Convergence Analysis for 1D Neural Network Method

César Herrera[1]
Zhiqiang Cai[1] Anastassia Doktorova[1] Robert D. Falgout[2]

[1]Department of Mathematics, Purdue University
[2]Lawrence Livermore National Laboratory

Finite Element Circus
April 2025

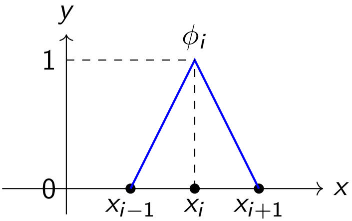
Table of Contents

Free Knot Linear Splines

- C^0 **piecewise linear functions on a fixed mesh in** $[0,1]$**:**

$$S_1^0(\Delta) = \left\{ \sum_{i=1}^{n} c_i \phi_i(x) : c_i \in \mathbb{R} \right\},$$

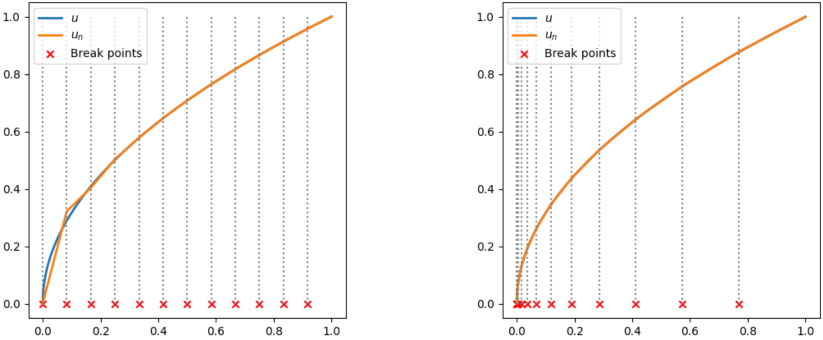
where



- C^0 **piecewise linear functions on a moving mesh in** $[0,1]$**:**

$$S_1^0(n) = \left\{ \sum_{i=1}^{n} c_i \phi_i(x; x_{i-1}, x_i, x_{i+1}) : c_i \in \mathbb{R}, x_i \in [0,1] \right\}$$

Free Knot Linear Splines



(a) **Fixed** uniform mesh

(b) **Moving** mesh

Figure: Continuous piecewise linear approximation to $u(x) = \sqrt{x}$ with 12 breakpoints.

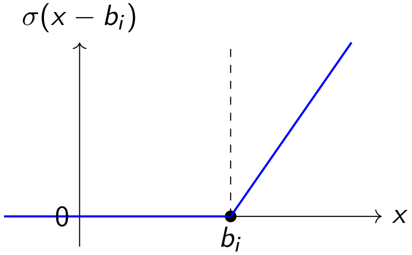
Free Knot Linear Splines

- **In fact the error, is (L^∞ norm)**
  - $\mathcal{O}(n^{1/2})$ **on a fixed uniform mesh.**
  - $\mathcal{O}(n)$ **on a moving mesh.**
- **Drawbacks**
  1. **Finding optimal breaking points $\leftrightarrow$ solving a high-dimensional non-convex optimization problem**
  2. **2D or 3D free knot splines?**

Shallow Neural Networks

- **One-dimensional shallow neural network:**

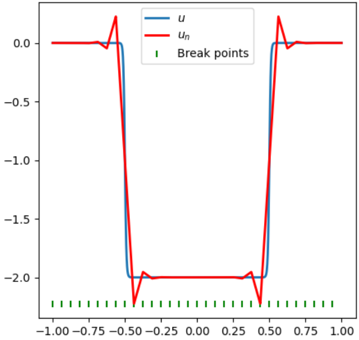
$$\mathcal{M}_n([0,1]) = \mathcal{M}_n(I) = \left\{ c_0 + \sum_{i=1}^{n} c_i \sigma(x - b_i) : c_i \in \mathbb{R}, b_i \in [0,1] \right\}$$



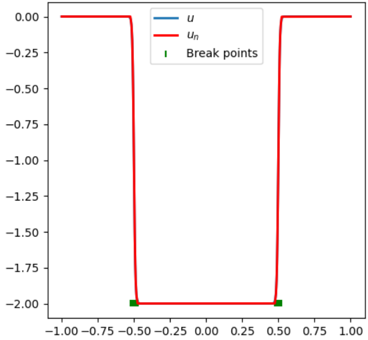
- **Free knot linear splines and shallow neural networks are equivalent[1].**

[1] I. Daubechies et al. "Nonlinear Approximation and (Deep) ReLU Networks". English (US). in: *Constructive Approximation* 55.1 (Feb. 2022), pp. 127–172. ISSN: 0176-4276. DOI: 10.1007/s00365-021-09548-z.

Why Neural Networks?



(a) **Fixed** uniform mesh



(b) **Moving** mesh

Figure: Singularly perturbed reaction-diffusion equation approximated by NN with 32 breakpoints.

Table of Contents

Best Neural Network Approximation

For a function

$$u_n \in \mathcal{M}_n(I) = \left\{ c_0 + \sum_{i=1}^{n} c_i \sigma(x - b_i) : c_i \in \mathbb{R}, b_i \in [0,1] \right\}.$$

Linear parameters: $\mathbf{c} = (c_0, \ldots, c_n)^T \in \mathbb{R}^{n+1}$
Nonlinear parameters: $\mathbf{b} = (b_1, \ldots, b_n)^T \in \mathbb{R}^n$
Optimizable parameters: $\mathbf{r} = (c_0, c_1, \ldots, c_n, b_1, \ldots, b_n)^T \in \mathbb{R}^{2n+1}$
Best Neural Network Approximation: Given $J : \mathbb{R}^{2n+1} \to \mathbb{R}$, find $\mathbf{r}^* \in \mathbb{R}^{2n+1}$ such that

$$J(\mathbf{r}^*) = \min_{\mathbf{r} \in \mathbb{R}^{2n+1}} J(\mathbf{r}).$$

Examples of Functionals

1. For the one-dimensional diffusion problem

$$\begin{cases} -u''(x) = f(x), & x \in I = (0,1), \\ u(0) = 0, \quad u(1) = 0 & \end{cases}$$

   Best Ritz Approximation: Minimize

$$J(v) = \frac{1}{2}\int_0^1 \left(v(x)'\right)^2 dx - \int_0^1 f(x)v(x)dx$$

2. Given $u \in L^2([0,1])$.
   Nonlinear Least-Squares: Minimize

$$J(v) = \frac{1}{2}\int_0^1 \left(v(x) - u(x)\right)^2 dx$$

Table of Contents

Optimality Conditions

Given a twice differentiable $J : \mathbb{R}^{2n+1} \to \mathbb{R}$ and $\mathbf{r}^* = \begin{pmatrix} \mathbf{c}^* \\ \mathbf{b}^* \end{pmatrix}$ such that

$$J(\mathbf{r}^*) = \min_{\mathbf{r} \in \mathbb{R}^{2n+1}} J(\mathbf{r}).$$

Then optimality conditions yield

$$\nabla_{\mathbf{c}} J\left(\mathbf{r}^*\right) = \mathbf{0} \quad \text{and} \quad \nabla_{\mathbf{b}} J\left(\mathbf{r}^*\right) = \mathbf{0}.$$

Optimality Conditions (Elliptic PDEs)

For a given fixed $\mathbf{b}$

$$\nabla_{\mathbf{c}} J(\mathbf{r}) = \nabla_{\mathbf{c}} J(\mathbf{c}, \mathbf{b}) = 0$$

is a system of **linear** equations for $\mathbf{c}$.

Difficulties:

- **Coefficient matrix**

$$A(\mathbf{b}) = \int_0^1 \mathbf{H}\mathbf{H}^T dx = \left(\int_0^1 \sigma'(x - b_i)\sigma'(x - b_j) dx \right)_{ij}$$

- **Mass matrix**

$$M(\mathbf{b}) = \int_0^1 \mathbf{\Sigma}\mathbf{\Sigma}^T dx = \left(\int_0^1 \sigma(x - b_i)\sigma(x - b_j) dx \right)_{ij}$$

are both **dense** and **ill-conditioned**.

Optimality Conditions (Elliptic PDEs)

For a given fixed $\mathbf{c}$

$$\nabla_{\mathbf{b}} J(\mathbf{r}) = \nabla_{\mathbf{b}} J(\mathbf{c}, \mathbf{b}) = 0$$

is a system of **nonlinear** algebraic equations for $\mathbf{b}$.

Difficulties:

- ReLU is **not differentiable** everywhere.
- The Hessian matrix $\nabla^2_{\mathbf{b}} J(\mathbf{r})$ could be **singular**.

How to overcome these difficulties?

- **arXiv:2404.17750**
- **arXiv:2407.01496**

Block Newton Methods

Require: Initial network parameters $\mathbf{c}^{(0)}, \mathbf{b}^{(0)}$, and target function J
Ensure: Network parameters $\mathbf{c}, \mathbf{b}$

for $k = 0, 1 \ldots$ **do**

▷ *Linear parameters*

$$\mathbf{c}^{(k+1)} \leftarrow \mathbf{c}^{(k)} - \left[\nabla^2_{\mathbf{c}} J(\mathbf{c}^{(k)}, \mathbf{b}^{(k)})\right]^{-1} \nabla_{\mathbf{c}} J(\mathbf{c}^{(k)}, \mathbf{b}^{(k)})$$

▷ *Nonlinear parameters*

$$\mathbf{b}^{(k+1)} \leftarrow \mathbf{b}^{(k)} - \left[\nabla^2_{\mathbf{b}} J(\mathbf{c}^{(k+1)}, \mathbf{b}^{(k)})\right]^{-1} \nabla_{\mathbf{b}} J(\mathbf{c}^{(k+1)}, \mathbf{b}^{(k)})$$

end for

Table of Contents

Fixed Point Iteration

The method outlined before can be considered as a **fixed point iteration**

$$\mathbf{r}^{k+1} = G(\mathbf{r}^k) \quad \text{for } k = 0, 1 \ldots .$$

for some differentiable function $G : \mathbb{R}^{2n+1} \to \mathbb{R}^{2n+1}$.

Local convergence condition[2]**:** For a minimizer $\mathbf{r}^*$, this method is locally convergent if

$$\|\mathbf{J}_G(\mathbf{r}^*)\| = \sigma < 1. \tag{1}$$

Questions:

1. How can we compute $\mathbf{J}_G(\mathbf{r}^*)$ for our particular G?
2. Under what conditions does (1) hold?

---

[2] J.M. Ortega and W.C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, 1970.

Jacobian of G and Convergence Condition

Hessian Matrix

$$\nabla^2 J(\mathbf{r}) = \begin{pmatrix} \nabla_{\mathbf{c}} \left(\nabla_{\mathbf{c}} J(\mathbf{r})\right)^T & \nabla_{\mathbf{c}} \left(\nabla_{\mathbf{b}} J(\mathbf{r})\right)^T \\ \nabla_{\mathbf{b}} \left(\nabla_{\mathbf{c}} J(\mathbf{r})\right)^T & \nabla_{\mathbf{b}} \left(\nabla_{\mathbf{b}} J(\mathbf{r})\right)^T \end{pmatrix} = \begin{pmatrix} \mathcal{A}_{11}(\mathbf{r}) & \mathcal{A}_{12}(\mathbf{r}) \\ \mathcal{A}_{21}(\mathbf{r}) & \mathcal{A}_{22}(\mathbf{r}) \end{pmatrix}.$$

Lemma

If $\mathbf{r}^*$ is a local minimum and $\nabla^2 J(\mathbf{r}^*)$ is s.p.d. then

$$\mathbf{J}_G(\mathbf{r}^*) = I - \begin{pmatrix} \mathcal{A}_{11}(\mathbf{r}^*) & \mathbf{0} \\ \mathcal{A}_{21}(\mathbf{r}^*) & \mathcal{A}_{22}(\mathbf{r}^*) \end{pmatrix}^{-1} \nabla^2 J(\mathbf{r}^*).$$

Local Convergence Condition: $\nabla^2 J(\mathbf{r}^*)$ is s.p.d.

Table of Contents

1D Diffusion Problem

For the one-dimensional diffusion problem

$$\begin{cases} -u''(x) = f(x), & x \in I = (0,1), \\ u(0) = 0, \quad u(1) = 0. \end{cases}$$

Best Ritz Approximation: Minimize

$$J(v) = \frac{1}{2}\int_0^1 \left(v(x)'\right)^2 dx - \int_0^1 f(x)v(x)dx$$

1D Diffusion Problem

Hessian Matrix (at the minimizer $\mathbf{r}^*$)

$$\nabla^2 J(\mathbf{r}) = \begin{pmatrix} \mathcal{A}_{11}(\mathbf{r}^*) & \mathcal{A}_{22}(\mathbf{r}^*) \\ \mathcal{A}_{21}(\mathbf{r}^*) & \mathcal{A}_{22}(\mathbf{r}^*) \end{pmatrix},$$

where

- **Hessian of nonlinear parameters:**

$$\mathcal{A}_{22}(\mathbf{r}^*) = \nabla^2_{\mathbf{b}} J(\mathbf{b}) = -\mathrm{diag}(c_1 f(b_1), c_2 f(b_2), \ldots, c_n f(b_n))$$

- **Invertibility condition:**

$$c_i f(b_i) \neq 0.$$

Convergence Condition

- **Local convergence condition:**

$$|c_i f(b_i)| \geq \frac{3c_i^2}{2h_i}, \tag{2}$$

where $h_i = \min\{b_{i+1} - b_i, b_i - b_{i-1}\}$.

- **Fixing neurons:** If some optimal b_i does not satisfy (2) that neuron will be **fixed** and **local convergence** is **garanteed**.

Numerical Experiment

Approximating the function

$$u(x) = x\left(\exp\left(-\frac{(x-\frac{1}{3})^2}{0.01}\right) - \exp\left(-\frac{4}{9\times 0.01}\right)\right)$$

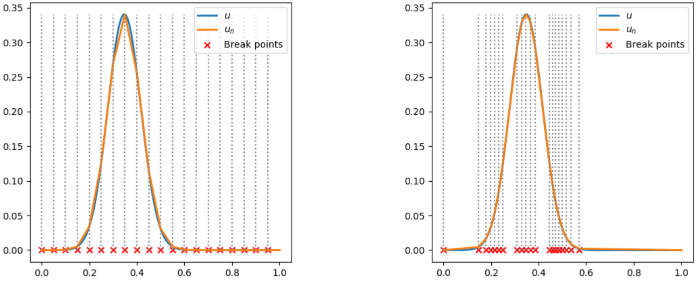


Figure: $u(x)$ approximated by NN. Left: 20 uniform breakpoints, $e_n = 0.250$. Right: optimized NN model with 20 breakpoints, 100 iterations, $e_n = 0.102$.

Summary and Final Remarks

- NN Methods were analized as fixed point iterations.
- **Sufficient condition:** $\nabla^2 J(\mathbf{r}^*)$ s.p.d.
- **Explicit formulas** for $\nabla^2 J(\mathbf{r}^*)$ in:
  - 1D diffusion problems
  - 1D diffusion-reaction problems
  - 1D least-squares approximation
- Conditions incorporated into the **algorithm implementation**.

Thanks!

For more details, see

- Z. Cai, A. Doktorova, R. D. Falgout, and C. Herrera. **Efficient Shallow Ritz Method For 1D Diffusion Problems.** **arXiv:2404.17750**, 2024.
- Z. Cai, A. Doktorova, R. D. Falgout, and C. Herrera. **Fast Iterative Solver For Neural Network Method: II. 1D Diffusion-Reaction Problems And Data Fitting.** **arXiv:2407.01496**, 2024.
- Z. Cai, A. Doktorova, R. D. Falgout, and C. Herrera. **Convergence Analysis for 1D Neural Network Method.** **To be submitted**.