates (feature space), acting thus as a filter. We conside the (univariate) case, where $N_{\mathcal{L}+1} = 1$, and denote $u := x^{(\mathcal{L}+1)}$. Then, matrix $A^{\mathcal{L}+1}$ reduce to a vector $w = (w_1, w_2, \dots, w_{N_{\mathcal{L}}}) \in \mathbb{R}^{N_{\mathcal{L}}}$ that we denote as *external weigths*.
In this case, the whole network is a map $\mathcal{F} : x^{(0)} \in \mathbb{R}^{N_0} \to u \in \mathbb{R}$ between the input domain and the output domain that can be expressed by composition:

$$\mathcal{F}(x^{(0)}) = \Phi^{(\mathcal{L}+1)} \circ \cdots \circ \Phi^{(1)}(x^{(0)})$$

Usually the final output is taken as linear combination of evaluations done at the last hidden layer, so that transfer functions $\psi_i^{(\mathcal{L}+1)}$ of the last layer are the identity functions:

$$\sum_{j=1}^{N_{\mathcal{L}}} A_j^{(\mathcal{L}+1)} \psi_j^{(\mathcal{L})}(z_j^{(\mathcal{L})}) = \sum_{j=1}^{N_{\mathcal{L}}} w_j \psi_j^{(\mathcal{L})}(z_j^{(\mathcal{L})}) .$$

Universal approximation theorem for FNN with additive nodes

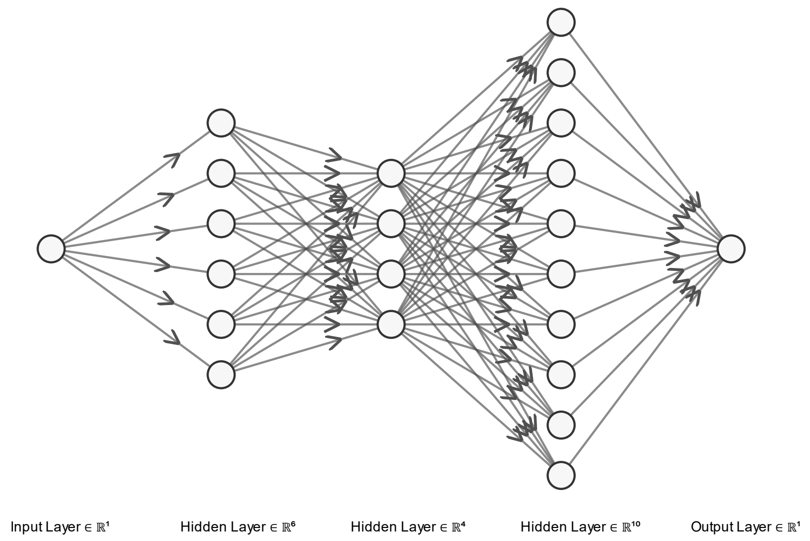
Consider FFN with the same transfer function ψ in all internal layers, with $\psi \in C(\mathbb{R})$. Then the space $\mathcal{M}_{\mathcal{L}}(\psi)$ generated by linear combination of the last hidden layer output $x^{(\mathcal{L})}$, i.e. defined by:

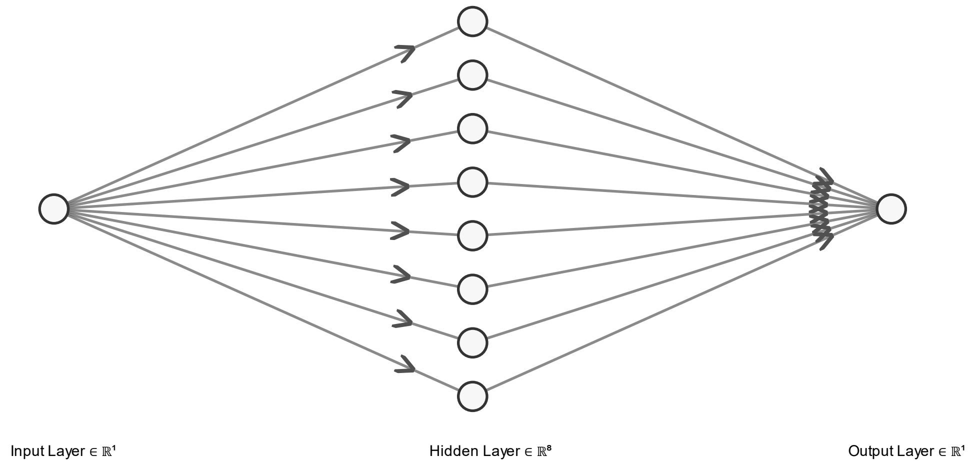
$$\mathcal{M}_{\mathcal{L}}(\psi) = \left\{ \sum_{j=1}^{N_{\mathcal{L}}} w_j \psi(A_j^{(\mathcal{L})} x^{(\mathcal{L}-1)} + \beta_j^{(\mathcal{L})}) : w_j, \beta_j^{(l)} \in \mathbb{R}, A_j^{(l)} \in \mathbb{R}^{N_{(l-1)}}, \right.$$
$$\left. : j = 1, \ldots, N_l, l = 1, \ldots, \mathcal{L} + 1 \right\}$$

is dense in $C(\mathbb{R}^{N_0})$, in the topology of uniform convergence on compacta, if and only if ψ is not a polynomial.

No need to consider a deep structure!

Deep NN vs Single Layer NN





Single layer case / Extreme Learning Machine

In the single layer case one obtains:

$$\sum_{j=1}^{N} w_j \psi_j(z_j) \ .$$

where N are the number of neurons. Exploring explicitly z_i we have:

$$\sum_{j=1}^{N} w_j \psi_j \left(\beta_j + \sum_{i=1}^{N_0} \alpha_{ij} x_i^{(0)} \right) \ .$$

Moreover, we take the same activation function $\psi_j = \psi$.

We refer to such functions as:

$$v(x; A, \beta; w) = \sum_{j=1}^{N} w_j \psi(\alpha_j \cdot x + \beta_j)$$

If we fix a-priori both internal weights A and bias β according to a continuous sampling distribution probability and keep only the external weights w as trainable parameters the single - hidden - layer - feedforward network is referred as Extreme Learning Machine (ELM).

Universal Approximation theorem for ELM

Universal approximation

Let the coefficients α_j, β_j in the function sequence $\{\psi(\alpha_j \cdot x + \beta_j)\}_{j=1}^N$ be randomly generated according to any continuous sampling distribution and call

$$\tilde{v}^N \in \mathrm{span}\{\psi(\alpha_j \cdot x + \beta_j)\,,\ j = 1 \ldots N\}$$

the ELM function determined by ordinary least square solution of $\|f(x) - \tilde{v}^N(x)\|$, where f is a continuous function.
Then, if $\mathcal{M}(\psi)$ spans L^2, this implies that $\lim_{N\to\infty} \|f - \tilde{v}^N\|_2 = 0$ with probability 1.

Interpolation / Collocation equations with ELM

Within the ELM framework, the interpolation problem leads to a system of linear equations, where the only unknowns are the external weights w. Consider M points x_i such that $y_i = f(x_i)$ for $i = 1, \ldots, M$. Then the interpolation problem reads:

$$\textit{Find } w \textit{ such that} \qquad \sum_{j=1}^{N} w_j \psi_j(x_i) = y_i, \qquad i = 1, \ldots, M \, .$$

Here N is the number of neurons and $\psi_j(x)$ is used to denote $\psi(\alpha_j \cdot x + \beta_j)$.

Thus, this is a linear system of M equations and N unknowns.

Interpolation with ELM, univariate case

Approximation of discrete data - Interpolation

Let $(x_i, y_i)\,,\ i = 1, \dots, M$ be a set of points such that $x_i < x_{i+1}$, and take the ELM network with N neurons $\tilde{u}(x; w)$ such that the internal weights α and the biases β are randomly generated independently from the data according to any continuous probability distribution. Then, $\forall \varepsilon > 0$, there exists $N < M$ and a choice of w such that $\|(\tilde{u}(x_i; w) - y_i)_i\| < \varepsilon$. Here $\|\cdot\|$ denotes the L^2 Euclidean norm of vectors, the analogous of the Frobenius norm. Moreover, if $N \geq M$ then w can be found such that $\|(\tilde{u}(x_i; w) - y_i)_i\| = 0$.

The use of NN for the resolution of PDEs

The use of NN for PDEs problems, main references

Lagaris, I.E., Likas, A. and Fotiadis, D.I., 1998. Artificial neural networks for solving ordinary and partial differential equations. *IEEE transactions on neural networks*, *9*(5), pp.987-1000.

Weinan, E., Han, J., & Jentzen, A. (2017). Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Communications in Mathematics and Statistics*, *5*(4), 349-380.

E, W., Yu, B. The Deep Ritz Method: A Deep Learning-Based Numerical Algorithm for Solving Variational Problems. *Commun. Math. Stat.* **6,** 1–12 (2018).

Sirignano, J., & Spiliopoulos, K. (2018). DGM: A deep learning algorithm for solving partial differential equations. *Journal of computational physics*, *375*, 1339-1364.

Berg, J. and Nyström, K., 2018. A unified deep artificial neural network approach to partial differential equations in complex geometries. *Neurocomputing*, 317, pp.28-41.

Physics-informed neural networks (PINNs)

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, *378*, 686-707.

Mishra, S., & Molinaro, R. (2020). Estimates on the generalization error of physics informed neural networks (PINNs) for approximating PDEs. *IMA Journal of Numerical Analysis 2022*

Lu, Lu, Xuhui Meng, Zhiping Mao, and George Em Karniadakis. "DeepXDE: A deep learning library for solving differential equations." *SIAM Review* 63, no. 1 (2021): 208-228.

Karniadakis, George Em, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. "Physics-informed machine learning." Nature Reviews Physics 3, no. 6 (2021): 422-440.

Extreme learning machine collocation method

- ✓ We use Extreme Learning Machine random projection networks as discrete space.
- ✓ Free paramenters are fixed via collocation equations.
- ✓ We underdeterminate the collocation equations.
- ✓ In the parabolic case: semidiscretize in time and solve the elliptic PDEs.

Calabrò, F., Fabiani, G., & Siettos, C. (2021). Extreme learning machine collocation for the numerical solution of elliptic PDEs with sharp gradients. *Computer Methods in Applied Mechanics and Engineering*, *387*, 114188.	→ Liner elliptic PDEs.
Fabiani, G., Calabrò, F., Russo, L. & Siettos, C. (2021). Numerical solution and bifurcation analysis of nonlinear partial differential equations with extreme learning machines. *J Sci Comput* **89,** 44	→ Nonlinear elliptic PDEs: tuning points of bifurcation diagrams.
Calabrò, F., Cuomo, S., di Serafino, D., Izzo, G. & Messina, E. (2022). The effect of time discretization on the solution of parabolic PDEs with ANNs, in preparation	→ Linear Parabolic PDEs. (W.I.P.)

Extreme learning machine collocation method

Problem formulation: the collocation equations (1/2)

The general problem is:

$$\partial_t u = \mathcal{L}(u) + f(t,x)$$

Numerical solution at fixed time will be:

$$\tilde{u}^{[n]}(x) = \sum_i \alpha_i^{[n]} \psi_i(x),$$

Ad example, with the θ-method it solves:

$$-\theta \Delta t \mathcal{L}(\tilde{u}^{[n]}) + \tilde{u}^{[n]}(x) = \\ \tilde{u}^{[n-1]}(x) + \theta \Delta t f(t_n, x) + \\ (1-\theta)\Delta t \left[\mathcal{L}(\tilde{u}^{[n-1]}) + f(t_{n-1}, x)\right].$$

In the stationary case:

$$\mathcal{L}(u(x)) = f(x)\,,\ \mathcal{B}(u(x)) = g(x)$$

And for the solution
$\tilde{u}(x; w) = \sum_{i=1}^{N} w_i \psi_i(x)$
we ask exactness on points $\{x_j\} \in \Omega$,
$\{\xi_b\} \in \delta\Omega$:

$$\mathcal{L}(\tilde{u}(x_j)) = f(x_j)\,,\ \mathcal{B}(\tilde{u}(\xi_b)) = g(\xi_b)$$

By linearity,

$$\begin{cases} \mathcal{L}(\tilde{u}(x_j)) = \sum_{i=1}^{n} w_i \mathcal{L}(\psi_i(x_j)) \\ \mathcal{B}(\tilde{u}(\xi_b)) = \sum_{i=1}^{n} w_i \mathcal{B}(\psi_i(\xi_b)) \end{cases}$$

Extreme learning machine collocation method

Problem formulation: the collocation equations (2/2)

To present the approach, we have chose a steady-state one-dimensional ellpitic PDE.

$$-\mu u''(x) + \gamma u'(x) + \lambda u(x) = f(x),$$

where μ, γ and λ are the diffusion, advection and reaction terms, respectively.
The approximate solution $\tilde{u}(x; w) \approx u(x)$, $\tilde{u}(x; w) = \sum_{i=1}^{N} w_i \psi_i(x)$.
Collocating the equations on M points, we have that the approximate solution solves:

$$-\mu \tilde{u}''(x_j) + \gamma \tilde{u}'(x_j) + \lambda \tilde{u}(x_j) - f(x_j) = 0 \ .$$

By linearity:

$$-\mu \sum_i w_i \psi_i''(x_j) + \gamma \sum_i w_i \psi_i'(x_j) + \lambda \sum_i w_i \psi_i(x_j) - f(x_j) = 0 \ .$$

The previous is a $j-$th row (out of N) of a matrix problem $\Phi w - \mathrm{f} = 0$ where $\Phi_{j,i} = -\mu \psi_i''(x_j) + \gamma \psi_i'(x_j) + \lambda \psi_i(x_j)$ We choose to overparametrize the problem / underdeterminate the system ($N > M$) and solve it by Least Square procedure. This gives that the functions, constructed in a random way, can be "selected" by the LS method.

Extreme learning machine collocation method

The discrete space: Sigmoidal transfer functions

Basis functions:

$$\sigma_i(x) = \sigma(\alpha_i x + \beta_i) = \frac{1}{1+\exp(-\alpha_i x - \beta_i)} .$$

Parameters α are chosen randomly with respect to a uniform distribution. Then β are computed so that the inflection points are uniformly spaced.

The approximation space takes advantage of these different shapes. Then the solution is a linear combination of such sigmoidal functions:

$$\tilde{u}(x; w) = \sum_{i=1}^{n} w_i \sigma_i(x).$$

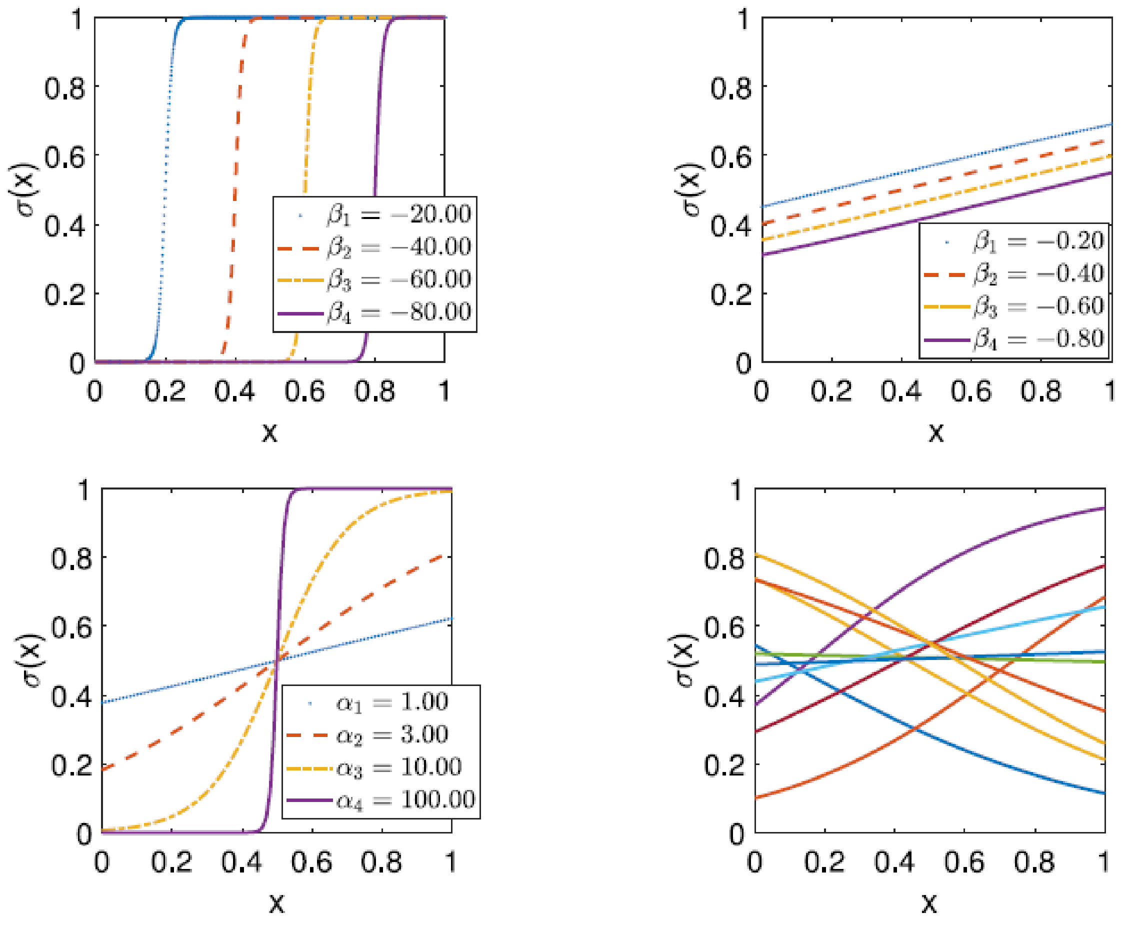


Fig. 1. The functions σ_i of (2.2) with varying parameters. On the top left panel, we have set $\alpha_i = 100$, on the top right, $\alpha_i = 1$. On the bottom left panel we show functions with a fixed center $C_i = 1/2$. On the bottom right panel, a set of 10 functions obtained by varying the coefficients as stated in Section 3 are depicted.

Extreme learning machine collocation method

Ex1: High order polynomial problem

Example 1)

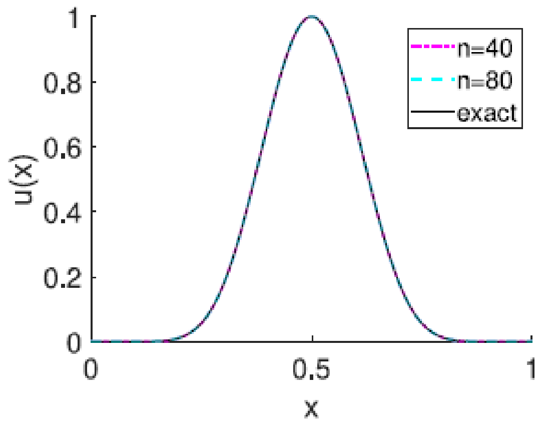
The linear problem:

$$\begin{cases} u'' = 2^{2p}p(1-x)^{p-2}x^{p-2}(-1+2x-2x^2+p(1-4x+4x^2)), \ 0<x<1 \\ u(0)=0\,, \ u(1)=0 \end{cases}$$

With exact solution:

$$u(x) = 2^{2p}x^p(1-x)^p$$

For comparison, we take the 7-points centered-stencil FD solution. L^2 norm is used for the evaluation of the error.



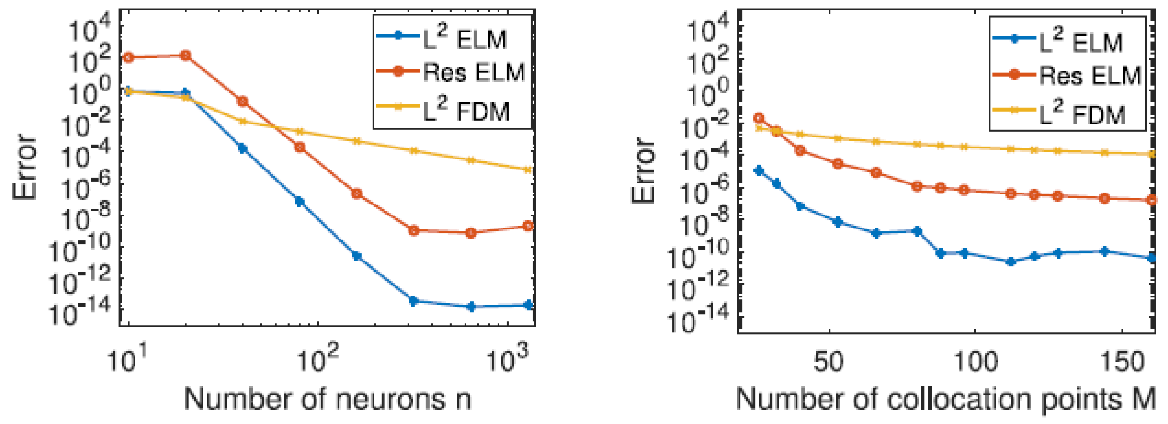


Fig. 5. Numerical tests for the solution of problem (5.2) with $p = 10$. On the top panel are depicted the exact-analytical (5.3) and the ELM numerical solutions with $n = 40, 80$; the number of collocation points M was set to $n/2$. On the bottom panels are shown the error and residual convergence: on the left panel, we have fixed $M = n/2$ and varied n and on the right panel we have fixed $n = 80$ and varied M.

Extreme learning machine collocation method

Ex2: Boundary layer problem: linear diffusion-reaction

Example 2)
The linear steady-state one-dimensional ellpitic PDE that exhibits steep gradient: μ<<λ

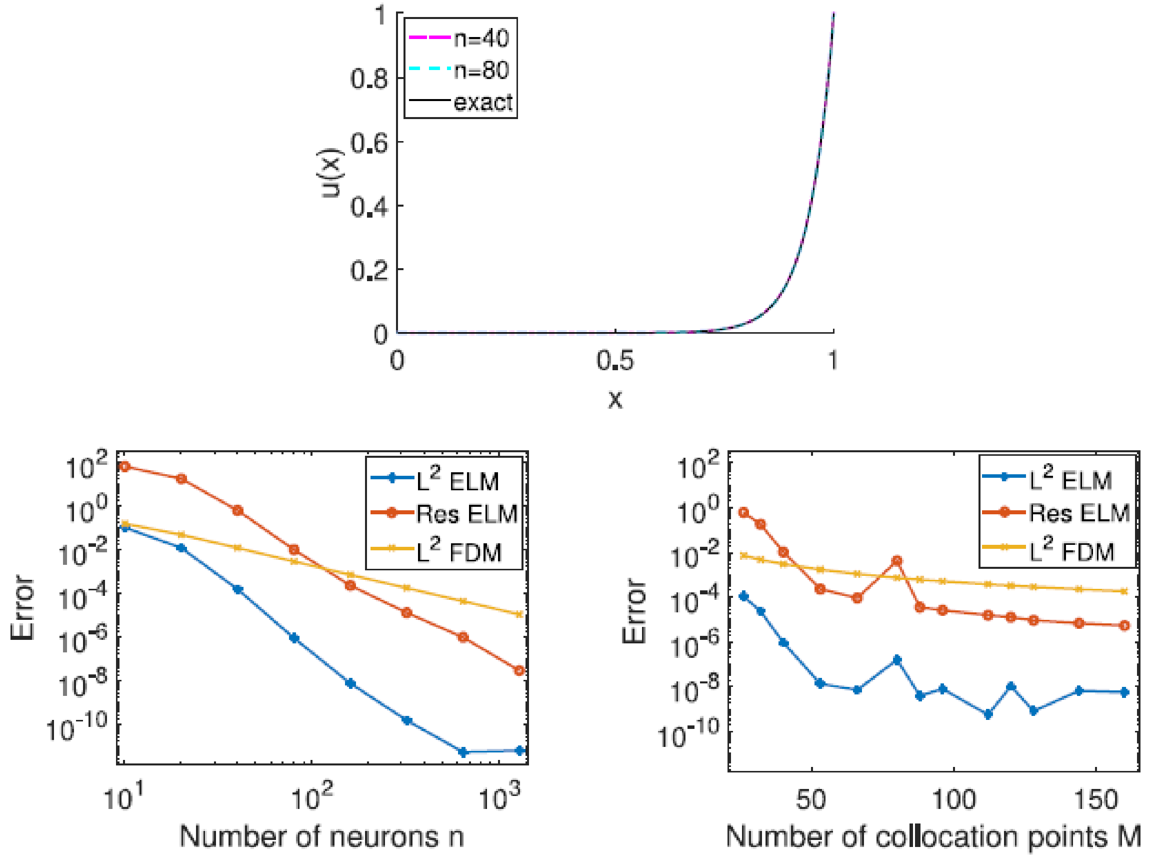


Fig. 7. Numerical tests for the solution of the problem (3.1) with $\mu = 1, \gamma = 0, \lambda = 300$. In this case, the Péclet number is $\mathbb{P}e_g = 50$, the boundary layer is of the order of 10^{-1}. On the top panel are shown the exact-analytical (5.6) and the ELM numerical solutions for $n = 40, 80$; the number of collocation points was set to $M = n/2$. On the bottom panels are depicted the error and the residual convergence: on the left panel we have fixed $M = n/2$ and varied n and on the right panel we have fixed $n = 80$ and varied M.

Extreme learning machine collocation method

Ex3: Nonlinear Burgers equation

$$\nu \frac{\partial^2 u}{\partial x^2} - u \frac{\partial u}{\partial x} = 0$$

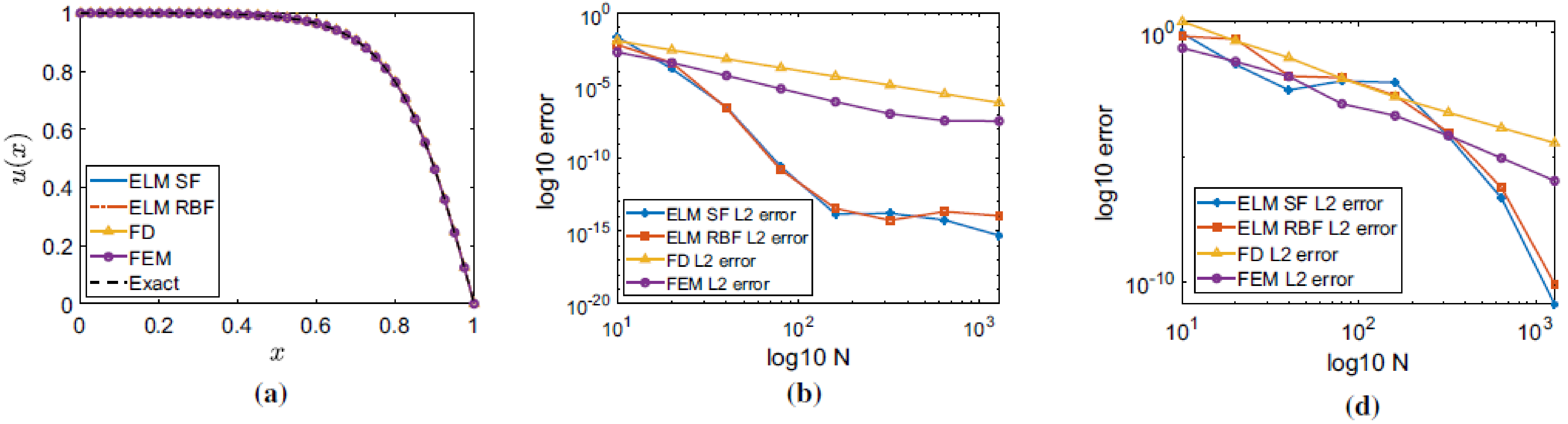


Fig. 1 Numerical solution and accuracy of the FD, FEM and the proposed machine learning (ELM) schemes for the one-dimensional viscous Burgers problem with Dirichlet boundary conditions (18), (19), (a,b) with viscosity $\nu = 0.1$: (a) Solutions for a fixed problem size $N = 40$; (b) L_2-norm of differences with respect to the exact solution (21) for various problem sizes. (c,d) with viscosity $\nu = 0.007$: (c) Solutions for a fixed problem size $N = 40$; (d) L_2-norm errors with respect to the exact solution for various problem sizes

Extreme learning machine collocation method

Ex4: Nonlinear Bratu problem, bifurcation diagram

$$\Delta u(x) + \lambda e^{u(x)} = 0 \;\; x \in \Omega,$$

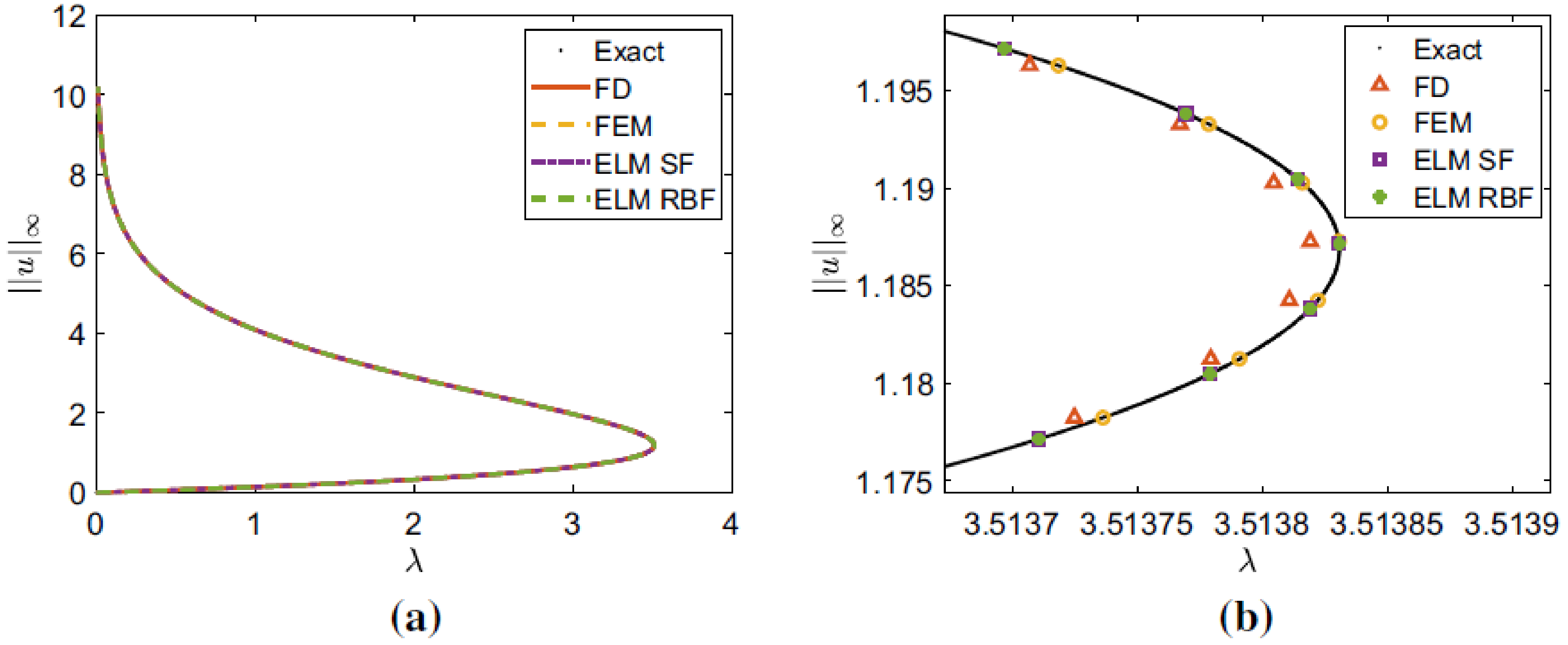


Fig. 7 **a** Bifurcation diagram for the one-dimensional Bratu problem (37), with a fixed problem size $N = 400$. **b** Zoom near the turning point

Extreme learning machine collocation method

Ex4: Nonlinear Bratu problem, convergence

$$\Delta u(\boldsymbol{x}) + \lambda e^{u(\boldsymbol{x})} = 0 \ \ \boldsymbol{x} \in \Omega ,$$

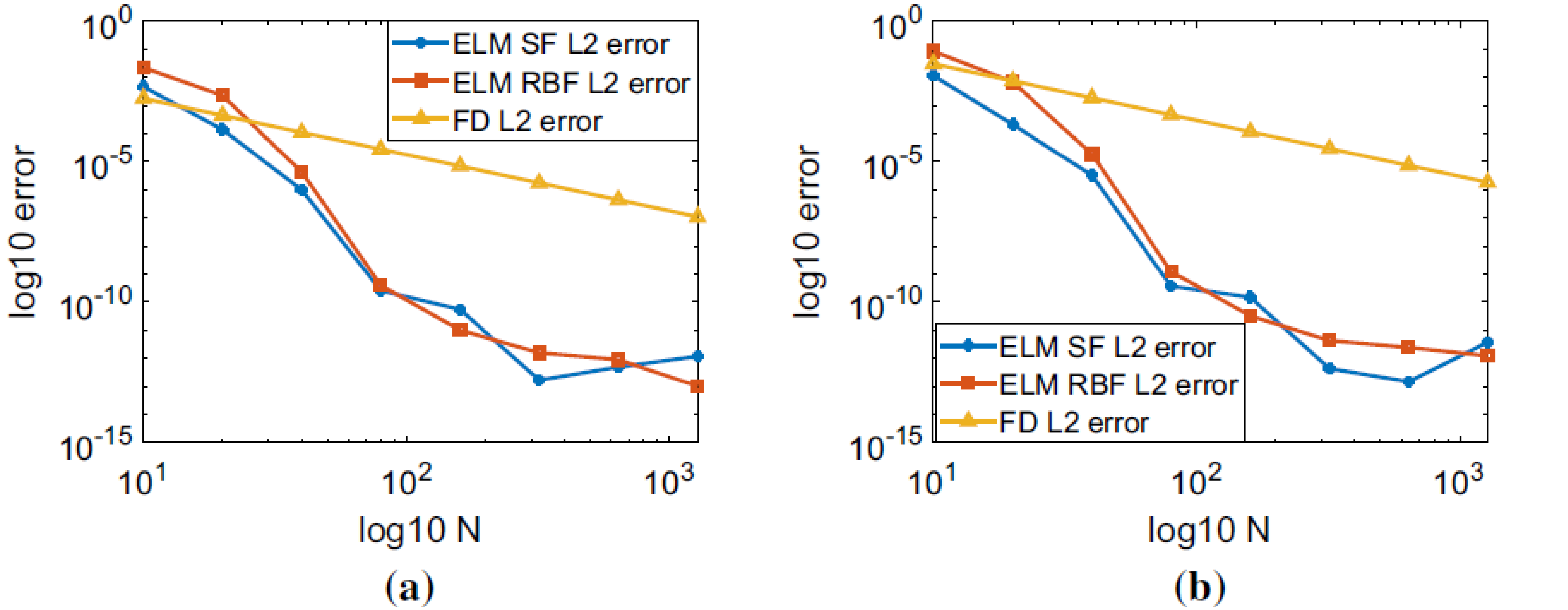


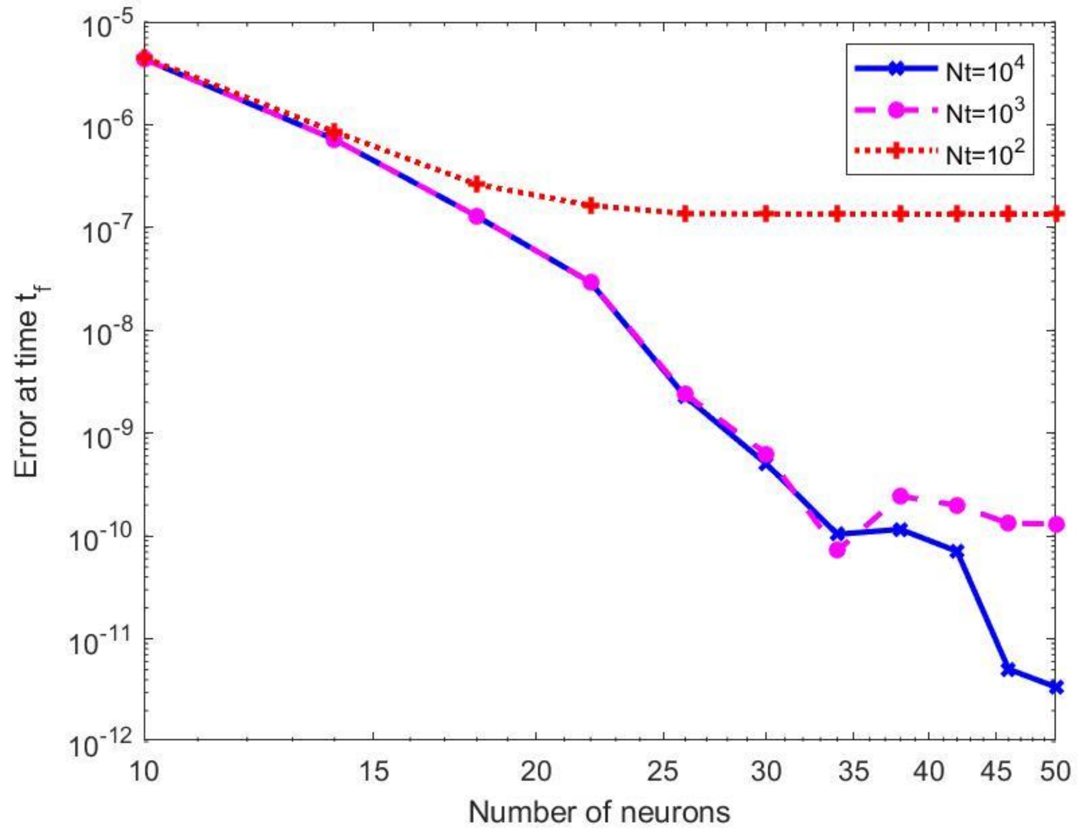
Fig. 6 Fixed point iterations: L_2-norm of the difference errors for the low and up branch Liouville–Bratu–Gelfand solution (39) for $\lambda = 2$: **a** L_2 errors with respect to N of the low branch solution, **b** L_2 errors with respect to N of the upper branch

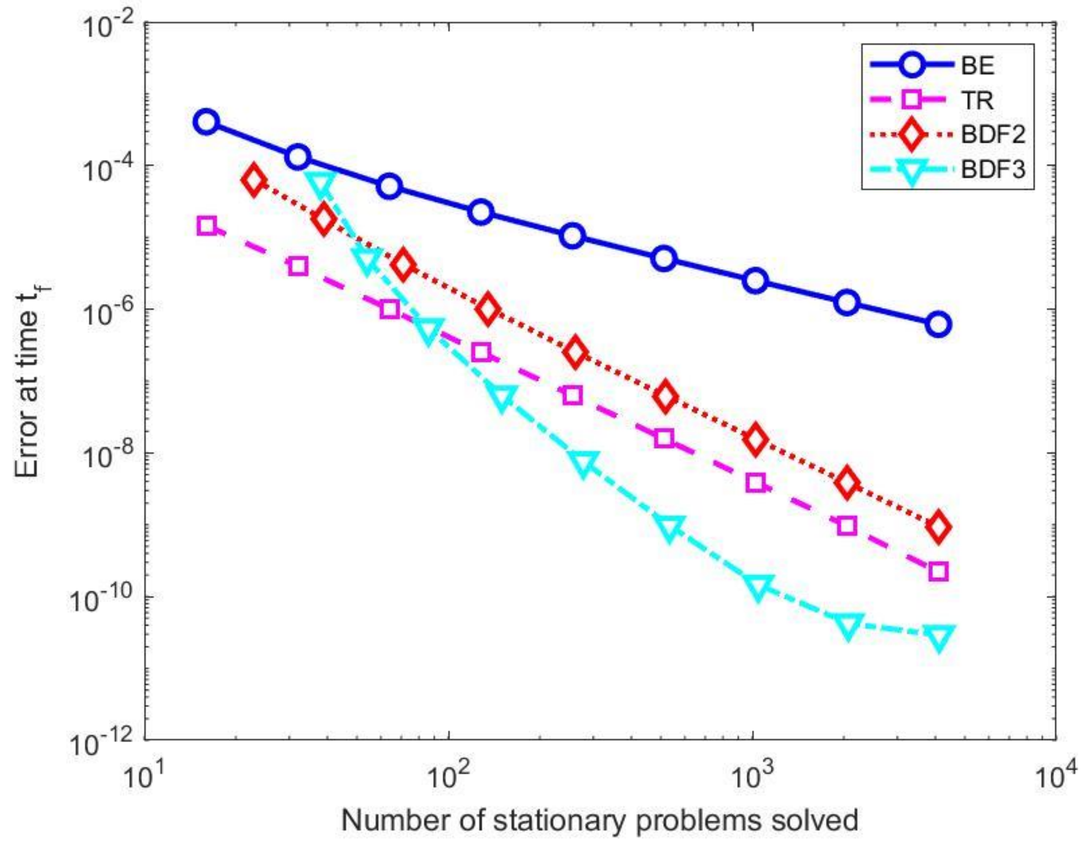
Extreme learning machine collocation method

Ex5: Parabolic problem

$$\begin{cases} \dfrac{\partial u}{\partial t} = \dfrac{\partial^2 u}{\partial x^2} & x \in [0,1]\,,\ \ t \in [0,1] \\ u(x,0) = \sin(\pi x) + \sin(3\pi x) & u(0,t) = u(2,t) = 0 \end{cases}$$

$$u(t,x) = e^{-\pi^2 t} \sin(\pi x) + e^{-3^2\pi^2 t} \sin(3\pi x)$$





Extreme learning machine collocation method

Remarks:

- The LS solution, in principle, can give some non zero residual on the collocation points. Nevertheless, we consider L^2 errors for the global solution and obtain good accuracy with very reasonable computational costs.
- Boundary terms can be penalized as usually done in collocation.
- $\tilde{u}(x; w)$ is a physics-informed neural network (PINN) that minimizes a strong-form loss function. Because we are using ELMs such problem is a linear underdeterminated problem.
- In the case of general NN, the function $v(x; A, \beta; w)$ depends non linearly by its parameters, so that some optimization procedure has to be performed.

Extreme learning machine collocation method

Main advantages / open questions

Collocation solution is Physic based, directly a global solution, no post processing is needed for evaluation of the solution. Data can be given in user-specified sites and/or can be given by the application.
The least square solution is capable of automatically selecting the important features, i.e. the functions in the space that are more influent for the solution. This leads to a one-shot automatic method and there is no need for adaptive procedures, such as those related to a-posteriori error estimates.

Achieved:

- ✓ No need for mesh generation.
- ✓ No need for a regular distribution of the collocation points.
- ✓ No problems related to difficult geometries.
- ✓ Good adaptivity.
- ✓ Good accuracy.
- ✓ BDF methods for tme marching.

Needs more investigation:

- ➢ Convergence properties are unavailable because usually, these are related to polynomial reproduction.
- ➢ Advantages, when used in variational formulations, are not clear (*quadrature?*).
- ➢ Treatment of local behaviors such as boundary conditions should be clarified: in the actual computations, many points on the boundaries are needed.

Thank You

For Your Attention

Prof. Francesco Calabrò
calabro@unina.it - +39 347 925 8767
Associate Professor in Numerical Analysis
University of Naples Federico II
Dipartimento di Matematica e Applicazioni "R. Caccioppoli"
Complesso Universitario Monte S. Angelo
Via Cintia 80126 Napoli (Italy)
https://www.docenti.unina.it/francesco.calabro
http://scholar.google.com/citations?user=MCjA5-4AAAAJ
https://www.linkedin.com/in/francesco-calabr%C3%B2-19ba0b5/

Related works on the use of ELMs for PDEs problems

Professor **Muzhou Hou**, the School of Mathematics and Statistics, Central South University, China

- H. Sun, M. Hou, Y. Yang, T. Zhang, F. Weng, F. Han, Solving partial differential equation based on bernstein neural network and extreme learning machine algorithm, *Neural Processing Letters* (2018) 1–20 .
- Yunlei Yang, Muzhou Hou, Hongli Sun, Tianle Zhang, Futian Weng & Jianshu Luo, Neural network algorithm based on Legendre improved extreme learning machine for solving elliptic partial differential equations, *Soft Computing* volume 24, pages1083–1096 (2020)

Professor **Balaji Srinivasan**, IIT Madras, India

- V Dwivedi, B Srinivasan, Physics Informed Extreme Learning Machine (PIELM)–A rapid method for the numerical solution of partial differential equations, *Neurocomputing* 391, 96-118 2020
- V Dwivedi, B Srinivasan, Solution of biharmonic equation in complicated geometries with physics informed extreme learning machine, *Journal of Computing and Information Science in Engineering* 20 (6) 4 2020
- V Dwivedi, N Parashar, B Srinivasan, Distributed learning machines for solving forward and inverse problems in partial differential equations *Neurocomputing* 420, 299-316 10 2021

Professor **Manoj Kumar**, Motilal Nehru National Institute of Technology Allahabad, Uttar Pradesh, India

- Panghal, Shagun, and Manoj Kumar. "Optimization free neural network approach for solving ordinary and partial differential equations." *Engineering with Computers* (2020): 1-14.

Professor **Suchuan Dong**, Purdue University, USA

- S. Dong & Z. Li. Local extreme learning machines and domain decomposition for solving linear and nonlinear partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 387, 114129, 2021

Professor **Roberto Furfaro** The University of Arizona, USA & Professor **Daniele Mortari**, Texas A&M University, USA

- E Schiassi, R Furfaro, C Leake, M De Florio, H Johnston, D. Mortari, Extreme theory of functional connections: A fast physics-informed neural network method for solving ordinary and partial differential equations *Neurocomputing* 457 (2021): 334-356.