Computational Mechanics (2023) 72:333–362
https://doi.org/10.1007/s00466-023-02336-5

ORIGINAL PAPER

Convolution Hierarchical Deep-learning Neural Networks (C-HiDeNN): finite elements, isogeometric analysis, tensor decomposition, and beyond

Ye Lu[1,7] · Hengyang Li[1] · Lei Zhang[2,3] · Chanwook Park[1] · Satyajit Mojumder[4] · Stefan Knapik[1] · Zhongsheng Sang[1] · Shaoqiang Tang[5] · Daniel W. Apley[6] · Gregory J. Wagner[1] · Wing Kam Liu[1]

Received: 26 November 2022 / Accepted: 30 March 2023 / Published online: 16 May 2023

Abstract
This paper presents a general Convolution Hierarchical Deep-learning Neural Networks (C-HiDeNN) computational framework for solving partial differential equations. This is the first paper of a series of papers devoted to C-HiDeNN. We focus on the theoretical foundation and formulation of the method. The C-HiDeNN framework provides a flexible way to construct high-order C^n approximation with arbitrary convergence rates and automatic mesh adaptivity. By constraining the C-HiDeNN to build certain functions, it can be degenerated to a specification, the so-called convolution finite element method (C-FEM). The C-FEM will be presented in detail and used to study the numerical performance of the convolution approximation. The C-FEM combines the standard C^0 FE shape function and the meshfree-type radial basis interpolation. It has been demonstrated that the C-FEM can achieve arbitrary orders of smoothness and convergence rates by adjusting the different controlling parameters, such as the patch function dilation parameter and polynomial order, without increasing the degrees of freedom of the discretized systems, compared to FEM. We will also present the convolution tensor decomposition method under the reduced-order modeling setup. The proposed methods are expected to provide highly efficient solutions for extra-large scale problems while maintaining superior accuracy. The applications to transient heat transfer problems in additive manufacturing, topology optimization, GPU-based parallelization, and convolution isogeometric analysis have been discussed.

Keywords Convolution FEM and HiDeNN · Tensor decomposition · Reduced order modeling · Additive manufacturing · High-order smoothness · Isogeometric analysis (IGA)

1 Introduction

Numerical analysis using the finite element method (FEM) has been widely used for the understanding and design of complex scientific and engineering problems. The demands for high-fidelity solutions and faster computations have been crucial with the increasing degrees of freedom of the discretized numerical systems and the complexity of the physical problems. In addition, higher-order smoothness and convergence rates have been demanded since the invention of FEM [1]. This has motivated the development of the meshfree-type approximations [2–10] and the isogeometric analysis in the recent past [11–13].

Unlike FEM, meshfree methods only require a set of nodes in the domain of interest and do not have real connectivi-

✉ Ye Lu
yelu@umbc.edu

✉ Wing Kam Liu
w-liu@northwestern.edu

1 Department of Mechanical Engineering, Northwestern University, Evanston, IL, USA

2 The State Key Laboratory of Nonlinear Mechanics, Institute of Mechanics, Chinese Academy of Sciences, Beijing, China

3 School of Engineering Science, University of Chinese Academy of Sciences, Beijing, China

4 Theoretical and Applied Mechanics Program, Northwestern University, Evanston, IL, USA

5 HEDPS and LTCS, College of Engineering, Peking University, Beijing, China

6 Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, USA

7 Department of Mechanical Engineering, University of Maryland Baltimore County, 1000 Hilltop Cir, Baltimore, MD 21250, USA

ties between nodes. They provide an alternative way to FEM for the modeling and analysis of problems where there is a significant or discontinuous change of the geometry, such as crack propagation, penetration, and large deformation. Some of the important developments of this type are diffuse element method [6], element free Galerkin methods [14], reproducing kernel particle methods [2, 3], h-p clouds [15, 16]. An important feature that meshfree methods use for constructing the approximation function basis is the partition of unity, which has also enabled the development of the so-called Generalized FEM, independently initiated by Duarte and Oden (under the name of cloud-based hp FEM [15, 17]) and Babuška and Melenk (under the name of partition of unity FEM [18, 19]).

However, due to the lack of Kronecker delta property, meshfree methods usually need special treatments to enforce Dirichlet boundary conditions and lose optimal convergence rates compared to FEM. Many techniques have been developed to resolve this issue, e.g., Lagrangian multiplier technique [5], transformation technique [20], reproducing kernel interpolation technique [21], collocation technique [22], and including the coupling of the FEM and meshfree methods, like the hierarchical enrichment technique [23, 24] and the continuous blending method [25, 26]. Nevertheless, the methods do not have good scalability in parallel computations, as stated in [27]. The so-called reproducing kernel element method (RKEM) [27–29] has been since developed. In RKEM, a modified global FE shape function is combined with the reproducing kernel to construct a novel high-order interpolant. This method maintains the basic idea of finite elements and does not require any special treatment for boundary conditions.

In this work, we propose a more general framework for solving partial differential equations (PDEs), i.e., Convolution Hierarchical Deep-learning Neural Networks (C-HiDeNN), that generalizes the idea of RKEM and can degenerate to traditional FEM and meshfree methods. Therefore, the proposed framework provides a very flexible way to unify the two types of methods and has a great potential to adaptively maintain the approximation capability of each method in the computational domain. This is the first paper of a series in the special issue devoted to C-HiDeNN. This work focuses on the theoretical foundation and formulation of the method.

Application of deep learning neural networks (DeNN) for solving PDEs has attracted much interest in recent years. The universal approximation theorem [30] demonstrates that neural networks can approximate any continuous function, as long as they are sufficiently complex. This has encouraged many researchers and scientists to use neural networks as an approximator to PDE solutions. Physics-Informed Neural Networks (PINN) [31, 32] proposed to use directly physics-based PDEs as loss functions to train the parameters of neural networks for solutions. To preserve some desired properties (like partition of unity) of traditional approximation functions of PDE solution, the so-called partition of unity networks has been proposed [33]. Other PINN related methods like structure-preserving networks [34, 35], representative volume element networks [36] have also been explored. A comprehensive review of PINNs can be found in [37]. Nevertheless, the training procedure for PINNs is still computational challenging due to the large number of unknown parameters and the high non-linearity and non-convexity of the optimization problem. Furthermore, solving the numerical solution by directly minimizing the PDE-based residuals may require additional treatments for boundary conditions. This sometimes poses difficulties to enforce the exact required boundary conditions.

Unlike PINN, the C-HiDeNN is based on the weak form of the physical problem and does not require additional techniques for boundary conditions. The parameters of networks (weights and biases) are predefined in an explicit way and remain unchanged during the solution procedure. This avoids the expensive training process of traditional neural networks. In addition, higher-order smooth solutions with arbitrary convergence rates can be obtained easily by adjusting the parameters in the C-HiDeNN approximation without modifying the mesh. As an extension of our previous work on HiDeNN-FEM [38], the C-HiDeNN combines the general polynomial approximation (which can be the HiDeNN-FEM shape function) with a convolution patch function (which can be the meshfree type approximation) to achieve higher order continuity in the C-HiDeNN interpolant. Like the HiDeNN method, C-HiDeNN has the potential to automatically optimize the mesh coordinates (r-adpativity). In addition, the C-HiDeNN introduces a novel dilation parameter into the approximation, leading to a built-in length scale filter as meshfree methods [3]. The spatially adpatitive selection of this dilation parameter enables the so-called a-adaptivity. The dilation parameter is also found to have effects of regularization and numerical viscosity that can enhance the numerical stability in nonlinear analysis.

We will discuss the numerical performance based on a special example of C-HiDeNN, i.e., the Convolution FEM (C-FEM). In particular, the C-FEM combines the traditional FE shape function and radial basis interpolation and does not increase the overall degrees of freedom (DoFs) of the numerical system compared to FEM. It is shown that the C-FEM can give more accurate results than FEM with different controllable parameters that dictate the convergence rates. Orders of magnitude improvement in terms of accuracy have been achieved with C-FEM. This implies that C-FEM can use a very coarse mesh with much fewer DoFs for the same level of accuracy as FEM, therefore reducing the underlying computational cost while keeping high-fidelity solutions.

The convolution shape function can be directly applied to reduced order modeling techniques [39–44] to further accelerate the simulations. We will discuss the so-called convolution tensor decomposition (C-TD) method, based on the separation of spatial variables. The method can overcome the exponential growth of DoFs in large size problems and can be more accurate than FEM with orders of magnitude speedups. Hence, it has great potential to enable extra-large scale computing for, e.g., additive manufacturing process simulations, multiscale analysis of composite materials, and high-resolution topology optimization. The future development and applications of C-HiDeNN related to the graphics processing unit (GPU) computing and topology optimization are discussed in detail in [45, 46].

In summary, the proposed C-HiDeNN has following key features:

- General computational framework for solving PDEs with potentially automatic r- and a- adaptivity
- Arbitrary high order smoothness without modifying the mesh
- Convolution approximation based highly accurate and efficient reduced order models
- Built-in length scale control and enhanced numerical stability for nonlinear problems with the dilation parameter

The paper is organized as follows. We start with a general concept of convolution approximation in Sect. 2. The concept of C-HiDeNN is discussed in Sect. 3. The convolution reduced-order modeling methods is presented in Sect. 4. Section 5 presents the numerical results of the convolution methods and a parametric study on the controlling parameters. Then we briefly talk about the future development and applications of the method in Sect. 6. The paper will be closed with some concluding remarks.

2 General concept of convolution approximation

Before introducing the Convolution HiDeNN approximation, we present first the general idea of an element with the convolution approximation. Let us consider a traditional linear finite element with an approximation $u(\boldsymbol{\xi}) = \sum_{i\in A^e} N_i(\boldsymbol{\xi})u_i$, where $\boldsymbol{\xi}$ is the general parametric coordinate, A^e is the nodal set of the element, N_i is the shape function, u_i is the nodal coefficient. Then a convolution operation can be defined between $u(\boldsymbol{\xi})$ and a given smooth function $W(\boldsymbol{\xi})$ as

$$(u * W)(\boldsymbol{\xi}) = \int_{\Omega_{\text{patch}}} u(\boldsymbol{t})W(\boldsymbol{t}-\boldsymbol{\xi})d\boldsymbol{t} \tag{1}$$

where Ω_{patch} is the convolution patch domain, which can span outside an element. Since W can be an arbitrary high order smooth function, the resultant approximation u^c reads

$$u^c(\boldsymbol{\xi}) = (u * W)(\boldsymbol{\xi}) \tag{2}$$

This way, a convolution element can be defined with the novel approximation u^c. Figure 1 illustrates this concept in a discretized manner. As we can see, the element domain remains the same as the original element with the coordinate $\boldsymbol{\xi}$, and the mesh is not modified by the convolution operation. As a result, the overall degrees of freedom do not increase for the convolution approximation. However, we can notice that the element-wise connectivity is different from the traditional finite element, the final convolution shape function (assuming $\tilde{N}_k$ is known) should reside in the convolution patch domain, which reads

$$u^c(\boldsymbol{\xi}) = \sum_{k\in\Omega_{\text{patch}}} \tilde{N}_k(\boldsymbol{\xi})u_k \tag{3}$$

This approximation concept is the foundation of the proposed Convoluiton HiDeNN method and can be generalized to have adaptive patch sizes and different types of meshes such as isogeometric analysis [11]. As mentioned in the Introduction, similar ideas of convolution approximation were adopted in meshfree methods for high order approximations. We will present in this work how to construct a general convolution approximation and related variants with flexibility to degenerate to traditional finite element or meshfree methods (at least in the sense of approximation accuracy).

3 Convolution HiDeNN: theory and numerical implementation

Based on the previously presented convolution approximation concept, we introduce in this section the Convolution HiDeNN method, including the theoretical formulation and the numerical implementation aspects. Furthermore, we will discuss the theoretical convergence and the flexibility of constructing the high-order C^n continuity shape function in the C-HiDeNN framework.

3.1 General C-HiDeNN approximation

The C-HiDeNN can be seen as a general partially connected DeNN for solving partial differential equations with convolution element. Since DeNN has the universal approximation capability, we can define the C-HiDeNN approximation using two general functions, i.e., a general piece-wise polynomial function (e.g., finite element basis function) and a local patch function that operates at each nodal point as a

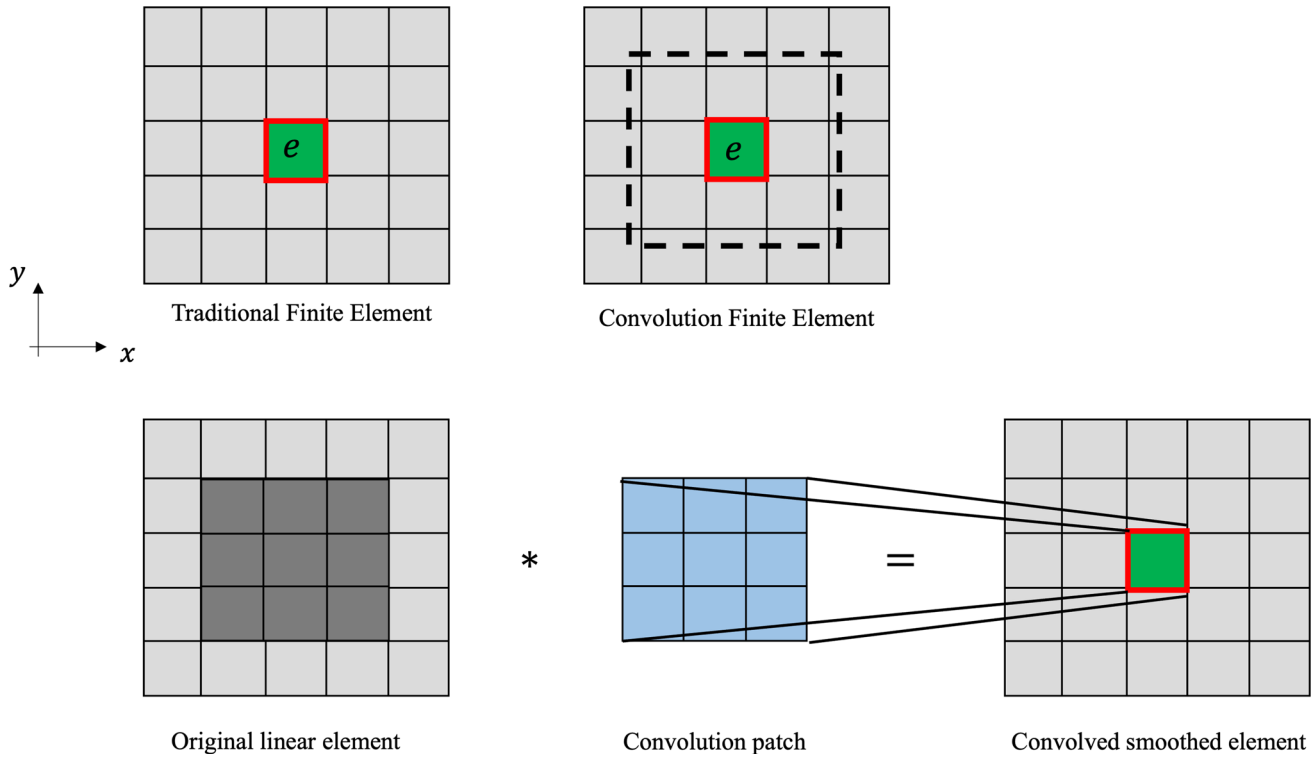


Fig. 1 Illustration of the convolution element

convolution operator. Assuming these functions are given and ignoring the implementation aspect, the DeNN should be able to reproduce these general functions with appropriate neuron connections and parameters (weights, biases, and activation functions). Hence we can look at first an equivalent approximation as below without looking into detailed DeNN structures and coefficients. The corresponding DeNN architecture for C-HiDeNN will be presented later.

Let A^e denote the support node index set of the piecewise polynomial associated with element e and A_s^i be a local convolution patch node set for each supporting node i. The convolutional approximation can be written as

$$\begin{aligned} u^{\text{C-HiDeNN}}(\boldsymbol{\xi}) &= \sum_{i\in A^e} \mathcal{N}_i(\boldsymbol{\xi}) \sum_{j\in A_s^i} \mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi}) d_j \\ &= \sum_{k\in \Omega_{\text{patch}} := \bigcup_{i\in A^e} A_s^i} \tilde{\mathcal{N}}_k(\boldsymbol{\xi}) d_k \end{aligned} \tag{4}$$

where $\boldsymbol{\xi}$ denotes the general parametric coordinate, d_j is the nodal coefficient. $\mathcal{N}_i(\boldsymbol{\xi})$ is the polynomial function associated with the node i, $\mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi})$ is the local convolution patch function with a dilation parameter "a" that considers the contribution from the neighborhood (A_s^i) of the node i. s is a defined patch size that controls the size of the neighborhood. Consequently, the convolution is defined between the two functions $\mathcal{N}$ and $\mathcal{W}$ with respect to the index sets of i and j. The convolution patch function plays a similar role as the kernel/window function in convolutional neural networks, which is expected to improve the solution smoothness and extract selected features at desired length scales by controlling the parameter a. In our work, we define this parameter a as the dilation parameter that is frequently used in meshfree methods. The final convolution shape function $\tilde{\mathcal{N}}$ is then defined by the products of $\mathcal{N}$ and $\mathcal{W}$ with the convolution patch domain $\Omega_{\text{patch}} := \bigcup_{i\in A^e} A_s^i$. As an example, Fig. 2 illustrates the formation procedure of the convolution patch for a 4-node element.

The choice of functions $\mathcal{N}_i(\boldsymbol{\xi})$ and $\mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi})$ can be very flexible and affects the implementation complexity and numerical performance. In general, both finite element [47] and isogeometric [11] approximations can be used for $\mathcal{N}$, meshfree type approximations (e.g., [2, 23, 48, 49]) can be suitable options for the local convolution patch function $\mathcal{W}$. In order to simplify the construction and the implementation of boundary conditions for solving partial differential equations (PDEs). We designed these two functions to satisfy certain conditions, as shown in Table 1.

Based on these conditions, a natural choice for the general polynomial interpolation is the HiDeNN-FEM approximation (see Appendix A) or simply traditional FE shape functions (linear, quadratic, cubic, etc.) [47, 50]. For the convolution patch function, the radial basis interpolation function [51, 52], modified reproducing kernel function [27], and selectively interpolating moving least-squares approximation [53] can be used. It is obvious that if the polynomial

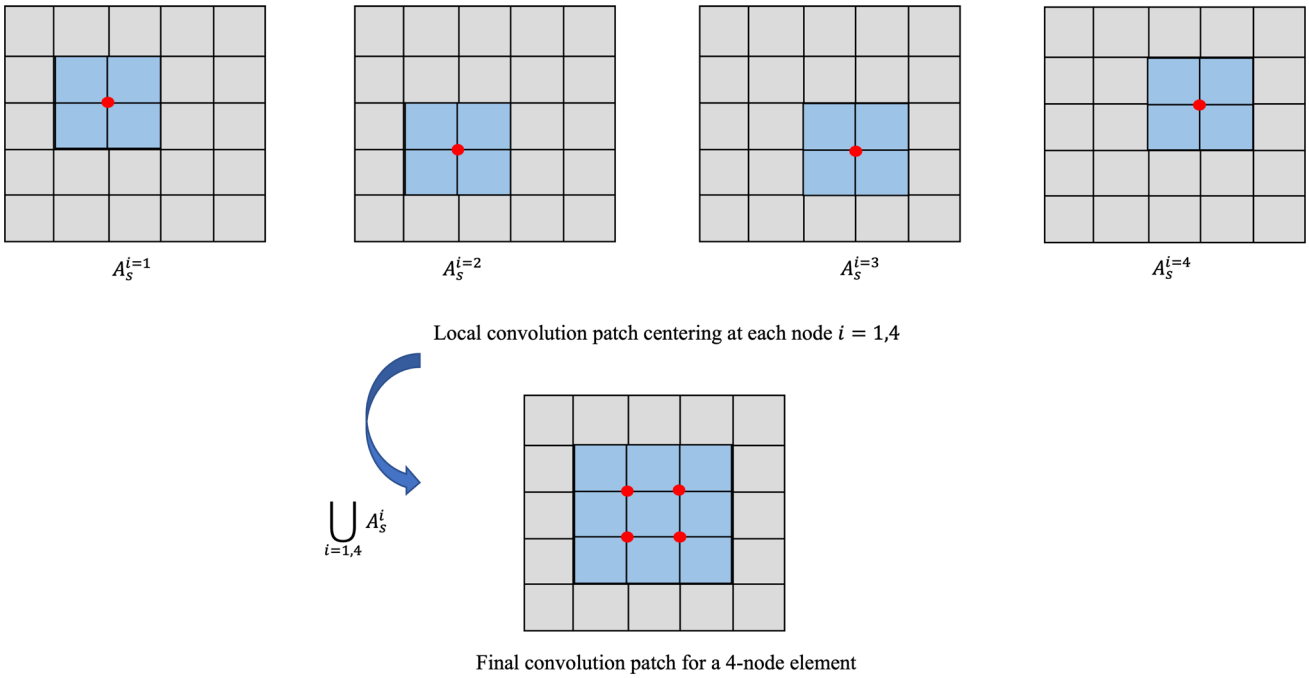


Fig. 2 Formation of the convolution patch for a 4-node element

Table 1 Conditions for constructing the convolution interpolation functions

General polynomial function $\mathcal{N}_i(\boldsymbol{\xi})$	Convolution patch function $\mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi})$
1. Compact support with $A^e \subset \bigcup_i A_s^i$	3. Kronecker delta: $\mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi}_k) = \delta_{ik}$
2. Partition of unity: $\sum_{i\in A^e} \mathcal{N}_i(\boldsymbol{\xi}) = 1, \forall \boldsymbol{\xi}$	4. Reproducing conditions: $\sum_{j\in A_s^i} \mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi}) P(\boldsymbol{\xi}_j) = P(\boldsymbol{\xi})$

where $P(\boldsymbol{\xi}_j)$ is a polynomial up to a certain order

function is chosen as constants, C-HiDeNN degenerates to meshfree approximations. On the contrary, if the dilation and patch become small enough, the C-HiDeNN reproduces FEM (at least in the sense of approximation accuracy). This implies the flexibility of the proposed C-HiDeNN framework and allows to easily degenerate the approximation to FEM, meshfree approximations, or a mixture of the two methods such as RKEM [27] or the so-called Generalized FEM (GFEM) [53]. Considering the universal approximation and the automatic adaptivity, the C-HiDeNN provides a unified framework to adaptively select appropriate approximations in the computational domain and is expected to keep the advantages of each method.

With these conditions, the Kronecker delta property is automatically satisfied, the nodal coefficient d_j in Equation (4) becomes the nodal solution u_j, as $u(\boldsymbol{\xi}_j) = u_j$. The resultant general convolutional interpolation reads

$$u^{\text{C-HiDeNN}}(\boldsymbol{\xi}) = \sum_{i\in A^e} \mathcal{N}_i(\boldsymbol{\xi}) \sum_{j\in A_s^i} \mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi}) u_j = \sum_{k\in \Omega_{\text{patch}}:=\bigcup_{i\in A^e} A_s^i} \tilde{\mathcal{N}}_k(\boldsymbol{\xi}) u_k \tag{5}$$

Compared to traditional meshfree methods, this approximation presents several advantages: (1) the connectivity of the nodes for each element remains unchanged during the computations, which avoids repetitive updating; (2) the enforcement of boundary conditions can be performed as usual finite element methods without difficulties; (3) the supporting domain of each convolution function is limited to a controllable finite domain by adjusting the dilation parameter and the local patch size, therefore remains compact.

It can be noticed that the current interpolation (5) is written in terms of a parametric system. This is for the convenience of numerical implementation like FEM. This is different from traditional meshfree methods as they are commonly built upon physical coordinates. From a general point of view, we can also consider the following convolution interpolation fully written in physical coordinates.

$$u^{\text{C-HiDeNN}}(\boldsymbol{x}) = \sum_{i\in A^e} \mathcal{N}_i(\boldsymbol{x}) \sum_{j\in A_s^i} \mathcal{W}_{a,j}^{\boldsymbol{x}_i}(\boldsymbol{x}) u_j \tag{6}$$

where the patch function $\mathcal{W}_{a,j}^{\boldsymbol{x}_i}(\boldsymbol{x})$ is completely defined in physical (real) space like traditional meshfree methods. This

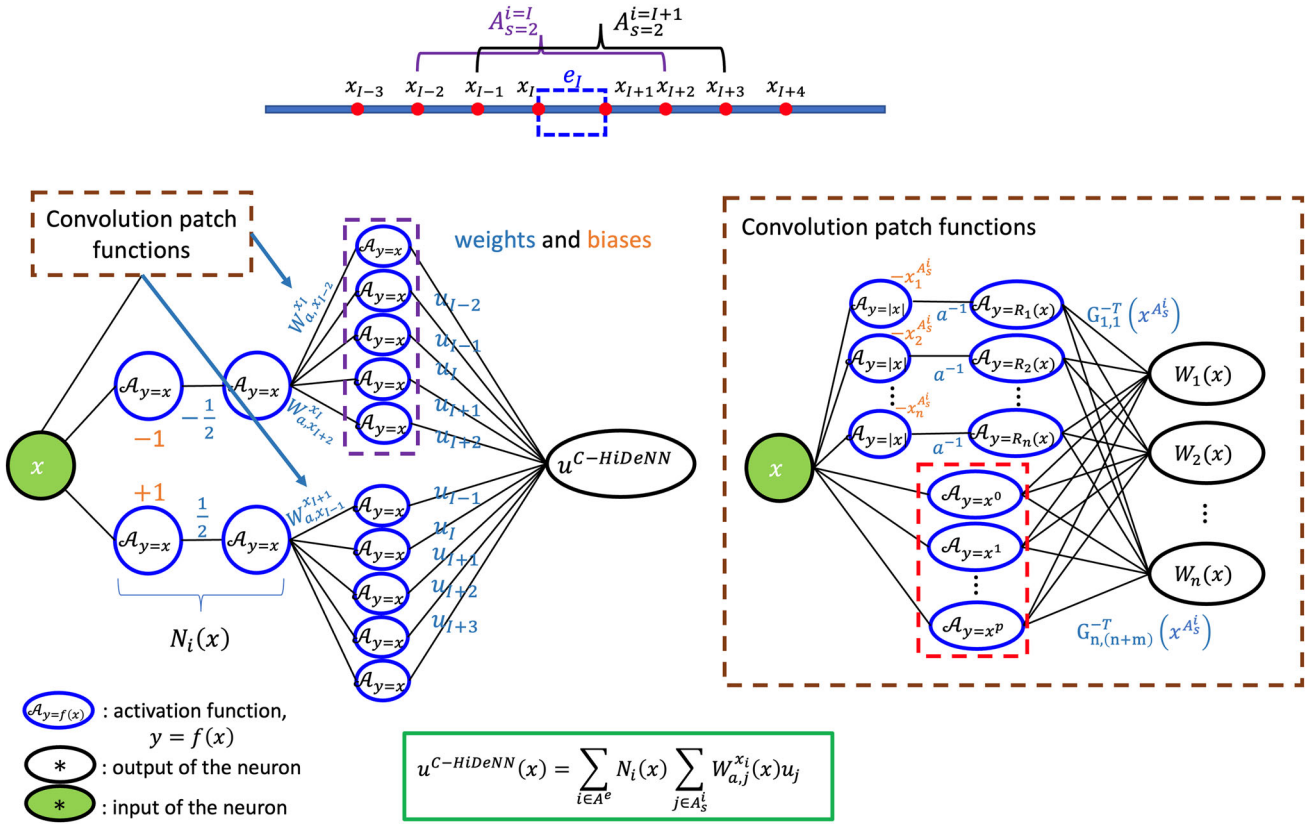


Fig. 3 Deep learning network structure of C-HiDeNN for a 2-node element with patch size $s = 2$. Detailed parameters of the network can be found in a companion paper [45]

formulation requires to update the patch function from element to element according to the real positions of supporting nodes.

With the previously defined C-HiDeNN formulation (5) or (6), we can now look at the neural network implementation of C-HiDeNN. The architecture of the resulting partially connected DeNN is illustrated in Fig. 3. This DeNN structure is defined in an analytical way, which means the weights, biases, and activation functions are prescribed as fixed so as to reproduce the desired C-HiDeNN approximation. No training for weights and biases are required. Hence, the DeNN coefficients (weights, biases, and activation functions) are omitted in the final C-HiDeNN approximation. Nevertheless, the designed structure still has the flexibility to relax the constraints on weights and biases and the neuron connections for a more general approximation, but the cost to pay is the training computational time. The approximation capability of neural networks of given fixed size can pose some questions and warrants more mathematical investigations in this case. An interesting work along this line can be found in [54].Another important aspect in this case is that training the parameters of neural networks can be highly non-convex and consequently very challenging [55–57]. This point is not the focus of current paper and will be investigated more in our future work.

Using the C-HiDeNN approximation, we can solve partial differential equations. As an example, assuming the potential energy Π exits for the problem, the C-HiDeNN solution can be obtained by solving following equation

$$u^{\text{C-HiDeNN}} = \underset{u_j^* \in \mathbb{R}, x_I^* \in \Omega \setminus \partial\Omega, a^* \in \mathbb{R}^+}{\arg\min} \Pi(u^{\text{C-HiDeNN}}(\boldsymbol{x})) \qquad (7)$$

where Ω denotes the entire domain, $\partial\Omega$ denotes the boundary. Here we can see that the nodal solution, coordinates, and the dilation parameter can be optimized together for highly accurate solutions. If mesh coordinates and dilation parameter do not change in the solution procedure, the C-HiDeNN is equivalent to an enhanced version of FEM with convolution approximation. Furthermore, it is possible to use different dilation parameters in different elements, leading to the so-called "a-adpativity". This will be discussed later in the paper.

Implementing the C-HiDeNN using neural networks is the focus of a companion paper in the same special issue [45], in which the graphics processing unit (GPU) is leveraged and discussed in detail. The focus of this paper is to present the basic concept and formulation with demonstrations on the high accuracy and various potential applications. Therefore, we will restrict thereafter our discussions to the

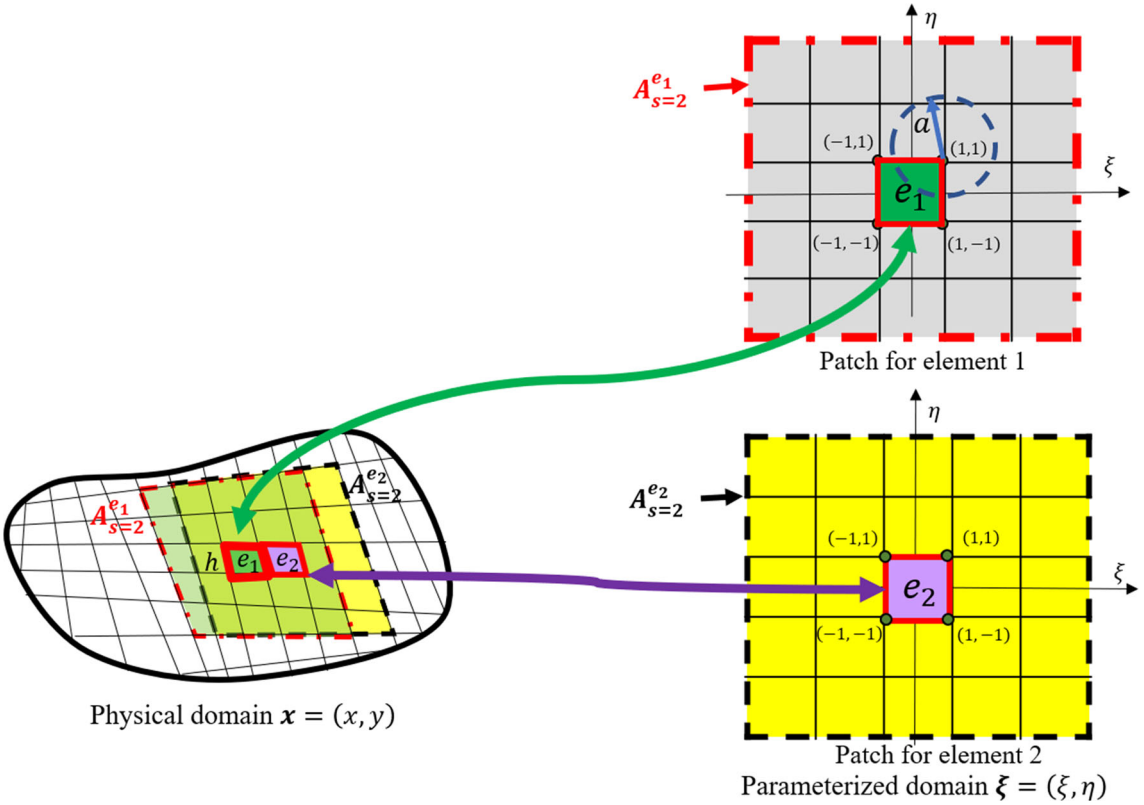


Fig. 4 Illustration of the element-wise convolution path domain with $s = 2$ and an arbitrary dilation parameter a, where $A_s^{e=1} := A_s^{e_1}$, $A_s^{e=2} := A_s^{e_2}$. The C-FEM does not increase the DoFs but uses the neighboring nodes of an element to achieve high-order smoothness

equivalent convolution finite element implementation, which is presented in the following.

3.2 A special case: Convolution FEM

We present here a special case of C-HiDeNN by restricting the $\mathcal{N}_i(\boldsymbol{\xi})$ and $\mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi})$, respectively, to a FE shape function and a radial basis interpolation function of a given order. Since the implementation can be done in a similar way to traditional FEM, this is referred to as Convolution FEM (C-FEM).

For clarification, let $N_i(\boldsymbol{\xi})$ denote the FE shape function, $W_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi})$ be the radial basis interpolation function. It is clear that the conditions in Table 1 are automatically satisfied with the C-FEM. The detailed derivation of the $W_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi})$ is given in Appendix B, in which we can define three parameters: dilation parameter, patch size, and polynomial order. The final displacement approximation can be written as below, with a newly defined shape function $\tilde{N}_k$, the so-called convolution FE shape function.

$$u^{\text{C-FEM}}(\boldsymbol{\xi}) = \sum_{i \in A^e} N_i(\boldsymbol{\xi}) \sum_{j \in A_s^i} W_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi}) u_j = \sum_{k \in A_s} \tilde{N}_k(\boldsymbol{\xi}) u_k, \tag{8}$$

where $A_s = \bigcup_{i \in A^e} A_s^i$ is the total supporting node set for the convolution shape function. Again, this supporting domain is larger than the original element region and includes the neighboring nodes that form the convolution patch A_s^i. Unlike specifying directly a higher-order FE shape function, this convolution shape function does not induce more DoFs in the mesh. Hence, it can increase the smoothness of the approximation without increasing significantly the overall computational cost. As an illustrative example, the supporting domain for a 2D convolution element with patch size 2 is shown in Fig. 4. For a better understanding of the convolution shape function $\tilde{N}_k$, we illustrate a specific 1D convolution approximation in Appendix C.

Similarly to FEM, we can define the isoparametric mapping between the natural coordinate and the physical coordinate.

$$\begin{aligned} x^{\text{C-FEM}}(\boldsymbol{\xi}) &= \sum_{i \in A^e} N_i(\boldsymbol{\xi}) \sum_{j \in A_s^i} W_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi}) x_j \\ &= \sum_{k \in A_s} \tilde{N}_k(\boldsymbol{\xi}) x_k, \end{aligned} \tag{9}$$

Hence we can derive the derivatives and perform the numerical integration in the same way as FEM. It should

be noticed that the integration points are still located inside each element (e.g., $[-1, 1] \times [-1, 1]$ for a 2D convolution element), although the patch expands the support to the neighborhood. The dilation a, patch size s, and the polynomial order p of the patch function will affect the span and smoothness of the final convolution shape function and dictate the convergence rates of the C-FEM.

Figure 5 illustrates some examples of the 1D convolution shape functions with different dilation parameters and polynomial orders of the patch function W for a patch size $s = 2$. The detailed derivation of these functions can be found in Appendix C. It can be seen that the final convolution shape function has 6 components and satisfies the Kronecker delta property and partition of unity in the element region. Increasing the dilation a leads to a larger influencing radius of each supporting node. Higher-order polynomials will make the final functions smoother, leading to a more regular shape. It is expected that there exists an optimal combination of these parameters, which provides the optimal convergence rate and accuracy for a given problem. This point will be studied numerically in this work using the numerical examples.

3.3 *a*-adaptivity

In addition to h-, p-, and r- adaptivity of traditional FEM, the proposed method enables the a-adaptivity. That is, the dilation parameter a can be varying from element to element in the the solution domain. As mentioned previously, the dilation parameter a is expected to control the length scale of approximation and its optimal value should depend on the solution behaviour. In heterogeneous materials or problems with complex stress distribution, the optimal characteristic length should be location-specific and defined by a spatially varying a. This unique feature of a-adaptivity makes C-HiDeNN approximation very suitable for various problems with different characteristic length scales without modifying the mesh.

The a-adaptivity can be done in an automatic fashion as r-adaptivity in HiDeNN method by optimizing the loss function (e.g., energy potential). The derivative of solution with respect to a can be derived without special difficulties. An alternative way to change a can be performed using conventional adaptation strategies by introducing appropriate error estimates. The implementation of a-adaptivity will be investigated more in our future work.

3.4 Remark for the treatment of boundary elements

The previous definition of the patch may encounter some issues for the elements on the boundary of the computational domain as there are no enough neighboring nodes for a given patch size. To address this point, we suggest ignoring the patch region that expands beyond the computational domain as shown in Fig. 6. A novel set of the patch function $\mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}$ can be calculated in the same way as previously but using the reduced support.

3.5 Discussion on the irregular mesh

As mentioned earlier, the convolution patch function $\mathcal{W}$ can be computed based on either a natural (parametric) coordinate system like traditional FEM or a physical coordinate system like traditional meshfree methods. If the natural coordinate system is adopted, additional adjustments are needed for irregular meshes for ensuring the optimal convergence rate of the method, according to our experience. By default, the natural coordinate system is a regular mesh with equally distancing nodes, which does not directly reflect the real spacing information from an irregular physical mesh. Computing the $\mathcal{W}$ based on such regular coordinates seems inappropriate for a physically irregular mesh. Hence, we propose modifying the nodal position of the patch in the parametric domain to reflect the relative distance between the nodes in physical space. Our experience showed that this modification helps achieve the expected convergence rates of the convolution FEM. Interested readers can be referred to Appendix D for a detailed description.

3.6 Discussion on implementation of the method

The computational framework can be implemented into an existing FE solver. Detailed implementation leveraging Google JAX for CPUs and GPUs based parallel computing can be found in a companion paper in the same special issue [45]. If we consider only a specific case of C-HiDeNN, i.e., the C-FEM, C-FEM shares many common development aspects with traditional FEM and the general partition of unity (POU)-FEM, such as using only linear finite element e.g., 3-node triangular (2D), or 4-node tetrahedral (3D) mesh generation, matrix assembly, and solution procedure. One major difference is in construction of the elementary stiffness matrix: the C-FEM considers some surrounding nodes outside each element, whereas traditional FEM does not; the C-FEM stiffness formulation shares similar expansion of the bandwidth of the stiffness matrix, to POU-FEM, though the superiority over POU-FEM or the generalized finite element method (GFEM) is that the "p-order" convergence and better accuracy can be obtained with the same size of the linear finite element mesh. In addition, the C-FEM can be easily degenerated to traditional FEM using a small dilation parameter (at least in the sense of approximation accuracy), which also simplifies the implementation of an adaptive adoption of different elements in different subdomains of the problem.

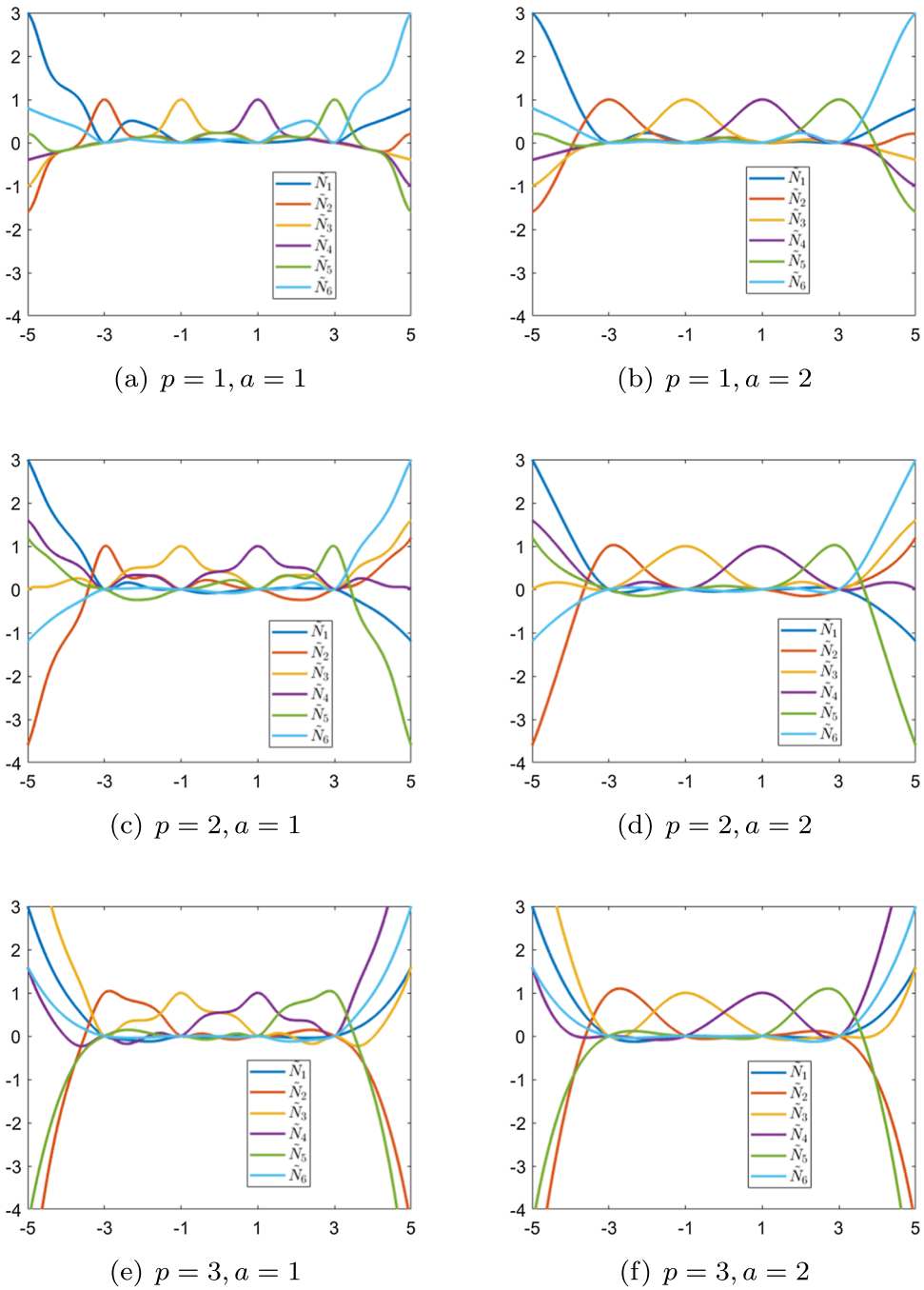


Fig. 5 Illustration of the convolution shape function, where the patch size $s = 2$, the support nodes locate at $\{-5, -3, -1, 1, 3, 5\}$, p denotes the polynomial order of the convolution patch function W. The original FE shape function remains linear for all the cases, whereas the convolution shape function is nonlinear with higher-order smoothness

3.7 Theoretical convergence and C^n continuity

3.7.1 Theoretical convergence rate

If taking $P(\boldsymbol{\xi})$ as p-th order polynomials in the reproducing conditions (see Table 1), C-HiDeNN interpolants (e.g., C-FEM) satisfy the reproducing property as well, i.e.,

$$\sum_{k \in A_s} \tilde{\mathcal{N}}_k(\xi)(\xi_k)^m = \xi^m, m = 0, 1, 2, \ldots, p. \tag{10}$$

Thus below theoretical results on the upper bound of the error of C-HiDeNN for elliptic equations can be expected, according to the previous work on the meshfree reproducing property [8, 27]

$$\|u^{\text{C-HiDeNN}}(\boldsymbol{x}) - u^{\text{Ext}}(\boldsymbol{x})\|_{H^1} \leq \tilde{C} h^p \|u^{\text{Ext}}\|_{H^{p+1}} \tag{11}$$

where $\tilde{C}$ is a constant, h the mesh size, p the reproduced polynomial order, and u^{Ext} denotes the exact solution. Detailed proof of the above equation can be found in Appendix E. It indicates that C-HiDeNN achieves the p-th order conver-

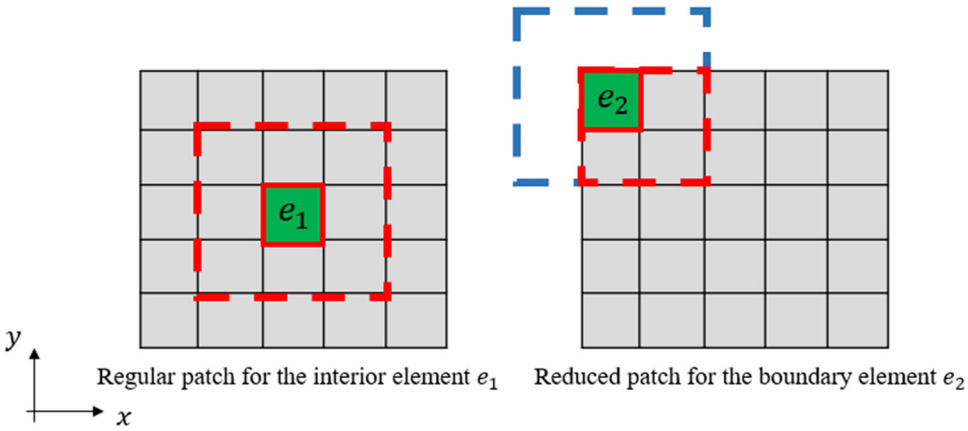


Fig. 6 Reduced patch for a convolution element on the boundary with patch size $s = 1$

gence rate when the interpolation functions satisfy the p-th order reproducing conditions, which is verified by our numerical examples.

3.7.2 Continuity

The C-HiDeNN interpolants can be rewritten in the form of shape functions with nodal values, i.e.,

$$\begin{aligned} u^{\text{C-HiDeNN}}(\boldsymbol{\xi}) &= \sum_{i\in A^e} \mathcal{N}_i(\boldsymbol{\xi}) \sum_{j\in A_s^i} \mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi}) u_j \\ &= \sum_{k\in A_s} \tilde{\mathcal{N}}_k(\boldsymbol{\xi}) u_k, \end{aligned} \tag{12}$$

where $\tilde{\mathcal{N}}_k(\boldsymbol{\xi})$ is the C-HiDeNN convolution shape function, defined by

$$\tilde{\mathcal{N}}_k(\boldsymbol{\xi}) = \sum_{i|k\in A_s^i} \mathcal{N}_i(\boldsymbol{\xi}) \mathcal{W}_{a,k}^{\boldsymbol{\xi}_i}. \tag{13}$$

According to the chain rule, the N-th order derivative of $\tilde{\mathcal{N}}_k(\boldsymbol{\xi})$ is

$$\begin{aligned} D^N \tilde{\mathcal{N}}_k(\boldsymbol{\xi}) = \sum_{i|k\in A_s^i} \big[& (D^N \mathcal{N}_i(\boldsymbol{\xi}) \mathcal{W}_{a,k}^{\boldsymbol{\xi}_i} \\ & + N (D^{N-1} \mathcal{N}_i(\boldsymbol{\xi}) (D \mathcal{W}_{a,k}^{\boldsymbol{\xi}_i}) \\ & + \cdots + \mathcal{N}_i(\boldsymbol{\xi}) (D^N \mathcal{W}_{a,k}^{\boldsymbol{\xi}_i}) \big], \end{aligned} \tag{14}$$

where D^N denotes the N-th order derivative, assuming $\mathcal{N}_i(\boldsymbol{\xi})$ is C^n continuous and $\mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}$ is C^m continuous. It clearly shows that the continuity of $\tilde{\mathcal{N}}_k(\boldsymbol{\xi})$ depends on the continuities of $\mathcal{N}_i(\boldsymbol{\xi})$ and $\mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}$. $D^N \tilde{\mathcal{N}}_k(\boldsymbol{\xi})$ is continuous up to $N = \min(n, m)$.

For 1D case, $\tilde{\mathcal{N}}_k(\boldsymbol{\xi})$ has C^N continuity with $N = \min(n+1, m)$, if the $D^{n+1}\mathcal{N}_i(\boldsymbol{\xi})$ is discontinuous across nodes, e.g., FEM shape functions. That is due to the Kronecker delta property of $\mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}$ and partition of unity property of $\mathcal{N}_i(\boldsymbol{\xi})$. The first term in (14) at the left and right limits of ξ_j is

$$\begin{aligned} & \sum_{i|k\in A_s^i} \left(\frac{d^N}{d\xi^N} \mathcal{N}_i(\xi_j^+) \right) \mathcal{W}_{a,k}^{\xi_i}(\xi_j^+) \\ & \quad = \sum_{i|k\in A_s^i} \left(\frac{d^N}{d\xi^N} \mathcal{N}_i(\xi_j^+) \right) \delta_{jk} \\ & \quad = \delta_{jk} \frac{d^N}{d\xi^N} \sum_{i|k\in A_s^i} \mathcal{N}_i(\xi_j^+) = 0, \end{aligned} \tag{15}$$

$$\begin{aligned} & \sum_{i|k\in A_s^i} \left(\frac{d^N}{d\xi^N} \mathcal{N}_i(\xi_j^-) \right) \mathcal{W}_{a,k}^{\xi_i}(\xi_j^-) \\ & \quad = \sum_{i|k\in A_s^i} \left(\frac{d^N}{d\xi^N} \mathcal{N}_i(\xi_j^-) \right) \delta_{jk} \\ & \quad = \delta_{jk} \frac{d^N}{d\xi^N} \sum_{i|k\in A_s^i} \mathcal{N}_i(\xi_j^-) = 0. \end{aligned} \tag{16}$$

The first term vanishes at nodes, so the continuity of (14) depends on the second term and the final term.

In general, we consider that $\mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}$ achieves higher order continuity than $\mathcal{N}_i(\boldsymbol{\xi})$, so the overall continuity is depending on the selection of $\mathcal{N}_i(\boldsymbol{\xi})$. We list the continuities of C-HiDeNN interpolants with FEM shape functions (C^0) and other shape functions with higher continuity in Table 2. Careful design of $\mathcal{N}_i(\boldsymbol{\xi})$ and $\mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}$ can lead to arbitrary high order C^N continuity of C-HiDeNN interpolants, which is difficult for traditional FEM. For example, $\mathcal{N}_i(\boldsymbol{\xi})$ can be chosen as B-spline basis functions.

Table 2 Continuities of C-HiDeNN interpolants with FEM shape functions (C^0) and other shape functions with high order of continuity. Assume that $\mathcal{W}_{a,j}^{\xi_i}$ is C^m continuous, $m \geq 2$

Continuities of C-HiDeNN interpolants	FEM shape functions $\mathcal{N}_i(\boldsymbol{\xi})$ (C^0)	$\mathcal{N}_i(\boldsymbol{\xi})$ (C^1)	$\mathcal{N}_i(\boldsymbol{\xi})$ (C^n)
1D	C^1	C^2	$C^{\min(n+1,m)}$
2D, 3D	C^0	C^1	$C^{\min(n,m)}$

4 C-HiDeNN-PGD/TD: reduced order methods

The convolution shape function can be directly applied to reduced-order methods (ROM) in place of the traditional FE shape function. We can expect that the resultant C-HiDeNN-ROM can be more accurate than the traditional ROM and even FEM, knowing that the traditional ROM can only be as accurate as FEM at best.

Reduced order methods (ROM) can be mainly categorized into two classes, (1) data-driven and non-intrusive methods, (2) intrusive projection-based ROM. The first class of methods is usually used as data compression or reduced order surrogate modeling with some precomputed simulation data or even experimental data. Examples of methods of this class are traditional tensor decomposition [58–61], High Order Proper Generalized Decomposition (HOPGD) [62–66]. The second class of methods requires an intrusive implementation and is usually used for accelerating physical simulations. Some of the well-known methods are the reduced basis method [67, 68], Proper Orthogonal Decomposition (POD) [69–71], the hyper reduction method [72, 73], and Proper Generalized Decomposition (PGD) [39–41, 74, 75]. A comprehensive discussion of the mathematical background and applications can be found in [58, 75, 76].

Regardless of the applicability of the convolution approximation to other ROM, we focus on the second class of methods, in particular, those using low rank approximations for solving partial differential equations (PDEs), the so-called HiDeNN-Tensor Decomposition (HiDeNN-TD) [39] and PGD methods [40, 42] both fall into this class. We will demonstrate that the convolution shape function can help improve the approximation capability of this type of method. To simplify terminology, we use TD and PGD to both refer to low-rank approximation based ROM of the second class that are used for solving PDEs. Their major difference is in the solution scheme. PGD uses an incremental strategy for solving different modes whereas TD requires additional iterations for optimizing all the modes together and reducing the total number of modes. Detailed discussions about these two methods can be found in [39].

By applying the C-HiDeNN shape function (5) into PGD or TD, the so-called Convolution HiDeNN-PGD/TD (C-HiDeNN-PGD/TD) approximation for a 3D problem is written as

$$\begin{aligned} u^{\text{C-HiDeNN-PGD/TD}}(\boldsymbol{\xi}) &= u^{\text{C-HiDeNN-PGD/TD}}(\xi,\eta,\gamma) \\ &= \sum_{m=1}^{M} \tilde{\boldsymbol{\mathcal{N}}}_\xi \boldsymbol{u}_\xi^{(m)} \tilde{\boldsymbol{\mathcal{N}}}_\eta \boldsymbol{u}_\eta^{(m)} \tilde{\boldsymbol{\mathcal{N}}}_\gamma \boldsymbol{u}_\gamma^{(m)}, \end{aligned} \tag{17}$$

where $\tilde{\boldsymbol{\mathcal{N}}}_\xi = [\tilde{\mathcal{N}}_1(\xi), \tilde{\mathcal{N}}_2(\xi), \ldots, \tilde{\mathcal{N}}_{n_\xi}(\xi)]$ is the 1D convolution shape function written in vector form, $\boldsymbol{u}_\xi^{(m)} = [u_1^{(m)}, u_2^{(m)}, \ldots, u_{n_\xi}^{(m)}]^T$ is the nodal solution vector for the m-th mode in the ξ-direction, assuming the total number of modes is M and is small enough compared to the system size. This definition is similar for $\tilde{\boldsymbol{\mathcal{N}}}_\eta, \boldsymbol{u}_\eta^{(m)}, \tilde{\boldsymbol{\mathcal{N}}}_\gamma, \boldsymbol{u}_\gamma^{(m)}$. It can be seen that the separation of variables allows adopting only 1D convolution shape function and solving 1D nodal solutions for a multidimensional problem. This advantage enables a high computational complexity reduction. In particular, the total number of degrees of freedom of the system is reduced from $n_x \times n_y \times n_z$ to $(n_x + n_y + n_z) \times M$, assuming n_x, n_y, n_z are respectively the number of nodes in the three directions (x, y, z) of a 3D field. This reduction leads to the so-called reduced order model.

Similarly to C-FEM, by fixing the weights and biases of the C-HiDeNN approximation and ignoring the neural network implementation, we can define the special case, the so-called C-TD/PGD, as

$$\begin{aligned} u^{\text{C-PGD/TD}}(\boldsymbol{\xi}) &= u^{\text{C-PGD/TD}}(\xi,\eta,\gamma) \\ &= \sum_{m=1}^{M} \tilde{\boldsymbol{N}}_\xi \boldsymbol{u}_\xi^{(m)} \tilde{\boldsymbol{N}}_\eta \boldsymbol{u}_\eta^{(m)} \tilde{\boldsymbol{N}}_\gamma \boldsymbol{u}_\gamma^{(m)}, \end{aligned} \tag{18}$$

This way, we can solve the problem in the same way as TD or PGD, the detailed solution procedure is presented in Appendix F. We expect to obtain a better solution with higher accuracy using the convolution approximation. If the number of modes is large enough, the final convolution ROM solution can be even more accurate than full-order FEM, which is usually impossible for traditional ROM (e.g., PGD [40] or TD [39, 44]).

For a better understanding of this point, let us look at a simple function space analysis shown in Fig. 7. For notation simplification, we use TD for both PGD and TD methods as their approximation space remains equivalent if we ignore the difference in the solution procedure. As demonstrated by our

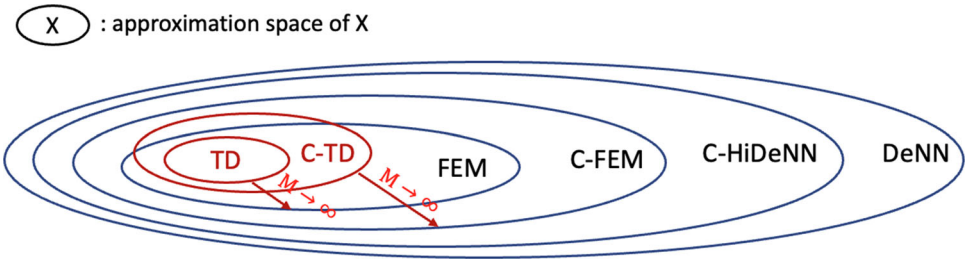


Fig. 7 Function approximation space for different approximation methods: tensor decomposition (TD), convolution TD (C-TD), finite element method (FEM), convolution FEM (C-FEM), convolution Hierarchical Deep-learning Neural Networks (C-HiDeNN), and Deep-learning Neural Networks (DeNN)

previous work [39, 44], the function space dictates the optimal solution accuracy of different approximation methods, i.e., larger approximation space usually leads more accurate solutions. Furthermore, the TD or PGD approximation space can converge to FEM if the number of modes tends to be infinite [39, 44]. The proof is based on the regular mesh and the assumption that the FEM shape function is the product of 1D shape functions used for PGD/TD along each direction. This property should remain similar for C-TD and C-FEM, i.e., C-TD converges to C-FEM with increasing number of modes. As the C-FEM has clearly a larger function space than FEM, the final C-TD approximation space with enough modes should be able to extend beyond the FEM space and therefore provide more accurate results than FEM. However, the original TD approximation does not have this ability according to this function space analysis.

Besides, the C-HiDeNN and DeNN should have larger function approximation space thanks to the universal approximation capability of the neural networks. They have the potential to provide the most accurate results, though high-performance solution schemes needed to be explored. We will explore more on this point in our future work.

For summary, we can expect the following result on the accuracy of different methods.

$$\begin{aligned}\|u^{\text{DeNN}} - u^{\text{Ext}}\|_E &\leq \|u^{\text{C-HiDeNN}} - u^{\text{Ext}}\|_E \\ &\leq \|u^{\text{C-FEM}} - u^{\text{Ext}}\|_E \\ &\leq \|u^{\text{C-PGD/TD}} - u^{\text{Ext}}\|_E \\ &\leq \|u^{\text{FEM}} - u^{\text{Ext}}\|_E \\ &\leq \|u^{\text{PGD/TD}} - u^{\text{Ext}}\|_E, \end{aligned} \tag{19}$$

where $\|\cdot\|_E$ denotes the energy norm, u^{Ext} denotes the exact solution. The detailed proof can be found in Appendix G.

5 Numerical results

In this section, we will study the numerical performance of the proposed convolution approximation methods, especially in terms of accuracy and convergence rates.

5.1 1D Poisson's problem and convergence studies

First, let us look at a 1D Poisson's problem with a distributed load b.

$$\begin{cases} \nabla \cdot (\nabla u(x)) + b(x) = 0, \quad \forall x \in \Omega := (-l, l) \\ u(-l) = 0, \\ u(l) = 0, \end{cases} \tag{20}$$

where $b(x) = -e^{-\frac{x^2}{c}}(-\frac{2x}{c})^2 - e^{-\frac{x^2}{c}}(-\frac{2}{c})$. In our work, $c = 0.01$ and $l = 0.6$ are adopted. The analytical solution to this problem is then

$$u^{\text{Ext}}(x) = e^{-\frac{x^2}{c}}, \tag{21}$$

For numerical solutions, we discretize the domain Ω using a regular mesh with uniformly distributed nodes. The linear two-node element is used for the reference FEM solution.

Regarding the C-FEM solution, we fix at this time the patch size $s = 3$ and dilation parameter $a = 3.72$ but change the polynomial order p of the patch function W from 1 to 4 to study their resultant solution behavior. We remark that changing the polynomial order does not affect the mesh size and the total number of degrees of freedom of the system remains unchanged. The linear FE shape function N in the equation (8) for C-FEM remains the linear FE shape function in this case.

We use 6 different meshes with $h = h_0, h_0/2, h_0/4, h_0/8, h_0/16, h_0/32$ and $h_0 = 0.05$ for the convergence study. The solution error is defined using the relative L^2-norm and energy norm as below

$$\epsilon_L = \frac{\|u - u^{\text{Ext}}\|_{L^2(\Omega)}}{\|u^{\text{Ext}}\|_{L^2(\Omega)}}, \tag{22}$$

$$\epsilon_E = \frac{\|u - u^{\text{Ext}}\|_E}{\|u^{\text{Ext}}\|_E} = \frac{\|u_{,x} - u^{\text{Ext}}_{,x}\|_{L^2(\Omega)}}{\|u^{\text{Ext}}_{,x}\|_{L^2(\Omega)}}, \tag{23}$$

where $u_{,x} = \frac{du}{dx}$.

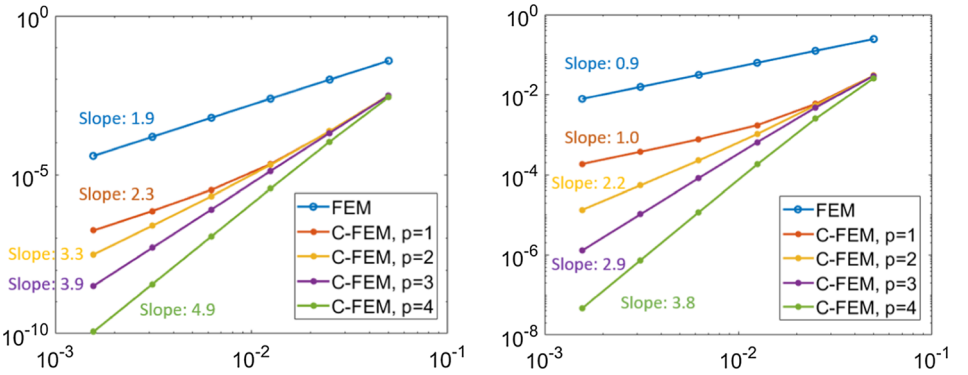


Fig. 8 Convergence study against the mesh refinement, x-axis: Log (h), y-axis: Log (ϵ)

(a) L^2-norm error (ϵ_L) in log-scale (b) Energy norm error (ϵ_E) in log-scale

As shown in Fig. 8, the C-FEM has higher order convergence rates than the original FEM. As expected, the two-node FE solution can only have a theoretical convergence rate of 2, whereas the C-FEM can achieve arbitrary convergence rates by simply adjusting the polynomial order without increasing the number of nodes. Under the L^2-norm and the energy norm measure, the expected convergence rates for C-FEM with a given polynomial order p are, respectively, $p+1$ and p. This is consistent to the theoretical analysis. Moreover, it can be seen that the C-FEM always provides better smooth solutions with an improved accuracy compared to the original FEM in all cases, even for $p = 1$. This confirms the performance of the proposed convolution approximation.

For better understanding the influence of different patch sizes and dilation parameters, we performed a parametric study for the ratio between the dilation a and patch size s. Figure 9 depicts the behavior of the L^2-norm error with the increasing dilation a for 6 different patch sizes. The overall behavior is that the error decreases with the increasing dilation before reaching a minimum value. The dilation a or the ratio a/s at this minimum may be considered the optimal parameter for the given patch size s. We can see that the optimal pair of a and s periodically appears when increasing the value of dilation. Nevertheless, we can see that the periodically appearing local minimum L^2 norm error remains almost identical when s is fixed. This means that we can take the smallest a that corresponds to the first local minimum as the best dilation parameter. If we plot the L^2-error convergence rates for different s with the best a, we can see a clear convergence slope of 2, as shown in Fig. 12a. Moreover, increasing the patch size s can improved the overall accuracy, although the convergence slope remains the same (see Fig. 12a). Interestingly, the final optimal a seems independent of mesh size h and only varies slightly when s increases.

Similar conclusions can be drawn from Figs. 10 and 11. By further looking at Fig. 12, we can observe that the optimal dilation remains around 3.8 in this example, regardless of the order p, mesh size h, and patch size s. This observation reflects that the optimal dilation might be related to some physical quantifies or characteristic length of a given problem. This point can be further explored in our future work.

We remark here that in this example the C-FEM approximation is constructed in the natural coordinate system and can be degenerated to linear FEM in the sense of approximation accuracy when $p = 1$ and $s = 1$ with a very small dilation, as shown in Fig. 12a.

In terms of computational expense, large s and a may result in a large bandwidth in the final global stiffness matrix. As the sparsity of the matrix can affect the efficiency of the matrix inversion, we suggest using the smallest s and a when satisfactory accuracy is obtained. In this example, $s = 4$ and $a = 3.8$ can be a good choice.

5.2 1D nonlinear heat conduction with phase transformation

In this study, we assess the stability of the C-FEM and FEM through the analysis of a nonlinear problem. Specifically, we consider a one-dimensional heat conduction problem in composite materials (see Fig. 13), where the two sides consist of two solid materials with distinct thermal capacities, densities, and thermal conductivity. The governing equation is

$$\frac{\partial}{\partial t}\left(\rho c_p T + f_L \Delta H\right) = -\nabla \cdot (k \nabla T) + Q \tag{24}$$

where ρ is the material density, c_p is the heat capacity, f_L is the liquid fraction with $f_L = 1$ for pure liquid state and $f_L = 0$ for pure solid sate, ΔH is the latent heat, k is the thermal conductivity, T is the temperature and Q represents the volumetric heat source. The material on the left-hand side has a density of 2200 g/m^3, a thermal capacity of 1700 J/K, and a thermal conductivity of 100 W/mK. The phase transformation is only considered for the right-hand side material, a phase transition from solid to liquid occurs at 660 K. In its

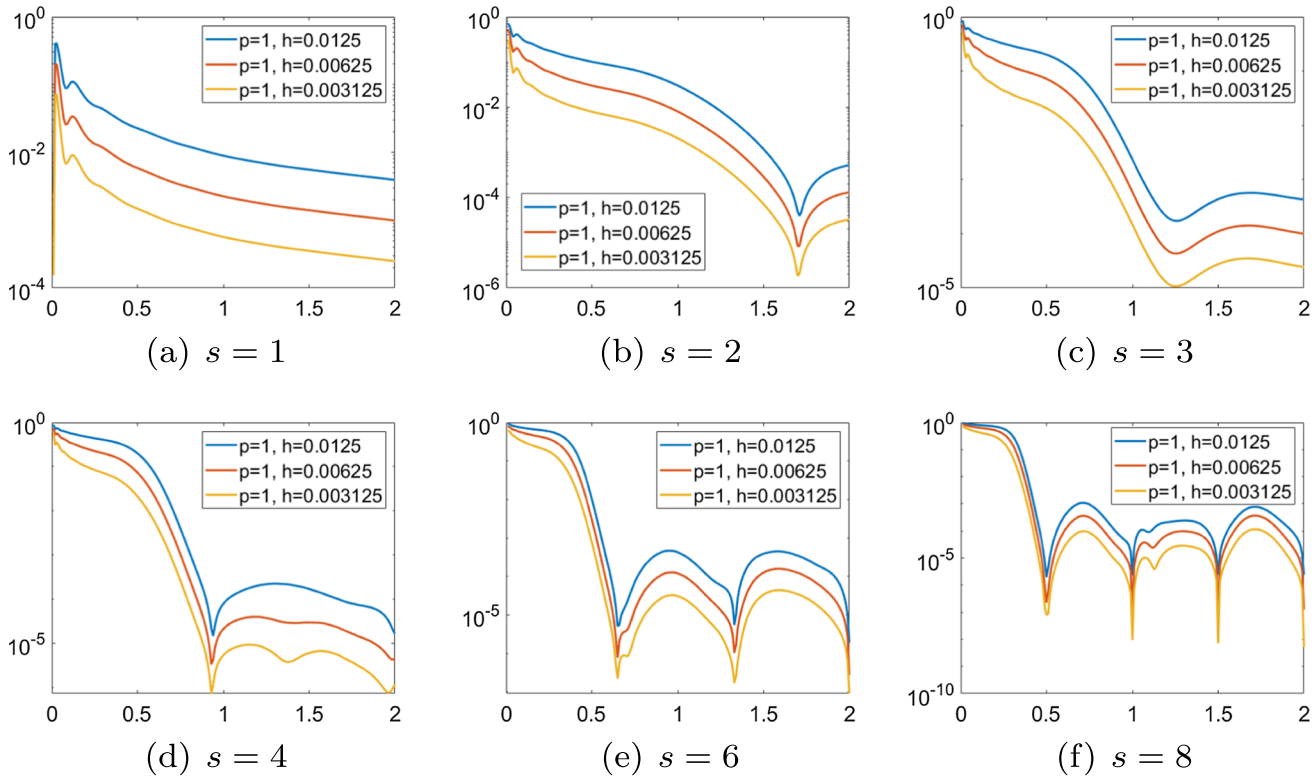


Fig. 9 Parametric study of the L^2-norm error on the different ratios of a/s for $p = 1$. x-axis: a/s, y-axis: L^2-norm error

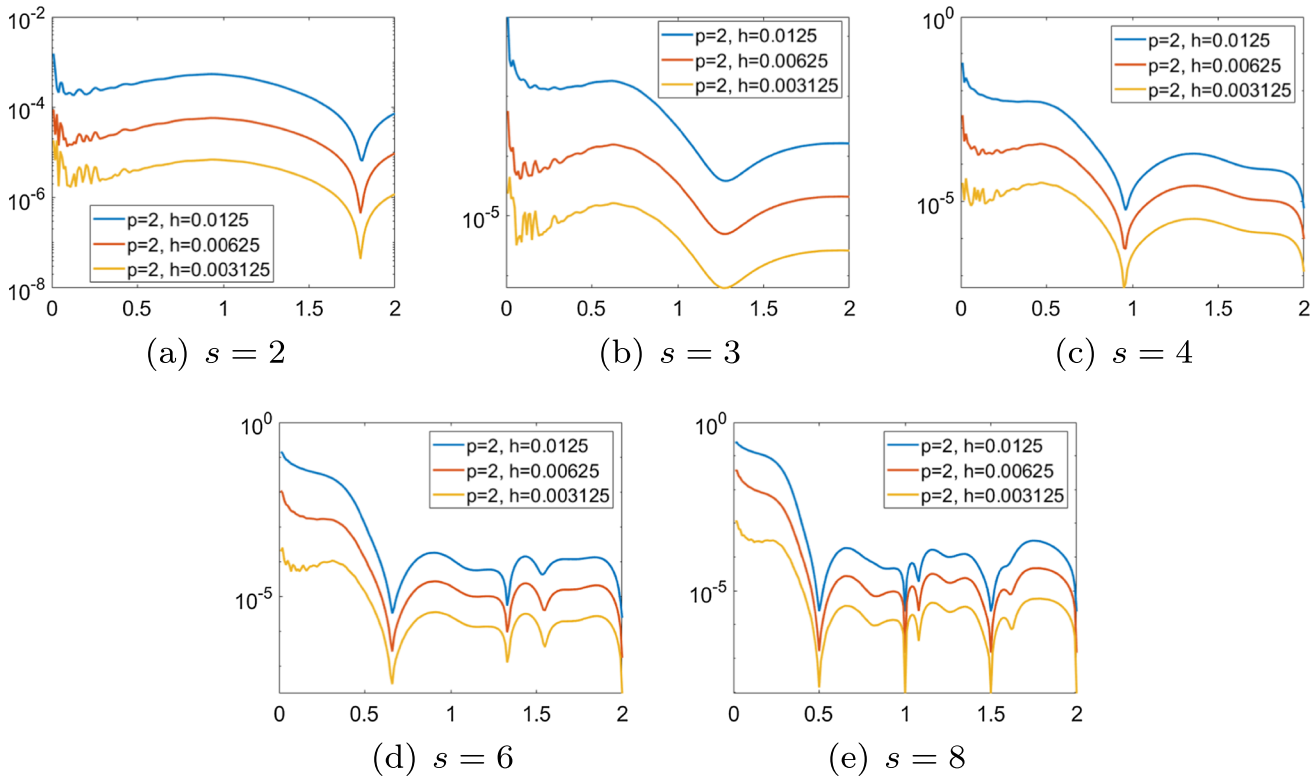


Fig. 10 Parametric study of the L^2-norm error on the different ratios of a/s for $p = 2$. x-axis: a/s, y-axis: L^2-norm error

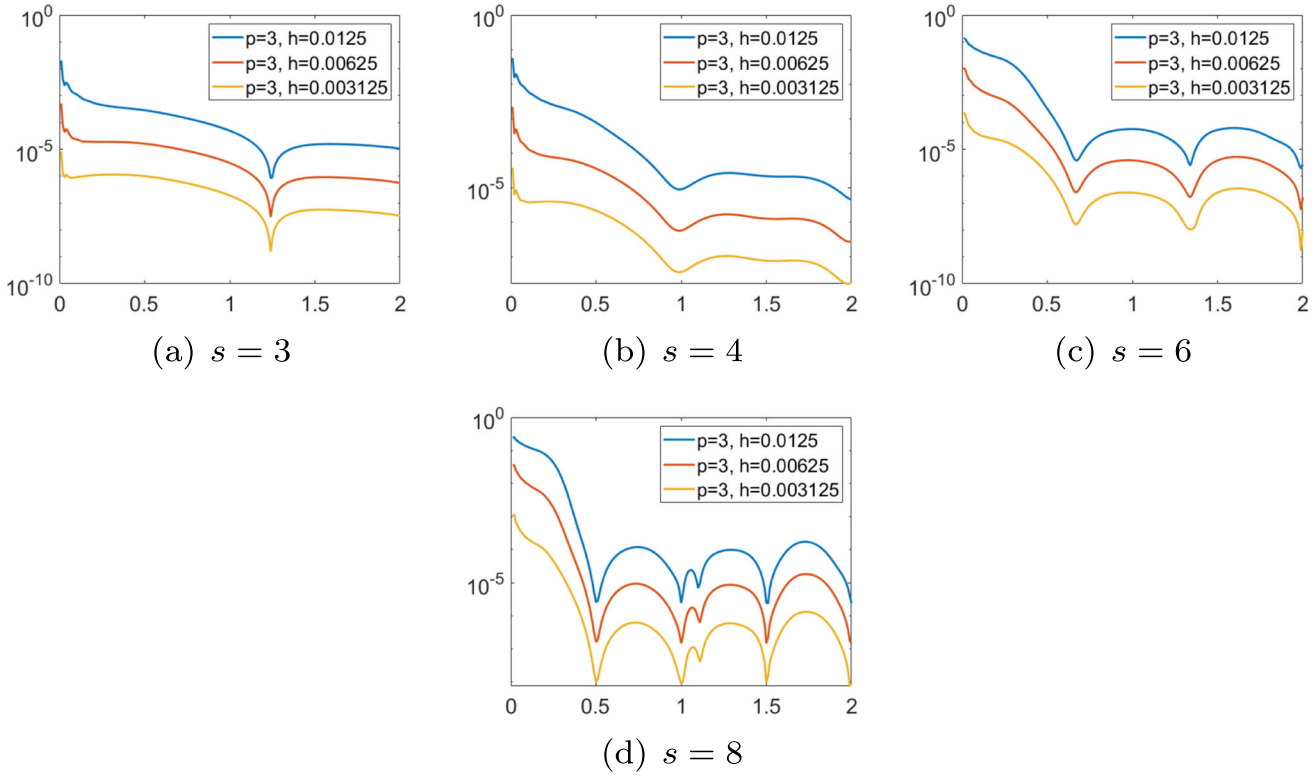


Fig. 11 Parametric study of the L^2-norm error on the different ratios of a/s for $p = 3$. x-axis: a/s, y-axis: L^2-norm error

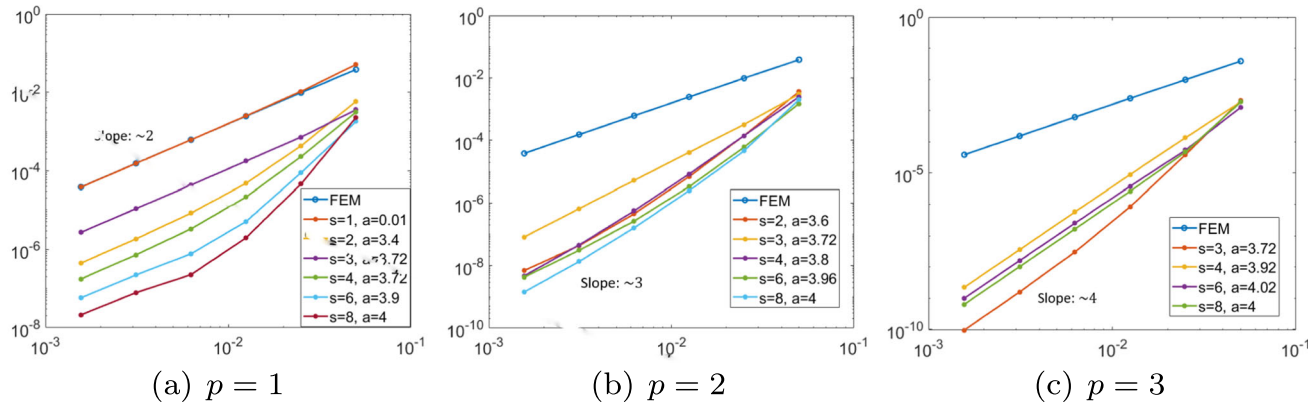
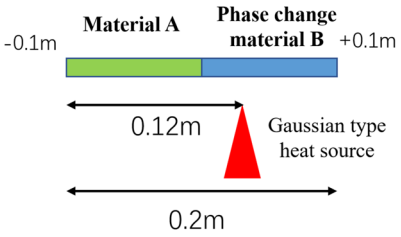


Fig. 12 The L^2-norm error convergence rates against the mesh refinement for different optimal combinations of a and s. x-axis: mesh size, y-axis: L^2-norm error

solid state, the material on the right-hand side has a density of 2555 g/m^3, a thermal capacity of 1190 J/K, and a thermal conductivity of 211 W/mK. Upon transition to its liquid state, the material maintains its original density and thermal capacity, but its thermal conductivity decreases to 91 W/mK. The latent heat for phase change is 3.98×10^6 J. Note that we do not consider the fluid flow in this example, the phase transformation is considered by material property variations and the produce of latent heat.

The initial temperature of the materials is 600 K, and a Gaussian heat source $Q = A \cdot \exp\left(-\frac{(x-0.02)^2}{r_a^2}\right)$, where $A = 1 \times 10^8$ and $r_a = 0.05$, is present at the position $x = 0.02$ m. The problem is discretized with an element size 5×10^{-4} m and the time step 0.25 s for both FEM and C-FEM. The reference solution is obtained by FEM using a much smaller mesh size 1×10^{-7} m as well as a shorter time step 1×10^{-3} s to ensure an accurate and stable solution.

This transient nonlinear problem is solved by the Newton-Raphson method with an implicit time integrator. The numerical solutions obtained by the C-FEM and FEM are depicted in Fig. 14. Here we studied the C-FEM with varying dilation parameters a. As shown in the figure, the traditional FEM



Material properties	Material A	Solid material B	Liquid material B
Density (g/m^3)	2200	2555	2555
Heat conductivity (J/K)	1700	1190	1190
Heat capacity (W/mK)	100	211	91
Phase change temperature (K)		660	
Latent heat (J)		3.98×10^6	

Fig. 13 Case setup for 1D nonlinear heat conduction with phase transformation

solutions (red curves) near the material phase transition point have some numerical instabilities due to the sudden change of heat. Similarly, when the dilation a used in the C-FEM method is very small, C-FEM solutions tend to have a similar unstable behavior to FEM solutions. This is because the C-FEM method with small a values closely resembles the traditional FEM method. However, when the dilation a used in the C-FEM method increases, the advantages of the C-FEM method begin to manifest and a stable solution consistent to the reference solution can be easily obtained without modifying the mesh, whereas FEM would need much finer meshes to achieve a stable solution.

The reason that the C-FEM method obtained stable numerical solutions in such nonlinear problems is the underlying regularization and numerical viscosity due to the dilation parameter. This is particularly desired in highly nonlinear problems to ensure the accuracy and physics-consistent solutions. This example demonstrates the significant advantages of the C-FEM method and shows its great potential to resolve the instability issues in nonlinear numerical analysis.

5.3 3D heat transfer problems for welding and additive manufacturing

We investigate two 3D heat transfer problems at this time for potential applications to problems with a concentrated heat source, like welding and additive manufacturing.

We focus here on the C-PGD/TD reduced order modeling method, since it allows to adopt directly the 1D convolution shape function for solving 3D problems and has great potential for extra-large scale numerical analysis. Since the TD can be seen as a general optimized PGD solution and they can achieve the same accuracy with enough modes as shown in [39], we do not distinguish C-TD and C-PGD here. We use C-TD as the unified name for this kind of method.

The first problem consists in studying the accuracy of the C-TD method with comparison to FEM. For this purpose, a heat conduction problem with manufactured load is designed as below

$$\begin{cases} \nabla \cdot \mathbf{q}(\mathbf{x}) = r(\mathbf{x}), \quad \forall \mathbf{x} \in \Omega \\ \mathbf{q} = -k \cdot \nabla T(\mathbf{x}), \\ T(\mathbf{x}) = 0, \quad \forall \mathbf{x} \in \partial\Omega_T, \end{cases} \tag{25}$$

where $\mathbf{q}$ is the heat flux, $T(\mathbf{x})$ is the temperature field, k is the material conductivity, Ω is the 3D cubic domain and considered large enough, $\partial\Omega_T$ is the bottom surface, $r(\mathbf{x})$ is the body heat source term with

$$r = -k\frac{4A(x^2+y^2+z^2)}{c^4}e^{-\frac{x^2+y^2+z^2}{c^2}} + k\frac{6A}{c^2}e^{-\frac{x^2+y^2+z^2}{c^2}} \tag{26}$$

The analytical solution to this problem is then

$$T^{\text{Ext}} = Ae^{-\frac{x^2+y^2+z^2}{c^2}} \tag{27}$$

In this work, we set $A = 1000$ and $c = 0.1$. The C-TD approximated solution is obtained by an alternating fix point algorithm as proposed by [40]. Figure 15 shows the final numerical solutions for FEM and C-TD. It is shown that they look very similar and can both provide accurate results if fine meshes are used. However, if we look at the convergence curve of both methods using the previously defined L^2-error (22), C-TD method does provide an overall better accuracy than FEM with respect to the exact solution, as shown in Fig. 16. If the original PGD/TD based ROM are used for this problem, they can only achieve the same accuracy as FEM at convergence. This confirms again the our theoretical analysis.

To further demonstrate the advantages of the C-PGD/TD reduced order model in terms of computational cost, we run a transient heat transfer problem with a moving heat source, as shown in Fig. 17. This kind of simulation is usually used to investigate the toolpath influence in additive manufacturing processes. Here, we used particularly C-PGD to investigate the efficiency. C-TD is expected to have a slight difference in the total computational time for the same accuracy. It can be

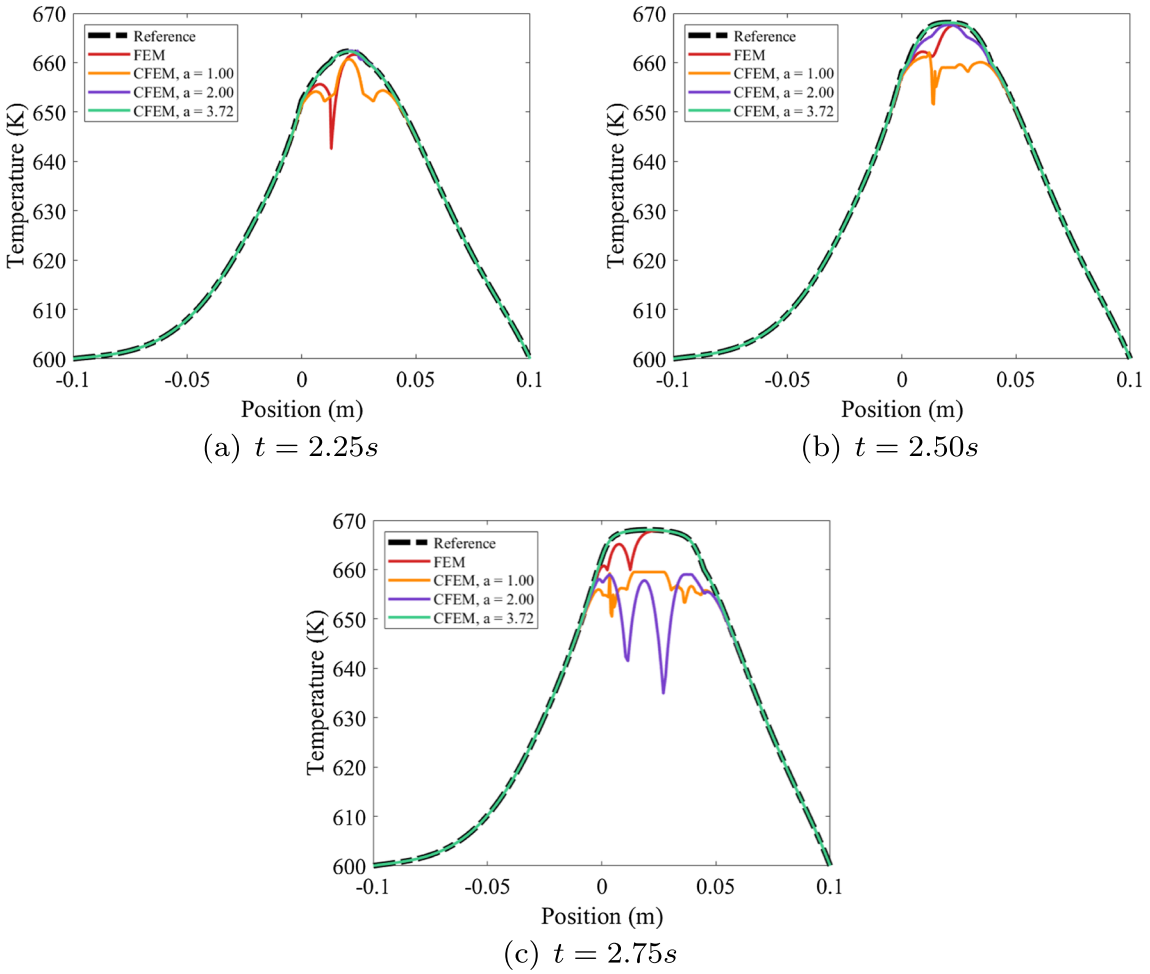


Fig. 14 The numerical simulation result comparison between the standard FEM and C-FEM with different dilation parameters. The patch size and polynomial order are both fixed at 3 in all C-FEM solutions.

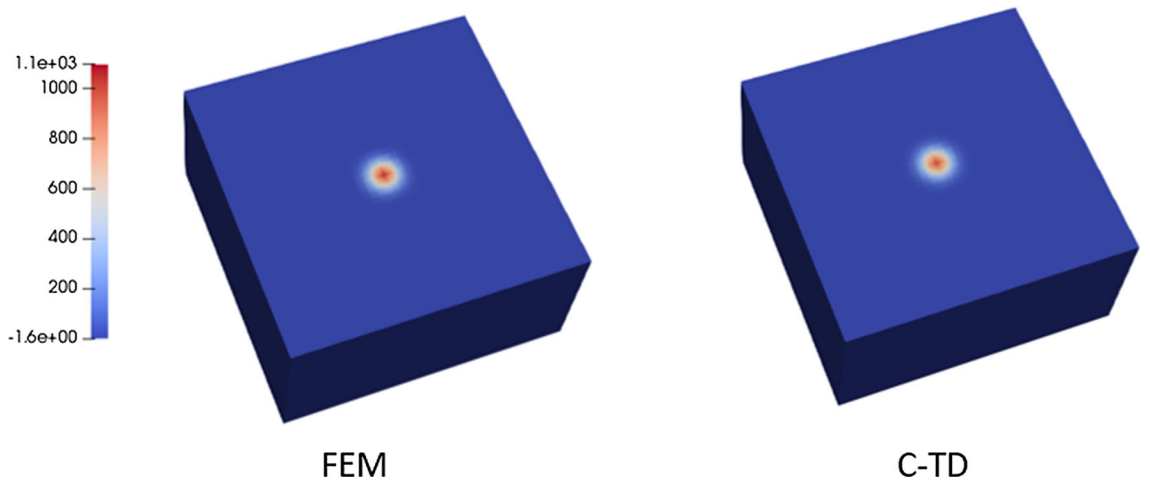


Fig. 15 Numerical solutions of FEM and C-TD for the temperature field

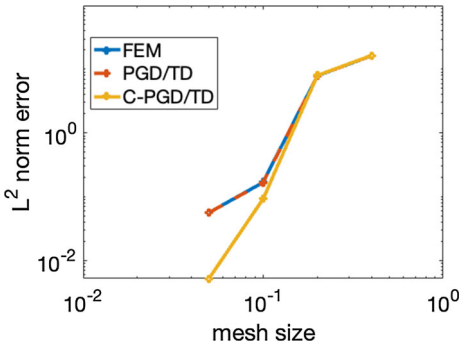


Fig. 16 L^2-norm error of the numerical solutions of FEM, PGD/TD, and C-PGD/TD against the mesh size

seen that the C-PGD provides visibly similar results to FEM. Since the analytical solution to this problem is not available, we do not focus on the accuracy comparison for this problem. However, in terms of computational cost, the C-PGD shows a significant speedup compared to FEM, especially when the mesh size is large. In our implementation, the total CPU time of FEM for the mesh size 1,774,094 (degrees of freedom) and 75 time steps is 545,118 s, whereas C-PGD takes about 73,896 s for the same computation, leading to a speedup of 7.37. Orders of magnitude speedups can be expected if larger meshes are used.

More importantly, as the C-PGD/TD can provide more accurate solutions than FEM, it has the possibility to use a coarser mesh for the same accuracy. Hence, the C-PGD/TD has a great potential to achieve high-performance computing with highly accurate solutions.

6 Future developments and applications

6.1 C-HiDeNN-Graphics Processing Unit (GPU) Computing

The convolution operations in C-HiDeNN, acting over the patch domains, are analogous to those in convolutional neural network (CNN). The definition of patch size s is similar to kernel size in CNN. The dilation parameter a affects kernel values and decide which kernel value is to be turned off. Therefore, advanced parallel programing tools originally developed for CNN or other machine learning techniques can also be applied to C-HiDeNN to reduce computation time. Figure 18 shows our preliminary results on graphics processing unit (GPU) computation of C-HiDeNN, revealing excellent speed-ups (approx. $\times 100$) compared to the same code running on CPU. For more details, readers may refer to [45].

6.2 C-HiDeNN for Isogeometric Analysis (IGA)

The flexibility and potential of the general framework of C-HiDeNN interpolants shown in Sect. 3.1 allows its extension to Isogeometric Analysis (IGA), called C-IGA. The key point of C-IGA is to design the local convolution patch functions $\mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi})$ in the parametric domain so as to reproduce the IGA basis functions such as B-spline and NURBS (nonuniform rational B-splines) basis functions, i.e.,

$$\sum_{j\in A_s^i} \mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi}) R(\boldsymbol{\xi}_j) = R(\boldsymbol{\xi}), \tag{28}$$

with $R(\boldsymbol{\xi})$ denoting IGA basis functions. This can be done by modifying the rhs (right-hand-side) of equation (48) with R, which defines the reproducing conditions of local convolution patch functions. Therefore, denoting the C-IGA interplation function as $\tilde{N}_k$, we have the following desired property

$$\sum_{k\in A_s} \tilde{N}_k(\boldsymbol{\xi}) R(\boldsymbol{\xi}_k) = R(\boldsymbol{\xi}). \tag{29}$$

C-IGA inherits the advantages of IGA: (1) exact geometry; (2) high order continuity; (3) compact support. Further more, C-IGA has the Kronecker delta property, and thus is convenient for the enforcement of boundary conditions.

Different from C-FEM, there exists three domains for computing: physical domain, parametric domain, and parent domain, as illustrated in Fig. 19. The mapping between physical domain and parametric domain is the C-IGA mapping:

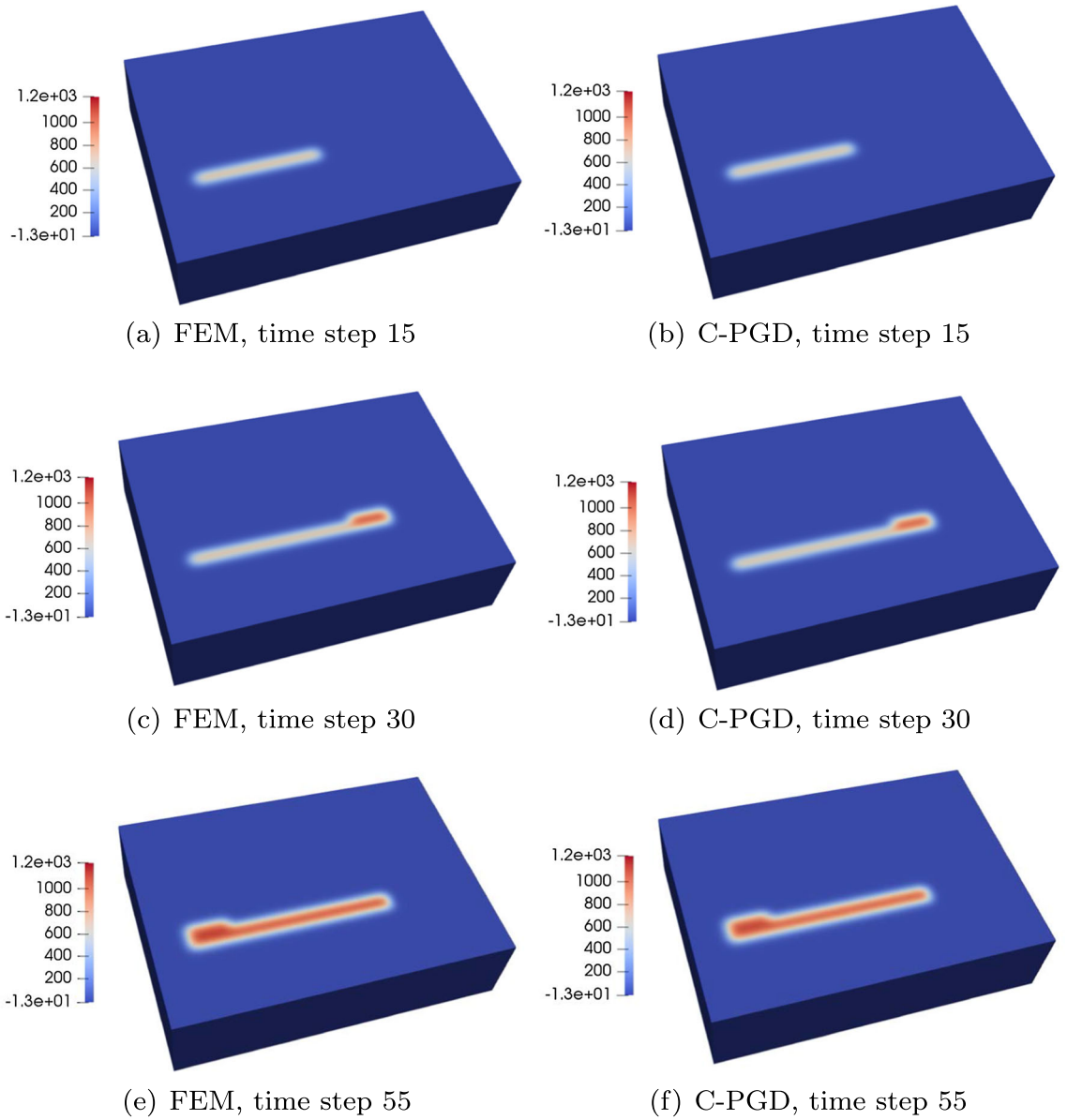
$$\boldsymbol{x}^{\text{C-IGA}}(\boldsymbol{\xi}) = \sum_{k\in A_s} \tilde{N}_k(\boldsymbol{\xi}) \boldsymbol{x}_k^{\text{IGA}} \tag{30}$$

where the coordinates $\boldsymbol{x}_k^{\text{IGA}}$ is obtained by the IGA mapping

$$\boldsymbol{x}_k^{\text{IGA}} = \boldsymbol{x}^{\text{IGA}}(\boldsymbol{\xi}_k) = \sum_i R_i(\boldsymbol{\xi}_k) \boldsymbol{B}_i \tag{31}$$

with $R_i(\boldsymbol{\xi})$ denoting IGA basis functions and $\boldsymbol{B}_i$ corresponding control points. Note that due to the reproducing property of C-IGA interpolants (29), the C-IGA mapping can reproduce precisely the IGA mapping and ensures the exact geometry representation like IGA

$$\begin{aligned}\boldsymbol{x}^{\text{C-IGA}}(\boldsymbol{\xi}) &= \sum_{k\in A_s} \tilde{N}_k(\boldsymbol{\xi}) x_k^{\text{IGA}} \\ &= \sum_{k\in A_s} \tilde{N}_k(\boldsymbol{\xi}) \sum_i R_i(\boldsymbol{\xi}_k) \boldsymbol{B}_i \\ &= \sum_i \boldsymbol{B}_i \sum_{k\in A_s} \tilde{N}_k(\boldsymbol{\xi}) R_i(\boldsymbol{\xi}_k)\end{aligned}$$



(a) FEM, time step 15 (b) C-PGD, time step 15

(c) FEM, time step 30 (d) C-PGD, time step 30

(e) FEM, time step 55 (f) C-PGD, time step 55

Fig. 17 Solution snapshots of FEM and C-PGD for the transient heat transfer problem

$$= \sum_i \boldsymbol{B}_i R_i(\boldsymbol{\xi}) = \boldsymbol{x}^{\mathrm{IGA}}(\boldsymbol{\xi}) \tag{32}$$

A numerical verification and comparison with FEM mapping is illustrated in Fig. 19. Here, we use the IGA produced geometry as a reference. With the same nodal position information, the C-IGA can reproduce exactly the reference geometry using the coordinate mapping, whereas linear FEM can only have straight lines between nodes.

Next, we can use the C-IGA interpolant to solve a Poisson's equation. As shown in Fig. 20, we consider 1D problem

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\mathrm{d}u}{\mathrm{d}x}\right) + b(x) = 0, x \in [0, 10] \tag{33}$$

with a body force

$$b(x) = -\frac{4\pi^2(x-2.5)^2 - 2\pi}{e^{\pi(x-2.5)^2}} - \frac{8\pi^2(x-7.5)^2 - 4\pi}{e^{\pi(x-7.5)^2}}. \tag{34}$$

To eliminate the boundary effect for IGA, we consider homogeneous boundary conditions, i.e.,

$$u(0) = u(10) = 0. \tag{35}$$

Note that due to the Kronecker delta property, C-IGA can easily handle the Dirichlet boundary conditions. The analyt-

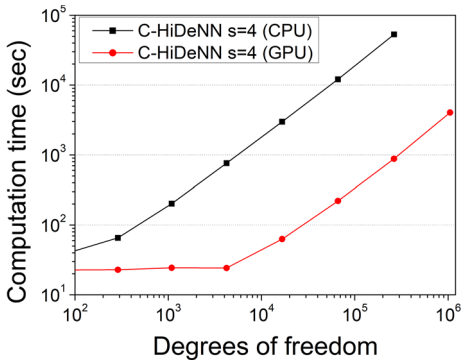


Fig. 18 Computation time for building C-HiDeNN shape functions for a 2D Poisson's equation when $s = 4$ with a large dilation. C-HiDeNN is coded with Python JAX library. Computing resources: CPU - Intel(R) Core(TM) i7-4790 CPU @ 3.60 GHz; GPU - NVIDIA A6000 48GB graphics memory

ical solution is

$$u(x) = (e^{-\pi(x-2.5)^2} - e^{-6.25\pi}) + 2(e^{-\pi(x-7.5)^2} - e^{-56.25\pi}) - \frac{e^{-6.25\pi} - e^{-56.25\pi}}{10}x. \tag{36}$$

As shown in Fig. 20, the preliminary results demonstrate that C-IGA achieves the same or higher convergence rate compared with conventional IGA. More studies will be reported in our future work.

6.3 C-HiDeNN for Topology Optimization (TO)

The other application for C-HiDeNN is the topology optimization (TO). TO aims to give the optimal structural design by relocating the materials in the design domain. The topology optimization equations with the Solid Isotropic Material with Penalization method (SIMP) approach can be written as

$$\begin{aligned} &min\ c(\rho) \\ &s.t.: \delta\Pi = \int_\Omega \nabla_s \boldsymbol{u}^T \boldsymbol{D}(\rho) \nabla_s \delta\boldsymbol{u} d\Omega - \int_\Omega \boldsymbol{f}^T \delta\boldsymbol{u} \rho d Omega \\ &\quad - \int_\Gamma \boldsymbol{t}^T \delta\boldsymbol{u} d\Gamma, \\ &\quad \rho_{min} \ll \rho \leq 1, 0 < \rho_{min} \ll 1, \\ &\quad \boldsymbol{g}(\rho) \leq 0, \end{aligned} \tag{37}$$

where $c(\rho)$ is the objective function with design variables ρ. The equilibrium equation is defined as $\delta\Pi$ in the principle of virtual work form. The design variable constraint is defined as $0 < \rho_{min} \ll \rho \leq 1$, where $\rho = \rho_{min}$ is no material and $\rho = 1$ is solid material. Other design constraints are defined in $\boldsymbol{g}$. To reduce the compuational cost in TO, we develop the C-HiDeNN-TD (Tensor Decomposition) method. After tensor decomposition and discretization, the principle of the virtual work can be converted to the following problems

For each mode m,

$$\begin{aligned} &\tilde{\boldsymbol{K}}_{u_x^{(m)}} \boldsymbol{u}_x^{(m)} = \boldsymbol{f}_{u_x^{(m)}},\ \tilde{\boldsymbol{K}}_{u_y^{(m)}} \boldsymbol{u}_y^{(m)} = \boldsymbol{f}_{u_y^{(m)}},\ \tilde{\boldsymbol{K}}_{u_z^{(m)}} \boldsymbol{u}_z^{(m)} = \boldsymbol{f}_{u_z^{(m)}}, \\ &\tilde{\boldsymbol{K}}_{v_x^{(m)}} \boldsymbol{v}_x^{(m)} = \boldsymbol{f}_{v_x^{(m)}},\ \tilde{\boldsymbol{K}}_{v_y^{(m)}} \boldsymbol{v}_y^{(m)} = \boldsymbol{f}_{v_y^{(m)}},\ \tilde{\boldsymbol{K}}_{v_z^{(m)}} \boldsymbol{v}_z^{(m)} = \boldsymbol{f}_{v_z^{(m)}}, \\ &\tilde{\boldsymbol{K}}_{w_x^{(m)}} \boldsymbol{w}_x^{(m)} = \boldsymbol{f}_{w_x^{(m)}},\ \tilde{\boldsymbol{K}}_{w_y^{(m)}} \boldsymbol{w}_y^{(m)} = \boldsymbol{f}_{w_y^{(m)}},\ \tilde{\boldsymbol{K}}_{w_z^{(m)}} \boldsymbol{w}_z^{(m)} = \boldsymbol{f}_{w_z^{(m)}}. \end{aligned} \tag{38}$$

where $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$ are the displacement components in the 3D problem, $(\cdot)_x, (\cdot)_y, (\cdot)_z$ are the decomposed 1D vectors in the x, y, z directions.. The stiffness matrix of each mode for each displacement is defined as $\tilde{K}$ with the integrated external force $\boldsymbol{f}$. Derivations of the above equations can be done in a similar way to Appendix F. By the decomposition, the 3D structural problem is decomposed into several 1D problems.

Figure 21 shows the cantilever beam design example with sample volume fraction and length scale control for different design resolutions. The chart shows that when increasing the number of DoFs, the computational cost increases from 10^4 to 10^6 h for traditional FEM-TO method. For the C-HiDeNN-TD-TO, it only increases from 33 to 112 h. The tensor decomposition makes the high resolution TO design much more affordable. Details of the C-HiDeNN-TD-TO and more studies can be found in [46].

7 Conclusion

A general C-HiDeNN framework has been developed for highly accurate and efficient solutions to scientific and engineering problems. The C-HiDeNN framework unifies the finite element and meshfree approximation methods and provides a flexible way to construct a desired approximation with high-order continuity. We demonstrated the capability of the proposed method using a special example of this method, the C-FEM. The proposed convolution method does not increase the overall degrees of freedom of the discretized system but provides higher accuracy and smoothness with controllable parameters. These parameters include the polynomial order of the convolution patch function, patch size, and dilation and can be easily modified to obtain arbitrary convergence rates and orders of magnitude higher accurate solutions without modifying the mesh.

Theoretically, we showed that the C-HiDeNN with the p-order reproducing condition can achieve the p-th order convergence rate in terms of H^1-norm. The C-HiDeNN and its variants, C-FEM and C-TD, can be more accurate than traditional FEM with the same mesh. The numerical analysis

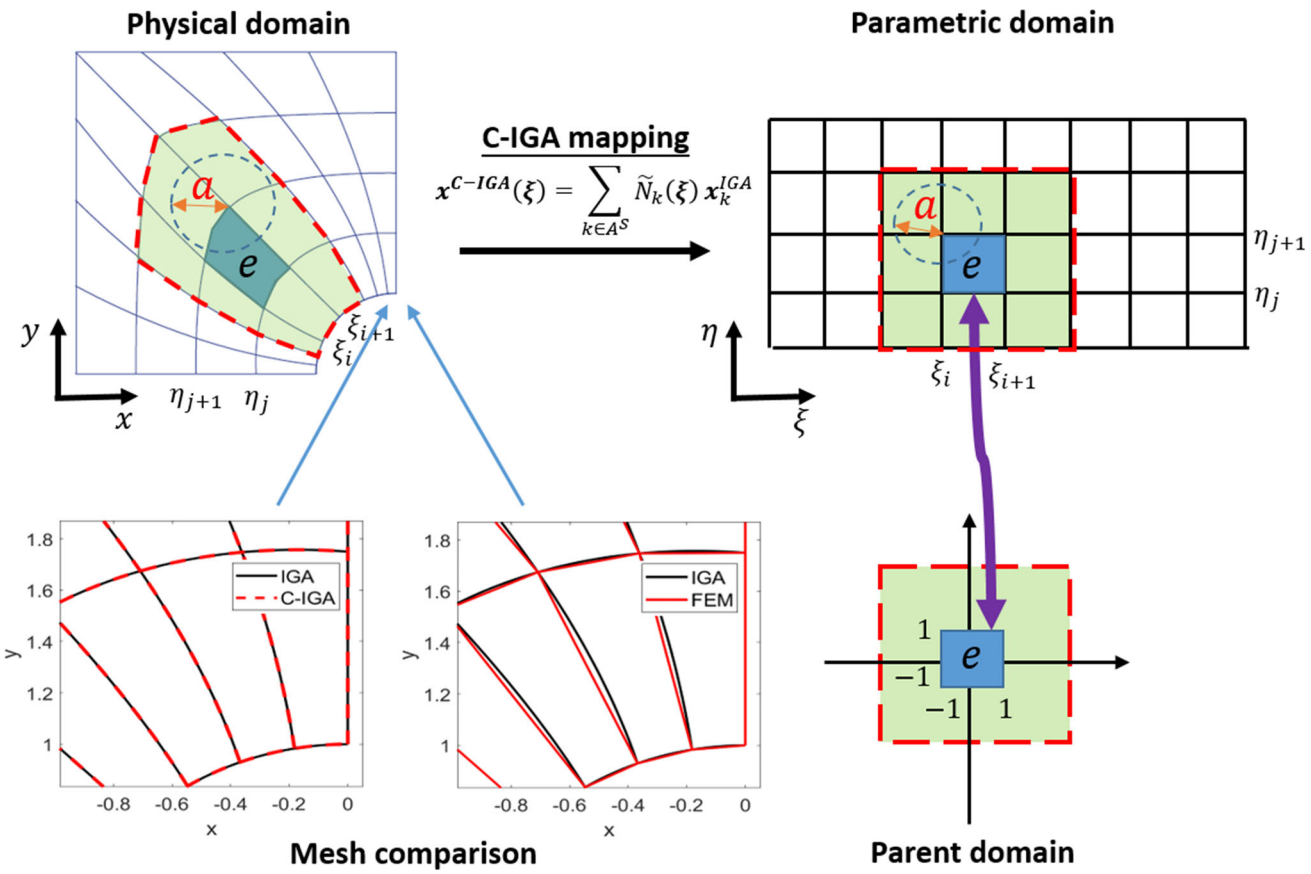


Fig. 19 Illustration of the geometrical mapping. The physical element is pulled back first to the parametric domain through the C-IGA mapping, then through a second mapping, one affine, to the parent element. Computation of radial basis functions $\mathcal{W}_{a,j}^{\xi_i}$ and Gaussian quadrature is performed over the parent domain. C-IGA mapping is equivalent to the IGA mapping, and is able to represent the exact geometry

has confirmed this result. The different optimal sets of the controlled parameters have been studied. We found that the optimal dilation might be related to some physical lengths and requires further investigations. Furthermore, it is found that the dilation parameter introduces some regularization and numerical viscosity into the solution procedure and can improve the solution stability for nonlinear analysis.

For future applications, the proposed C-HiDeNN framework has great potential for ultra-large-scale computing with high-fidelity solutions. This is essential to enable the so-called high-resolution topology design, which has been presented in our work. In addition, the C-HiDeNN and its variants, like C-TD, can be beneficial to many other computationally intensive problems like additive manufacturing process modeling, microstructural grain growth simulations, multi-scale analysis of materials, etc.

Acknowledgements S. Tang and L. Zhang would like to thank the support of the National Natural Science Foundation of China (NSFC) under contract Nos. 11832001, 11988102, and 12202451.

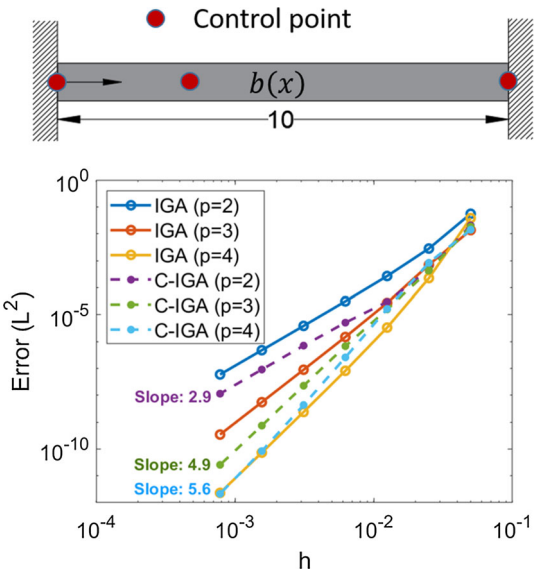


Fig. 20 1D numerical model for convergence study of C-IGA. IGA results are plotted by solid line, and C-IGA results are represented by dashed line. x-axis: mesh size h, y-axis: L^2-norm error

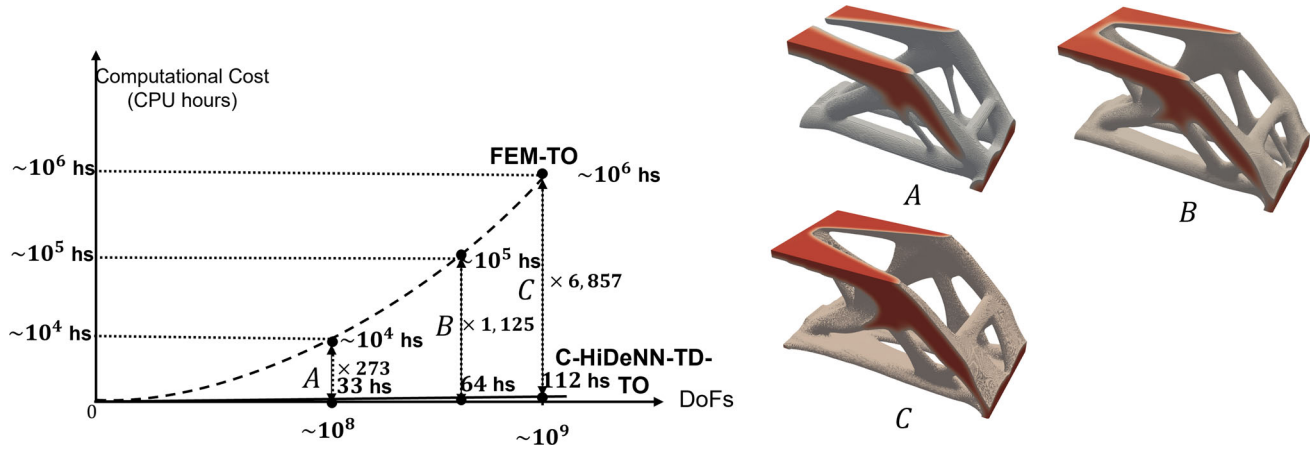


Fig. 21 DoFs vs Computational cost between FEM-TO and C-HiDeNN-TO for the cantilever beam design example. Higher resolution design (from A to C) leads to finer structures and smoothness

Appendix A Brief review of HiDeNN-FEM

The idea of HiDeNN-FEM is to use Deep-learning Neural Networks (DeNN) (We added the "e" instead of just using DNN because we prefer the acronym HiDeNN) to reconstruct the FE shape function by constraining the weights and biases with mesh coordinates, which reads

$$\mathcal{N}_I(\boldsymbol{x};\boldsymbol{x}_I^*,\boldsymbol{\mathcal{A}}) := \mathcal{F}_I(\boldsymbol{x};\boldsymbol{w}(\boldsymbol{x}_I^*),\boldsymbol{b}(\boldsymbol{x}_I^*),\boldsymbol{\mathcal{A}}) \tag{39}$$

where $\mathcal{F}_I$ stands for the fully connected DeNN structures with weights $\boldsymbol{w}$, biases $\boldsymbol{b}$, and the activation function $\boldsymbol{\mathcal{A}}$. $\mathcal{N}_I$ denotes the FE shape function for the node at position $\boldsymbol{x}_I^*$. Assuming a domain Ω is discretized by n points, we can write the HiDeNN-FEM approximation as

$$u^h(\boldsymbol{x}) = \sum_{I=1}^{n} \mathcal{N}_I(\boldsymbol{x};\boldsymbol{x}_I^*,\boldsymbol{\mathcal{A}}) u_I \tag{40}$$

where u_I is the discretized nodal solution of the problem, u^h is the approximated solution function. Considering the vector notation $\boldsymbol{\mathcal{N}} = [\mathcal{N}_1,\ldots,\mathcal{N}_n]$ and $\boldsymbol{u} = [u_1,\ldots,u_n]^T$, the equation (40) can be simplified as

$$u^h(\boldsymbol{x}) = \boldsymbol{\mathcal{N}}\left(\boldsymbol{x};\boldsymbol{x}^*,\boldsymbol{\mathcal{A}}\right)\boldsymbol{u} \tag{41}$$

The detailed construction of such shape functions using DeNN can be found in [38]. Figure 22 illustrates a detailed architecture of the partially connected DeNN, as an example of HiDeNN-FEM. It should be noticed that the FE shape function is only one of the choices, the HiDeNN structure allows one to easily switch from one to another by releasing the constraints on the weights $\boldsymbol{w}$ and biases $\boldsymbol{b}$.

Since the HiDeNN shape function serves the same role as FE shape function, the derivatives and integration of the shape function can be implemented in exactly the same way as FEM. Finally, the solution of HiDeNN-FEM can be obtained through an optimization problem in which both the displacement field and the mesh coordinates are simultaneously optimized. The problem reads

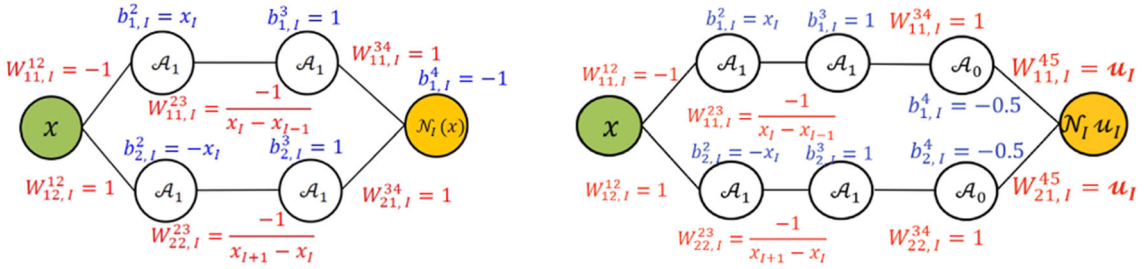
$$u^{\text{HiDeNN-FEM}} = \underset{u_I,\boldsymbol{x}_I^*\in\Omega\setminus\partial\Omega}{\arg\min} \; \Pi(u^h(\boldsymbol{x})) \tag{42}$$

where Ω denotes the entire domain, $\partial\Omega$ denotes the boundary, Π denotes the potential (when it exists) or the residual of the problem. Hence HiDeNN-FEM provides a new way to adaptively modify the mesh and reduces to FEM when the mesh is fixed. Moreover, it is shown that the HiDeNN-FEM has a function approximation space and gives more accurate results than FEM due to the flexibility of the HiDeNN framework [39].

Below, we summarize the key features of the HiDeNN-FEM method [38]:

- Partially connected DeNN with constrained weights and biases
- A flexible framework for function approximation and solving PDEs with automatic r-h-p adaptivity
- More accurate than traditional FEM
- Reduces to FEM when freezing the mesh coordinates

The developed novel Convolution HiDeNN method can be seen as an enhancement for highly smooth solutions and improved convergence rates without increasing the degrees of freedom (DoFs).



(a) DNN-based 1D shape function **(b)** DNN-based 1D interpolation function

Fig. 22 HiDeNN-FEM shape functions [38]. Weights and biases are predefined as constants. ReLU (rectified linear unit) activation function is used in the HiDeNN architecture

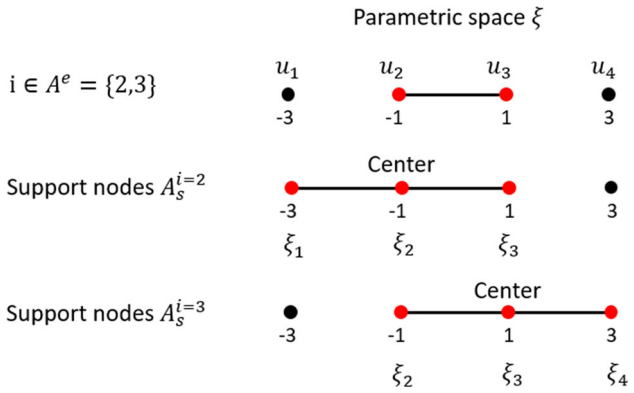


Fig. 23 Supporting nodes for a 1D convolution element with patch size $s = 1$

Appendix B Derivation of radial basis interpolation function

Let us consider the following 1D case for illustration purposes.

$$u^{\text{C-FEM}}(\xi) = \sum_{i \in A^e} N_i(\xi) \sum_{j \in A^i_s} W^{\xi_i}_{a,j}(\xi) u_j \tag{43}$$

Then we can define the following interpolation as part of the approximation centering around the i-th node in the element domain.

$$u^i(\xi) = \sum_{j \in A^i_s} W^{\xi_i}_{a,j}(\xi) u_j, \tag{44}$$

where the supporting node set of W is A^i_s with a given patch size s. Figure 23 illustrates a 1D convolution element with patch size $s = 1$ and the two-node shape functions for N_i. Now the question is how to compute the W based on the given supporting nodes.

Assuming the nodal solution value for the 4 nodes in Fig. 23 is $[u_1, u_2, u_3, u_4]$, we illustrate the radial basis interpolation procedure for the part centering around $i = 2$. In this case, the parametric coordinates for the support nodes are $\{-3, -1, 1\}$. Then we can consider the radial basis interpolation $u^{i=2}(\xi)$ has the following form

$$u^i(\xi) = \mathbf{\Psi}_a(\xi)\mathbf{k} + \mathbf{p}(\xi)\mathbf{l}, \tag{45}$$

where $\mathbf{\Psi}_a$ is a defined kernel function, which can be the reproducing kernel or cubic spline kernel [3, 4] with the dilation parameter a, $\mathbf{p}(\xi)$ is the polynomial basis vector of p-th order, $\mathbf{k} = [k_1, k_2, k_3]^T$ and $\mathbf{l} = [l_1, l_2, l_3]^T$ are the coefficient vector that helps to enforce the reproducing condition and Kronecker delta property. We give here a specific example for $\mathbf{\Psi}_a$ and $\mathbf{p}(\xi)$ using a cubic spline kernel and a second-order polynomial.

$$\begin{aligned} &\mathbf{\Psi}_a(\xi) = [\Psi_a(\xi - \xi_1),\ \Psi_a(\xi - \xi_2),\ \Psi_a(\xi - \xi_3)] \\ &\text{where}\quad \Psi_a(\xi - \xi_I) := \Psi_a(z) \quad \text{with} \quad z = \tfrac{|\xi - \xi_I|}{a} \\ &= \begin{cases} \frac{2}{3} - 4z^2 + 4z^3 & \forall z \in [0, \frac{1}{2}] \\ \frac{4}{3} - 4z + 4z^2 - \frac{4}{3}z^3 & \forall z \in [\frac{1}{2}, 1] \\ 0 & \forall z \in (1, +\infty) \end{cases}, \end{aligned} \tag{46}$$

and

$$\mathbf{p} = [1,\ \xi,\ \xi^2] \tag{47}$$

Now we can compute $\mathbf{k}$ and $\mathbf{l}$ by enforcing the below conditions.

$$\begin{cases} u^i(\xi_1) = u_1 \\ u^i(\xi_2) = u_2 \\ u^i(\xi_3) = u_3 \\ \sum \mathbf{k} = 0 \\ [\xi_1, \xi_2, \xi_3]\,\mathbf{k} = 0 \\ [\xi_1^2, \xi_2^2, \xi_3^2]\,\mathbf{k} = 0 \end{cases}, \tag{48}$$

Solving the above equations gives the solution to $\mathbf{k}$ and $\mathbf{l}$, which reads

$$\begin{cases} \mathbf{k} = \mathbf{K}\mathbf{u} \\ \mathbf{l} = \mathbf{L}\mathbf{u} \end{cases}, \tag{49}$$

with

$$\begin{cases} \mathbf{u}=[u_1,\ u_2,\ u_3]^T \\ \mathbf{L}=(\mathbf{P}^T\mathbf{R}_0\mathbf{P})^{-1}\mathbf{P}^T\mathbf{R}_0^{-1} \\ \mathbf{K}=\mathbf{R}_0^{-1}(\mathbf{I}-\mathbf{P}\mathbf{L}) \end{cases}, \tag{50}$$

and

$$\begin{cases} \mathbf{R}_0=\begin{pmatrix}\boldsymbol{\Psi}_a(\xi_1)\\ \boldsymbol{\Psi}_a(\xi_2)\\ \boldsymbol{\Psi}_a(\xi_3)\end{pmatrix}=\begin{pmatrix}\Psi_a(\xi_1-\xi_1) & \Psi_a(\xi_1-\xi_2) & \Psi_a(\xi_1-\xi_3)\\ \Psi_a(\xi_2-\xi_1) & \Psi_a(\xi_2-\xi_2) & \Psi_a(\xi_2-\xi_3)\\ \Psi_a(\xi_3-\xi_1) & \Psi_a(\xi_3-\xi_2) & \Psi_a(\xi_3-\xi_3)\end{pmatrix} \\ \mathbf{P}=\begin{pmatrix}\mathbf{p}(\xi_1)\\ \mathbf{p}(\xi_2)\\ \mathbf{p}(\xi_3)\end{pmatrix}=\begin{pmatrix}1 & \xi_1 & \xi_1^2\\ 1 & \xi_2 & \xi_2^2\\ 1 & \xi_3 & \xi_3^2\end{pmatrix} \end{cases}, \tag{51}$$

Finally, the radial basis interpolation with the computed coefficients reads

$$\begin{aligned} u^i(\xi)=\boldsymbol{\Psi}_a(\xi)\mathbf{k}+\mathbf{p}(\xi)\mathbf{l} &= \boldsymbol{\Psi}_a(\xi)\mathbf{K}\mathbf{u}+\mathbf{p}(\xi)\mathbf{L}\mathbf{u} \\ &= (\boldsymbol{\Psi}_a(\xi)\mathbf{K}+\mathbf{p}(\xi)\mathbf{L})\mathbf{u} \\ &= W_{a,1}^{\xi_i}(\xi)u_1+W_{a,2}^{\xi_i}(\xi)u_2 \\ &\quad +W_{a,3}^{\xi_i}(\xi)u_3 \\ &= \sum_{j\in A_s^i} W_{a,j}^{\xi_i}(\xi)u_j, \end{aligned} \tag{52}$$

where $W_{a,j}^{\xi_i}$ is obtained by identifying the corresponding coefficient of u_j. By analogy, we can compute the other convolution patch functions W with the support $A_s^{i=3}$. Detailed mathematical derivation and analysis of the radial basis interpolation can be found in [77].

Appendix C Illustration of a 1D convolution element

For a better understanding of the convolution shape function $\tilde{N}_k$, we illustrate here a 1D convolution approximation with N_i chosen to be linear and patch size $s=1$, see again Fig. 23 for the supporting nodes of one convolution element.

$$u^{\text{C-FEM}}(\xi)=\sum_{i\in A^e} N_i(\xi)\sum_{j\in A_s^i} W_{a,j}^{\xi_i}(\xi)u_j \tag{53}$$

In this case, the FE shape function nodal support set $A^e=\{2,3\}$, and then $A_s^{i=2}=\{1,2,3\}$, $A_s^{i=3}=\{2,3,4\}$. The equation (53) becomes

$$\begin{aligned} u^{\text{C-FEM}}(\xi) &= \sum_{i\in A^e} N_i(\xi)\sum_{j\in A_s^i} W_{a,j}^{\xi_i}(\xi)u_j \\ &= N_2(\xi)W_{a,1}^{\xi_2}(\xi)u_1+(N_2(\xi)W_{a,2}^{\xi_2}(\xi) \\ &\quad +N_3(\xi)W_{a,2}^{\xi_3}(\xi))u_2 \\ &\quad +(N_2(\xi)W_{a,3}^{\xi_2}(\xi) \\ &\quad +N_3(\xi)W_{a,3}^{\xi_3}(\xi))u_3+N_3(\xi)W_{a,4}^{\xi_3}(\xi)u_4 \\ &= \sum_{k\in A_s}\tilde{N}_k(\xi)u_k, \end{aligned} \tag{54}$$

where $A_s=\bigcup_{i\in A^e}A_s^i=\{1,2,3,4\}$. Therefore, there are in total 4 convolution shape functions for $s=1$. If $s=2$, we can expect 6 shape functions, as shown in Fig. 5.

Appendix D Modification of the natural coordinates for irregular meshes

The modification of the natural (parametric) coordinates of the patch nodes can be done according to the distance ratio in the physical domain. As shown in Fig. 24, if the mesh is irregular in the physical space, the corresponding patch nodes in parametric space are adapted. The neighbouring nodes x_1 and x_4 are clearly far from the element in the physical space. In order to reflect this distancing information in parametric domain. The coordinates ξ_1 and ξ_4 are modified as below

$$\xi_1=\xi_2-|\xi_2-\xi_3|\frac{|x_1-x_2|}{|x_2-x_3|},\quad \xi_4=\xi_3+|\xi_2-\xi_3|\frac{|x_3-x_4|}{|x_2-x_3|}, \tag{55}$$

that is

$$\frac{|\xi_1-\xi_2|}{|\xi_2-\xi_3|}=\frac{|x_1-x_2|}{|x_2-x_3|},\quad \frac{|\xi_3-\xi_4|}{|\xi_2-\xi_3|}=\frac{|x_3-x_4|}{|x_2-x_3|}, \tag{56}$$

where the right-hand-side of the above equation is the physical distance ratio and left-hand-side is the one in parametric space. By enforcing this condition, we are able to keep the nonuniform distribution of the nodes in parametric space. Here, the original physical element domain is $[x_2, x_3]$ and the corresponding parametric element domain $[\xi_2, \xi_3]$ remains the same as before, i.e., $[-1, 1]$.

From our experience this modification is necessary and can help achieve the optimal convergence rates when irregular meshes are used. Similar concept can be used in 2D/3D irregular meshes.

Appendix E Error estimate of C-HiDeNN interpolation

Following the procedure in the previous work on the meshfree reproducing property [8, 27], we can derive the error

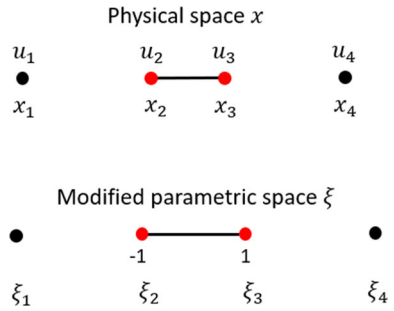


Fig. 24 Modified parametric coordinates for a 1D convolution element with patch size $s = 1$

estimate of C-HiDeNN interpolation, based on the reproducing property.

Theorem 1 *Let us assume an exact solution $u(x) \in C^{p+1}(\bar{\Omega}) \cap H^{p+1}(\Omega)$, where Ω is a bounded open set in $\mathbb{R}^n$. Define an interpolation operator $\mathcal{I}$*

$$\mathcal{I}u(x) = \sum_I \tilde{N}_I(x) u(x_I), \tag{57}$$

where the interpolation operator satisfies the reproducing property

$$\mathcal{I}x^m = \sum_I \tilde{N}_I(x)(x_I)^m = x^m, m = 0, 1, 2, \ldots, p. \tag{58}$$

In the refinement, the mesh is given and refined at the same scale. The shape function $\tilde{N}_I(x)$ can be regarded as a scaled reference shape function $\tilde{N}^0(\xi)$ in the parent domain with $\xi = (x - x_I)/h$. Suppose the boundary $\partial\Omega$ is smooth enough, then the following interpolation estimate holds

$$\|u(x) - \mathcal{I}u\|_{H^1(\Omega)} \leq C_1 h^p \|u\|_{H^{p+1}(\Omega)}. \tag{59}$$

where h is the mesh size and C_1 is a constant independent of h.

Proof We have

$$u(x) - \mathcal{I}u = u(x) - \sum_I \tilde{N}_I(x) u(x_I) = u(x) - \sum_{I \in \Lambda(x)} \tilde{N}_I(x) u(x_I), \tag{60}$$

where $\Lambda(x) = \{I | x \in \text{supp}\{\tilde{N}_I\} \cap \Omega\}$ is an index set supporting all the shape functions covering x.

Take Taylor expansion of $u(x_I)$ at x:

$$u(x_I) = u(x) + \sum_{m=1}^{p} \frac{1}{m!}(x_I - x)^m D^m u(x) + \frac{1}{(p+1)!}(x_I - x)^{p+1} D^{p+1} u(x + \theta(x_I - x)), \tag{61}$$

where $D^m u$ represents the m-th order derivative of $u(x)$, and $0 < \theta < 1$ depends on x and x_I. Substituting (61) into (60) yields

$$\begin{aligned} u(x) - \mathcal{I}u &= u(x) - \sum_{I \in \Lambda(x)} \tilde{N}_I(x) u(x_I) \\ &= u(x) - \sum_{I \in \Lambda(x)} \tilde{N}_I(x) \Bigg(u(x) + \sum_{m=1}^{p} \frac{1}{m!}(x_I - x)^m D^m u(x) \\ &\quad + \frac{1}{(p+1)!}(x_I - x)^{p+1} D^{p+1} u(x + \theta(x_I - x)) \Bigg) \\ &= u(x) - \Bigg(u(x) \sum_{I \in \Lambda(x)} \tilde{N}_I(x) \\ &\quad + \frac{1}{m!} D^m u(x) \sum_{m=1}^{p} \sum_{I \in \Lambda(x)} \tilde{N}_I(x)(x_I - x)^m \\ &\quad + \sum_{I \in \Lambda(x)} \frac{1}{(p+1)!}(x_I - x)^{p+1} D^{p+1} u(x + \theta(x_I - x)) \tilde{N}_I(x) \Bigg) \end{aligned} \tag{62, 63}$$

According to the reproducing property, we have

$$\mathcal{I}(y - x)^m = \sum_I \tilde{N}_I(x)(y - x_I)^m = \sum_{I \in \Lambda(x)} \tilde{N}_I(x)(y - x_I)^m = (y - x)^m, m = 1, 2, \ldots, p, \tag{64}$$

particularly, for $y = x$

$$\sum_{I \in \Lambda(x)} \tilde{N}_I(x)(x - x_I)^m = 0, m = 1, 2, \ldots, p. \tag{65}$$

So the deviation between $u(x)$ and its approximation $\mathcal{I}u$ is

$$u(x) - \mathcal{I}u = - \sum_{I \in \Lambda(x)} \frac{1}{(p+1)!}(x_I - x)^{p+1} D^{p+1} u(x + \theta(x_I - x)) \tilde{N}_I(x) \tag{66}$$

It deduces the following estimate

$$|u(x) - \mathcal{I}u| \leq \frac{1}{(p+1)!} \sum_{I \in \Lambda(x)} |x_I - x|^{p+1} |D^{p+1} u(x + \theta(x_I - x))| |\tilde{N}_I(x)| \tag{67}$$

Denote the support size of shape function $\tilde{N}_I(x)$ as rh, the maximum value of $|\tilde{N}_I(x)|$ is M. Note that the absolute value

of C-HiDeNN shape function is generally less than 1 (in the element domain). The following estimate holds

$$|u(x)-\mathcal{I}u|\leq\frac{(rh)^{p+1}}{(p+1)!}M\sum_{I\in\Lambda(x)}|D^{p+1}u(x+\theta(x_I-x))| \tag{68}$$

We can conclude that $\exists\, C_0$, with $0 < C_0 < \infty$, such that

$$\|u(x)-\mathcal{I}u\|_{L^2}\leq C_0 h^{p+1}\|u\|_{H^{p+1}(\Omega)}. \tag{69}$$

Next, we consider the derivative of $u(x)-\mathcal{I}u$. Taking the derivative of $\mathcal{I}u$ yields

$$D^1\mathcal{I}u=\sum_I D^1\tilde{N}_I(x)u(x_I). \tag{70}$$

The derivative of $u(x)-\mathcal{I}u$ is

$$\begin{aligned}
&D^1u(x)-D^1\mathcal{I}u \qquad (71)\\
&=D^1u(x)-\sum_{I\in\Lambda(x)}D^1\tilde{N}_I(x)u(x_I)\\
&=D^1u(x)-\sum_{I\in\Lambda(x)}D^1\tilde{N}_I(x)\Big(u(x)+\sum_{m=1}^{p}\frac{1}{m!}(x_I-x)^m\\
&\quad\times D^mu(x)+\frac{1}{(p+1)!}(x_I-x)^{p+1}D^{p+1}u(x+\theta(x_I-x))\Big)\\
&=u(x)-\Big(u(x)\sum_{I\in\Lambda(x)}D^1\tilde{N}_I(x)+\frac{1}{m!}D^mu(x)\\
&\quad\times\sum_{m=1}^{p}\sum_{I\in\Lambda(x)}D^1\tilde{N}_I(x)(x_I-x)^m\\
&\quad+\sum_{I\in\Lambda(x)}\frac{1}{(p+1)!}(x_I-x)^{p+1}D^{p+1}\\
&\quad\times u(x+\theta(x_I-x))D^1\tilde{N}_I(x)\Big) \qquad (72)
\end{aligned}$$

Taking the derivative of (64) with respect to x yields

$$\begin{aligned}
D^1\mathcal{I}(y-x)^m&=\sum_{I\in\Lambda(x)}D^1\tilde{N}_I(x)(y-x_I)^m\\
&=-m(y-x)^{m-1},\ m=1,2,...,p \qquad (73)
\end{aligned}$$

For $y=x$, we have

$$\sum_{I\in\Lambda(x)}D^1\tilde{N}_I(x)(x-x_I)^m=-\delta_{1,m},\ m=1,2,...,p \tag{74}$$

This leads to

$$\begin{aligned}
&D^1u(x)-D^1\mathcal{I}u\\
&=-\sum_{I\in\Lambda(x)}\frac{1}{(p+1)!}(x_I-x)^{p+1}D^{p+1}\\
&\quad\times u(x+\theta(x_I-x))D^1\tilde{N}_I(x) \qquad (75)
\end{aligned}$$

The reference shape function $\tilde{N}^0(\xi)$ is defined in the parent domain with $\xi=(x-x_I)/h$, independent of the element size h. Thus

$$D^1\tilde{N}_I(x)=\frac{1}{h}D^1_\xi\tilde{N}^0(\xi). \tag{76}$$

The estimate for the derivative of $u(x)-\mathcal{I}u$ becomes

$$\begin{aligned}
&|D^1u(x)-D^1\mathcal{I}u|\\
&=\Big|\sum_{I\in\Lambda(x)}\frac{1}{(p+1)!h}(x_I-x)^{p+1}D^{p+1}\\
&\quad u(x+\theta(x_I-x))D^1_\xi\tilde{N}^0(\xi)\Big|\\
&\leq\frac{r^{p+1}h^p}{(p+1)!}K\sum_{I\in\Lambda(x)}|D^{p+1}u(x+\theta(x_I-x))|, \qquad (77)
\end{aligned}$$

where K is the maximum value of $|D^1_\xi\tilde{N}^0(\xi)|$. We can conclude that $\exists\, C_1$ with $0 < C_1 < \infty$, such that

$$\|u(x)-\mathcal{I}u\|_{H^1}\leq C_1h^p\|u\|_{H^{p+1}(\Omega)}. \tag{78}$$

. □

For a linear second-order elliptic boundary value problem, we can use Céa's inequality [27], therefore

$$\begin{aligned}
\|u^{\text{C-HiDeNN}}(\boldsymbol{x})-u^{\text{Ext}}(\boldsymbol{x})\|_{H^1}&\leq c\inf_{v^h\in V^h}\|v^h(\boldsymbol{x})-u^{\text{Ext}}(\boldsymbol{x})\|_{H^1}\\
&\leq c\|\mathcal{I}u^{\text{Ext}}-u^{\text{Ext}}(\boldsymbol{x})\|_{H^1}\\
&\leq c\times C_1h^p\|u^{\text{Ext}}\|_{H^{p+1}}, \qquad (79)
\end{aligned}$$

where V^h represents the C-HiDeNN interpolation space.

Appendix F Illustration of C-PGD/TD solution procedure

The C-PGD solution can be solved through the so-called alternating fixed point algorithm as traditional PGD method [40], or through a minimization problem [44]. Let us consider a 3D Poisson problem as below

$$\nabla^2u(x,y,z)+b(x,y,z)=0\text{ in }\Omega\quad\text{with }\nabla u.\boldsymbol{n}=0\text{ on }\partial\Omega \tag{80}$$

where ∇ denotes the gradient operator, b is the body source term with the assumption that $b(x,y,z)=b_x(x)b_y(y)b_z(z)$.

The domain Ω is a regular domain, $\boldsymbol{n}$ is the normal vector on the surface. The weak form of the problem can be obtained by multiplying both sides of the above equation by the test function δu, which reads

$$\int_{\Omega} (\nabla \delta u)^T \nabla u \, d\Omega - \int_{\Omega} \delta u \, b \, d\Omega = 0 \tag{81}$$

Assuming the solution can be accurately approximated by M modes and $M-1$ modes are already computed, the following decomposition for u and δu read

$$\begin{aligned} u(x, y, z) = &\sum_{m=1}^{M-1} u_x^{(m)}(x) u_y^{(m)}(y) u_z^{(m)}(z) \\ &+ u_x^{(M)}(x) u_y^{(M)}(y) u_z^{(M)}(z) \end{aligned} \tag{82}$$

and

$$\begin{aligned} \delta u(x, y, z) = &\delta u_x^{(M)} u_y^{(M)} u_z^{(M)} + u_x^{(M)} \delta u_y^{(M)} u_z^{(M)} \\ &+ u_x^{(M)} u_y^{(M)} \delta u_z^{(M)} \end{aligned} \tag{83}$$

For notation simplification, we can omit the superscript M in the above equations. Therefore,

$$u(x, y, z) = \sum_{m=1}^{M-1} u_x^{(m)}(x) u_y^{(m)}(y) u_z^{(m)}(z) + u_x(x) u_y(y) u_z(z) \tag{84}$$

and

$$\delta u(x, y, z) = \delta u_x u_y u_z + u_x \delta u_y u_z + u_x u_y \delta u_z \tag{85}$$

The convolution approximation is then applied to each of the separated 1D functions $u_x(x)$, $u_y(y)$, $u_z(z)$, which reads

$$u_x(x) = \tilde{\boldsymbol{N}}(x) \boldsymbol{u}_x, \; u_y(y) = \tilde{\boldsymbol{N}}(y) \boldsymbol{u}_y, \; u_z(z) = \tilde{\boldsymbol{N}}(z) \boldsymbol{u}_z \tag{86}$$

where $\tilde{\boldsymbol{N}}$ is the convolution shape function vector formed by the 1D convolution functions in a patch domain. $\boldsymbol{u}_x, \boldsymbol{u}_y, \boldsymbol{u}_z$ are the associated nodal solution vectors in three directions. Similarly, the same convolution approximation can be applied to $\delta u_x, \delta u_y, \delta u_z$ and $u_x^{(m)}, u_y^{(m)}, u_z^{(m)}$.

The $u_x(x), u_y(y), u_z(z)$ can be computed by alternatively fixing two of the functions. For example, we can assume u_y, u_z are given by assumed values, then $\delta u_y = 0$ and $\delta u_z = 0$, $\delta u(x, y, z) = \delta u_x u_y u_z$, the weak form (81) for solving u_x under the decomposition reads

$$\begin{aligned} &\int_{\Omega} (\nabla(\delta u_x u_y u_z))^T \nabla \left(\sum_{m=1}^{M-1} u_x^{(m)} u_y^{(m)} u_z^{(m)} + u_x u_y u_z \right) d\Omega \\ &- \int_{\Omega} \delta u_x u_y u_z \, b \, d\Omega = 0 \end{aligned} \tag{87}$$

with

$$\begin{aligned} \nabla(\delta u_x u_y u_z) &= \left[\frac{\partial \delta u_x}{\partial x} u_y u_z, \; \delta u_x \frac{\partial u_y}{\partial y} u_z, \delta u_x u_y \frac{\partial u_z}{\partial z} \right]^T \\ &= \Big[\tilde{\boldsymbol{B}}_x \delta \boldsymbol{u}_x \tilde{\boldsymbol{N}}_y \boldsymbol{u}_y \tilde{\boldsymbol{N}}_z \boldsymbol{u}_z, \; \tilde{\boldsymbol{N}}_x \delta \boldsymbol{u}_x \tilde{\boldsymbol{B}}_y \boldsymbol{u}_y \tilde{\boldsymbol{N}}_z \boldsymbol{u}_z, \\ &\quad \tilde{\boldsymbol{N}}_x \delta \boldsymbol{u}_x \tilde{\boldsymbol{N}}_y \boldsymbol{u}_y \tilde{\boldsymbol{B}}_z \boldsymbol{u}_z \Big]^T \end{aligned} \tag{88}$$

and

$$\begin{aligned} \nabla(u_x^{(m)} u_y^{(m)} u_z^{(m)}) &= \left[\frac{\partial u_x^{(m)}}{\partial x} u_y^{(m)} u_z^{(m)}, \; u_x^{(m)} \frac{\partial u_y^{(m)}}{\partial y} u_z^{(m)}, \right. \\ &\quad \left. u_x^{(m)} u_y^{(m)} \frac{\partial u_z^{(m)}}{\partial z} \right]^T \\ &= \Big[\tilde{\boldsymbol{B}}_x \boldsymbol{u}_x^{(m)} \tilde{\boldsymbol{N}}_y \boldsymbol{u}_y^{(m)} \tilde{\boldsymbol{N}}_z \boldsymbol{u}_z^{(m)} \\ &\quad \tilde{\boldsymbol{N}}_x \boldsymbol{u}_x^{(m)} \tilde{\boldsymbol{B}}_y \boldsymbol{u}_y^{(m)} \tilde{\boldsymbol{N}}_z \boldsymbol{u}_z^{(m)}, \\ &\quad \tilde{\boldsymbol{N}}_x \boldsymbol{u}_x^{(m)} \tilde{\boldsymbol{N}}_y \boldsymbol{u}_y^{(m)} \tilde{\boldsymbol{B}}_z \boldsymbol{u}_z^{(m)} \Big]^T \end{aligned} \tag{89}$$

where $\tilde{\boldsymbol{B}}_x = \partial \tilde{\boldsymbol{N}}_x / \partial x$, $\tilde{\boldsymbol{B}}_y = \partial \tilde{\boldsymbol{N}}_y / \partial y$, $\tilde{\boldsymbol{B}}_z = \partial \tilde{\boldsymbol{N}}_z / \partial z$ Therefore, the Eq. (87) becomes

$$\begin{aligned} &\sum_{m=1}^{M-1} \int_{\Omega} \delta \boldsymbol{u}_x^T \tilde{\boldsymbol{B}}_x^T \tilde{\boldsymbol{B}}_x \boldsymbol{u}_x^{(m)} \boldsymbol{u}_y^T \tilde{\boldsymbol{N}}_y^T \tilde{\boldsymbol{N}}_y \boldsymbol{u}_y^{(m)} \boldsymbol{u}_z^T \tilde{\boldsymbol{N}}_z^T \tilde{\boldsymbol{N}}_z \boldsymbol{u}_z^{(m)} \, d\Omega \\ &+ \sum_{m=1}^{M-1} \int_{\Omega} \delta \boldsymbol{u}_x^T \tilde{\boldsymbol{N}}_x^T \tilde{\boldsymbol{N}}_x \boldsymbol{u}_x^{(m)} \boldsymbol{u}_y^T \tilde{\boldsymbol{B}}_y^T \tilde{\boldsymbol{B}}_y \boldsymbol{u}_y^{(m)} \boldsymbol{u}_z^T \tilde{\boldsymbol{N}}_z^T \tilde{\boldsymbol{N}}_z \boldsymbol{u}_z^{(m)} \, d\Omega \\ &+ \sum_{m=1}^{M-1} \int_{\Omega} \delta \boldsymbol{u}_x^T \tilde{\boldsymbol{N}}_x^T \tilde{\boldsymbol{N}}_x \boldsymbol{u}_x^{(m)} \boldsymbol{u}_y^T \tilde{\boldsymbol{N}}_y^T \tilde{\boldsymbol{N}}_y \boldsymbol{u}_y^{(m)} \boldsymbol{u}_z^T \tilde{\boldsymbol{B}}_z^T \tilde{\boldsymbol{B}}_z \boldsymbol{u}_z^{(m)} \, d\Omega \\ &+ \int_{\Omega} \delta \boldsymbol{u}_x^T \tilde{\boldsymbol{B}}_x^T \tilde{\boldsymbol{B}}_x \boldsymbol{u}_x \boldsymbol{u}_y^T \tilde{\boldsymbol{N}}_y^T \tilde{\boldsymbol{N}}_y \boldsymbol{u}_y \boldsymbol{u}_z^T \tilde{\boldsymbol{N}}_z^T \tilde{\boldsymbol{N}}_z \boldsymbol{u}_z \, d\Omega \\ &+ \int_{\Omega} \delta \boldsymbol{u}_x^T \tilde{\boldsymbol{N}}_x^T \tilde{\boldsymbol{N}}_x \boldsymbol{u}_x \boldsymbol{u}_y^T \tilde{\boldsymbol{B}}_y^T \tilde{\boldsymbol{B}}_y \boldsymbol{u}_y \boldsymbol{u}_z^T \tilde{\boldsymbol{N}}_z^T \tilde{\boldsymbol{N}}_z \boldsymbol{u}_z \, d\Omega \\ &+ \int_{\Omega} \delta \boldsymbol{u}_x^T \tilde{\boldsymbol{N}}_x^T \tilde{\boldsymbol{N}}_x \boldsymbol{u}_x \boldsymbol{u}_y^T \tilde{\boldsymbol{N}}_y^T \tilde{\boldsymbol{N}}_y \boldsymbol{u}_y \boldsymbol{u}_z^T \tilde{\boldsymbol{B}}_z^T \tilde{\boldsymbol{B}}_z \boldsymbol{u}_z \, d\Omega \\ &- \int_{\Omega} \delta \boldsymbol{u}_x^T \tilde{\boldsymbol{N}}_x^T b_x \boldsymbol{u}_y^T \tilde{\boldsymbol{N}}_y^T b_y \boldsymbol{u}_z^T \tilde{\boldsymbol{N}}_z^T b_z \, d\Omega \\ &= 0 \end{aligned} \tag{90}$$

The final discretized form for solving u_x is

$$\begin{aligned} &\sum_{m=1}^{M-1} \tilde{\boldsymbol{K}}_{xx} \boldsymbol{u}_x^{(m)} \boldsymbol{u}_y^T \tilde{\boldsymbol{M}}_{yy} \boldsymbol{u}_y^{(m)} \boldsymbol{u}_z^T \tilde{\boldsymbol{M}}_{zz} \boldsymbol{u}_z^{(m)} \\ &+ \sum_{m=1}^{M-1} \tilde{\boldsymbol{M}}_{xx} \boldsymbol{u}_x^{(m)} \boldsymbol{u}_y^T \tilde{\boldsymbol{K}}_{yy} \boldsymbol{u}_y^{(m)} \boldsymbol{u}_z^T \tilde{\boldsymbol{M}}_{zz} \boldsymbol{u}_z^{(m)} \\ &+ \sum_{m=1}^{M-1} \tilde{\boldsymbol{M}}_{xx} \boldsymbol{u}_x^{(m)} \boldsymbol{u}_y^T \tilde{\boldsymbol{M}}_{yy} \boldsymbol{u}_y^{(m)} \boldsymbol{u}_z^T \tilde{\boldsymbol{K}}_{zz} \boldsymbol{u}_z^{(m)} \end{aligned}$$

$$
\begin{aligned}
&+\tilde{\boldsymbol{K}}_{xx}\boldsymbol{u}_x\boldsymbol{u}_y^T\tilde{\boldsymbol{M}}_{yy}\boldsymbol{u}_y\boldsymbol{u}_z^T\tilde{\boldsymbol{M}}_{zz}\boldsymbol{u}_z\\
&+\tilde{\boldsymbol{M}}_{xx}\boldsymbol{u}_x\boldsymbol{u}_y^T\tilde{\boldsymbol{K}}_{yy}\boldsymbol{u}_y\boldsymbol{u}_z^T\tilde{\boldsymbol{M}}_{zz}\boldsymbol{u}_z\\
&+\tilde{\boldsymbol{M}}_{xx}\boldsymbol{u}_x\boldsymbol{u}_y^T\tilde{\boldsymbol{M}}_{yy}\boldsymbol{u}_y\boldsymbol{u}_z^T\tilde{\boldsymbol{K}}_{zz}\boldsymbol{u}_z\\
&-\tilde{\boldsymbol{Q}}_x\boldsymbol{u}_y^T\tilde{\boldsymbol{Q}}_y\boldsymbol{u}_z^T\tilde{\boldsymbol{Q}}_z\\
&=0
\end{aligned}
\tag{91}
$$

where $\tilde{\boldsymbol{K}}_{xx}=\int_{\Omega_x}\tilde{\boldsymbol{B}}_x^T\tilde{\boldsymbol{B}}_x\,dx$, $\tilde{\boldsymbol{K}}_{yy}=\int_{\Omega_y}\tilde{\boldsymbol{B}}_y^T\tilde{\boldsymbol{B}}_y\,dy$, $\tilde{\boldsymbol{K}}_{zz}=\int_{\Omega_z}\tilde{\boldsymbol{B}}_z^T\tilde{\boldsymbol{B}}_z\,dz$, $\tilde{\boldsymbol{M}}_{xx}=\int_{\Omega_x}\tilde{\boldsymbol{N}}_x^T\tilde{\boldsymbol{N}}_x\,dx$, $\tilde{\boldsymbol{M}}_{yy}=\int_{\Omega_y}\tilde{\boldsymbol{N}}_y^T\tilde{\boldsymbol{N}}_y\,dy$, $\tilde{\boldsymbol{M}}_{zz}=\int_{\Omega_z}\tilde{\boldsymbol{N}}_z^T\tilde{\boldsymbol{N}}_z\,dz$, with $\Omega=\Omega_x\times\Omega_y\times\Omega_z$.

Rearranging the above equation leads to the following linear system of equations

$$
\tilde{\boldsymbol{K}}\boldsymbol{u}_x=\tilde{\boldsymbol{Q}} \tag{92}
$$

with

$$
\begin{aligned}
\tilde{\boldsymbol{K}} &= \tilde{\boldsymbol{K}}_{xx}\boldsymbol{u}_y^T\tilde{\boldsymbol{M}}_{yy}\boldsymbol{u}_y\boldsymbol{u}_z^T\tilde{\boldsymbol{M}}_{zz}\boldsymbol{u}_z+\tilde{\boldsymbol{M}}_{xx}\boldsymbol{u}_y^T\tilde{\boldsymbol{K}}_{yy}\boldsymbol{u}_y\boldsymbol{u}_z^T\tilde{\boldsymbol{M}}_{zz}\boldsymbol{u}_z\\
&\quad+\tilde{\boldsymbol{M}}_{xx}\boldsymbol{u}_y^T\tilde{\boldsymbol{M}}_{yy}\boldsymbol{u}_y\boldsymbol{u}_z^T\tilde{\boldsymbol{K}}_{zz}\boldsymbol{u}_z\\
\tilde{\boldsymbol{Q}} &= \tilde{\boldsymbol{Q}}_x\boldsymbol{u}_y^T\tilde{\boldsymbol{Q}}_y\boldsymbol{u}_z^T\tilde{\boldsymbol{Q}}_z-\sum_{m=1}^{M-1}\tilde{\boldsymbol{K}}_{xx}\boldsymbol{u}_x^{(m)}\boldsymbol{u}_y^T\tilde{\boldsymbol{M}}_{yy}\boldsymbol{u}_y^{(m)}\boldsymbol{u}_z^T\tilde{\boldsymbol{M}}_{zz}\boldsymbol{u}_z^{(m)}\\
&\quad-\sum_{m=1}^{M-1}\tilde{\boldsymbol{M}}_{xx}\boldsymbol{u}_x^{(m)}\boldsymbol{u}_y^T\tilde{\boldsymbol{K}}_{yy}\boldsymbol{u}_y^{(m)}\boldsymbol{u}_z^T\tilde{\boldsymbol{M}}_{zz}\boldsymbol{u}_z^{(m)}\\
&\quad-\sum_{m=1}^{M-1}\tilde{\boldsymbol{M}}_{xx}\boldsymbol{u}_x^{(m)}\boldsymbol{u}_y^T\tilde{\boldsymbol{M}}_{yy}\boldsymbol{u}_y^{(m)}\boldsymbol{u}_z^T\tilde{\boldsymbol{K}}_{zz}\boldsymbol{u}_z^{(m)}
\end{aligned}
\tag{93}
$$

By solving the above equation, an estimate of $\boldsymbol{u}_x$ and therefore u_x can be obtained. Similarly, we can solve for u_y and u_z. This alternative fixed point procedure should be repeated until the convergence the product of $u_xu_yu_z$. This is the solution for one C-PGD mode and can be used for computing incrementally all the modes by varying M from 1 to a given number. The total number of modes can be determined by the convergence criterion: $\|\sum_{m=1}^{M-1}u_x^{(m)}(x)u_y^{(m)}(y)u_z^{(m)}(z)-u_x^{(M)}(x)u_y^{(M)}(y)u_z^{(M)}(z)\|$ is small enough.

By adopting the TD definition in [39], the C-TD can use the same solution procedure to give a rough estimate number of modes and then optimize all the current modes together. The advantages in doing so is to reduce the necessary number of modes. In our work, we are interested in the final convergent accuracy of C-PGD/TD with comparison to traditional FEM and PGD/TD. The number of modes is ignored. Hence, we do not distinguish the C-PGD and C-TD in this paper as they both can give a better accuracy than traditional methods with the proposed convolution approximation.

Appendix G Error bound analysis of different methods

Assuming a convex potential energy Π exits for a problem, the numerical solution to the problem can be obtained by solving the following minimization problem

$$
u^h=\arg\min_{u^{h*}\in V^h}\Pi(u^{h*}(\boldsymbol{x})) \tag{94}
$$

where V^h denotes a given approximation space. Now, consider two approximation spaces V_1^h, V_2^h with $V_1^h\subset V_2^h$. Their corresponding solutions by solving the above problem are respectively u_1^h and u_2^h. It is easy to know that the optimized potential energy has following relationship

$$
\Pi(u^{\text{Ext}})\le\Pi(u_2^h)\le\Pi(u_1^h) \tag{95}
$$

where u^{Ext} is the exact solution. By the convexity of the problem, we have

$$
\|u_2^h-u^{\text{Ext}})\|_E\le\|u_1^h-u^{\text{Ext}})\|_E \tag{96}
$$

By analogy, denoting the approximation spaces of TD with infinite modes, C-TD with infinite modes, FEM, C-FEM, C-HiDeNN, DeNN respectively by V^{TD}, $V^{\text{C-TD}}$, V^{FEM}, $V^{\text{C-FEM}}$, $V^{\text{C-FEM}}$, $V^{\text{C-HiDeNN}}$, V^{DeNN}, with the following relationship

$$
V^{\text{TD}}\subset V^{\text{FEM}}\subset V^{\text{C-TD}}\subset V^{\text{C-FEM}}\subset V^{\text{C-HiDeNN}}\subset V^{\text{DeNN}} \tag{97}
$$

The optimized potential energy follows

$$
\begin{aligned}
\Pi(u^{\text{Ext}}) &\le \Pi(u^{\text{DeNN}})\\
&\le \Pi(u^{\text{C-HiDeNN}})\\
&\le \Pi(u^{\text{C-FEM}})\le\Pi(u^{\text{C-TD}})\\
&\le \Pi(u^{\text{FEM}})\le\Pi(u^{\text{TD}})
\end{aligned}
\tag{98}
$$

Therefore, we have

$$
\begin{aligned}
\|u^{\text{DeNN}}-u^{\text{Ext}}\|_E &\le \|u^{\text{C-HiDeNN}}-u^{\text{Ext}}\|_E\\
&\le \|u^{\text{C-FEM}}-u^{\text{Ext}}\|_E\\
&\le \|u^{\text{C-PGD/TD}}-u^{\text{Ext}}\|_E\\
&\le \|u^{\text{FEM}}-u^{\text{Ext}}\|_E\\
&\le \|u^{\text{PGD/TD}}-u^{\text{Ext}}\|_E
\end{aligned}
\tag{99}
$$

Here, since we are only interested in the approximation space of TD/PGD, we do not distinguish the two methods in this analysis.

References

1. Liu WK, Li S, Park HS (2022) Eighty years of the finite element method: birth, evolution, and future. Arch Comput Methods Eng 29:4431–4453
2. Liu WK, Jun S, Zhang YF (1995) Reproducing kernel particle methods. Int J Numer Methods Fluids 20(8–9):1081–1106
3. Liu WK, Jun S, Li S, Adee J, Belytschko T (1995) Reproducing kernel particle methods for structural dynamics. Int J Numer Methods Eng 38(10):1655–1679
4. Chen JS, Liu WK, Hillman MC, Chi SW, Lian Y, Bessa MA (2017) Reproducing kernel particle method for solving partial differential equations. In: Stein E, de Borst R, Hughes TJR (eds) Encyclopedia of computational mechanics, 2nd edn. Wiley, Hoboken, pp 1–44
5. Belytschko T, Lu YY, Gu L (1994) Element-free Galerkin methods. Int J Numer Methods Eng 37(2):229–256
6. Nayroles B, Touzot G, Villon P (1992) Generalizing the finite element method: diffuse approximation and diffuse elements. Comput Mech 10(5):307–318
7. Liu M, Liu G (2010) Smoothed particle hydrodynamics (sph): an overview and recent developments. Arch Comput Methods Eng 17(1):25–76
8. Liu W-K, Li S, Belytschko T (1997) Moving least-square reproducing kernel methods (I) methodology and convergence. Comput Methods Appl Mech Eng 143(1–2):113–154
9. Liu WK, Chen Y, Uras RA, Chang CT (1996) Generalized multiple scale reproducing kernel particle methods. Comput Methods Appl Mech Eng 139(1–4):91–157
10. Li S, Liu WK (1996) Moving least-square reproducing kernel method part ii: Fourier analysis. Comput Methods Appl Mech Eng 139(1–4):159–193
11. Hughes TJ, Cottrell JA, Bazilevs Y (2005) Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. Comput Methods Appl Mech Eng 194(39–41):4135–4195
12. Bazilevs Y, Calo VM, Cottrell JA, Evans JA, Hughes TJR, Lipton S, Scott MA, Sederberg TW (2010) Isogeometric analysis using t-splines. Comput Methods Appl Mech Eng 199(5–8):229–263
13. De Lorenzis L, Wriggers P, Hughes TJ (2014) Isogeometric contact: a review. GAMM-Mitteilungen 37(1):85–123
14. Belytschko T, Organ D, Gerlach C (2000) Element-free Galerkin methods for dynamic fracture in concrete. Comput Methods Appl Mech Eng 187(3–4):385–399
15. Duarte CA, Oden JT (1996) An hp adaptive method using clouds. Comput Methods Appl Mech Eng 139(1–4):237–262
16. Duarte CA, Oden JT (1996) H-p clouds-an h-p meshless method. Numer Methods Partial Differ Equ Int J 12(6):673–705
17. Oden JT, Duarte C, Zienkiewicz OC (1998) A new cloud-based hp finite element method. Comput Methods Appl Mech Eng 153(1–2):117–126
18. Melenk JM, Babuška I (1996) The partition of unity finite element method: basic theory and applications. Comput Methods Appl Mech Eng 139(1–4):289–314
19. Griebel M, Schweitzer MA (2000) A particle-partition of unity method for the solution of elliptic, parabolic, and hyperbolic PDEs. SIAM J Sci Comput 22(3):853–890
20. Chen J-S, Wang H-P (2000) New boundary condition treatments in meshfree computation of contact problems. Comput Methods Appl Mech Eng 187(3–4):441–468
21. Chen J-S, Han W, You Y, Meng X (2003) A reproducing kernel method with nodal interpolation property. Int J Numer Methods Eng 56(7):935–960
22. Wagner GJ, Liu WK (2000) Application of essential boundary conditions in mesh-free methods: a corrected collocation method. Int J Numer Methods Eng 47(8):1367–1379
23. Wagner GJ, Liu WK (2001) Hierarchical enrichment for bridging scales and mesh-free boundary conditions. Int J Numer Methods Eng 50(3):507–524
24. Han W, Wagner GJ, Liu WK (2002) Convergence analysis of a hierarchical enrichment of Dirichlet boundary conditions in a mesh-free method. Int J Numer Methods Eng 53(6):1323–1336
25. Huerta A, Fernández-Méndez S (2000) Enrichment and coupling of the finite element and meshless methods. Int J Numer Methods Eng 48(11):1615–1636
26. Huerta A, Fernández-Méndez S, Liu WK (2004) A comparison of two formulations to blend finite elements and mesh-free methods. Comput Methods Appl Mech Eng 193(12–14):1105–1117
27. Liu WK, Han W, Lu H, Li S, Cao J (2004) Reproducing kernel element method. Part i: theoretical formulation. Comput Methods Appl Mech Eng 193(12–14):933–951
28. Li S, Lu H, Han W, Liu WK, Simkins DC (2004) Reproducing kernel element method part ii: globally conforming Im/Cn hierarchies. Comput Methods Appl Mech Eng 193(12–14):953–987
29. Lu H, Li S, Simkins DC Jr, Liu WK, Cao J (2004) Reproducing kernel element method part iii: generalized enrichment and applications. Comput Methods Appl Mech Eng 193(12–14):989–1011
30. Hornik K, Stinchcombe M, White H (1989) Multilayer feedforward networks are universal approximators. Neural Netw 2(5):359–366
31. Cai S, Mao Z, Wang Z, Yin M, Karniadakis GE (2022) Physics-informed neural networks (pinns) for fluid mechanics: a review. Acta Mech Sin 37:1727–1738
32. Raissi M, Perdikaris P, Karniadakis GE (2021) Physics informed learning machine. Mar. 30. US Patent 10,963,540
33. Lee K, Trask NA, Patel RG, Gulian MA, Cyr EC (2021) Partition of unity networks: deep hp-approximation. arXiv:2101.11256
34. Jin P, Zhang Z, Zhu A, Tang Y, Karniadakis GE (2020) Sympnets: intrinsic structure-preserving symplectic networks for identifying Hamiltonian systems. Neural Netw 132:166–179
35. Hernández Q, Badías A, González D, Chinesta F, Cueto E (2021) Structure-preserving neural networks. J Comput Phys 426:109950
36. Cheng L, Wagner GJ (2022) A representative volume element network (RVE-net) for accelerating RVE analysis, microscale material identification, and defect characterization. Comput Methods Appl Mech Eng 390:114507
37. Cuomo S, Di Cola VS, Giampaolo F, Rozza G, Raissi M, Piccialli F (2022) Scientific machine learning through physics-informed neural networks: where we are and what's next. J Sci Comput 92(3):88
38. Zhang L, Cheng L, Li H, Gao J, Yu C, Domel R, Yang Y, Tang S, Liu WK (2021) Hierarchical deep-learning neural networks: finite elements and beyond. Comput Mech 67(1):207–230
39. Zhang L, Lu Y, Tang S, Liu WK (2022) Hidenn-td: reduced-order hierarchical deep learning neural networks. Comput Methods Appl Mech Eng 389:114414
40. Ammar A, Mokdad B, Chinesta F, Keunings R (2006) A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids. J Nonnewton Fluid Mech 139(3):153–176
41. Chinesta F, Ladeveze P, Cueto E (2011) A short review on model order reduction based on proper generalized decomposition. Arch Comput Methods Eng 18(4):395–404
42. Kazemzadeh-Parsi MJ, Ammar A, Duval JL, Chinesta F (2021) Enhanced parametric shape descriptions in PGD-based space separated representations. Adv Model Simul Eng Sci 8(1):1–28
43. Lu Y, Jones KK, Gan Z, Liu WK (2020) Adaptive hyper reduction for additive manufacturing thermal fluid analysis. Comput Methods Appl Mech Eng 372:113312
44. Lu Y, Li H, Saha S, Mojumder S, Al Amin A, Suarez D, Liu Y, Qian D, Kam Liu W (2021) Reduced order machine learning finite element methods: concept, implementation, and future applications. Comput Model Eng Sci 129(3):1351–1371. https://doi.org/10.32604/cmes.2021.017719

45. Park C, Lu Y, Saha S, Xue T, Guo J, Mojumder S, Apley D, Wagner G, Liu W (2023)Convolution hierarchical deep-learning neural network (c-hidenn) with graphics processing unit (GPU) acceleration. Comput Mech
46. Li H, Knapik S, Li Y, Park C, Guo J, Mojumder S, Lu Y, Chen W, Apley D, Liu W, Convolution Hierarchical Deep-Learning Neural Network Tensor Decomposition (C-HiDeNN-TD) for high-resolution topology optimization. Comput Mech (2023). https://doi.org/10.1007/s00466-023-02333-8
47. Belytschko T, Liu WK, Moran B, Elkhodary K (2014) Nonlinear finite elements for continua and structures. Wiley, Hoboken
48. Bessa M, Foster J, Belytschko T, Liu WK (2014) A meshfree unification: reproducing kernel peridynamics. Comput Mech 53(6):1251–1264
49. Li S, Liu WK (2007) Meshfree particle methods. Springer, Berlin
50. Hughes TJ (2012) The finite element method: linear static and dynamic finite element analysis. Courier Corporation, North Chelmsford
51. Wendland H (1999) Meshless Galerkin methods using radial basis functions. Math Comput 68(228):1521–1531
52. Wang J, Liu G (2002) A point interpolation meshless method based on radial basis functions. Int J Numer Methods Eng 54(11):1623–1648
53. Tian R (2013) Extra-dof-free and linearly independent enrichments in GFEM. Comput Methods Appl Mech Eng 266:1–22
54. Petersen P, Raslan M, Voigtlaender F (2021) Topological properties of the set of functions generated by neural networks of fixed size. Found Comput Math 21:375–444
55. Bartlett PL, Ben-David S (2002) Hardness results for neural network approximation problems. Theor Comput Sci 284(1):53–66
56. Blum AL, Rivest RL (1992) Training a 3-node neural network is NP-complete. Neural Netw 5(1):117–127. https://doi.org/10.1016/S0893-6080(05)80010-3
57. Judd JS (1987) Learning in networks is hard. In: Proceedings of 1st international conference on neural networks, San Diego, California, IEEE
58. Kolda TG, Bader BW (2009) Tensor decompositions and applications. SIAM Rev 51(3):455–500
59. Sidiropoulos ND, De Lathauwer L, Fu X, Huang K, Papalexakis EE, Faloutsos C (2017) Tensor decomposition for signal processing and machine learning. IEEE Trans Signal Process 65(13):3551–3582
60. Papalexakis EE, Faloutsos C, Sidiropoulos ND (2012) Parcube: sparse parallelizable tensor decompositions. In: Machine learning and knowledge discovery in databases: European conference, ECML PKDD 2012, Bristol, UK, September 24–28, 2012. Proceedings, Part I 23, Springer, pp 521–536
61. Song J (2001) Optimal representation of high-dimensional functions and manifolds in low-dimensional visual space (in Chinese). Chin Sci Bull 46(12):977–984
62. Lu Y, Blal N, Gravouil A (2018) Adaptive sparse grid based HOPGD: toward a nonintrusive strategy for constructing space–time welding computational vademecum. Int J Numer Methods Eng 114(13):1438–1461
63. Lu Y, Blal N, Gravouil A (2018) Multi-parametric space–time computational vademecum for parametric studies: application to real time welding simulations. Finite Elem Anal Des 139:62–72
64. Lu Y, Blal N, Gravouil A (2019) Datadriven HOPGD based computational vademecum for welding parameter identification. Comput Mech 64(1):47–62
65. Badrou A, Bel-Brunon A, Hamila N, Tardif N, Gravouil A (2020) Reduced order modeling of an active multi-curve guidewire for endovascular surgery. Comput Methods Biomech Biomed Eng 23(sup1):S23–S24
66. Blal N, Gravouil A (2019) Non-intrusive data learning based computational homogenization of materials with uncertainties. Comput Mech 64(3):807–828
67. Rozza G, Huynh DB, Patera AT (2008) Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations: application to transport and continuum mechanics. Arch Comput Methods Eng 15(3):229
68. Rozza G, Veroy K (2007) On the stability of the reduced basis method for stokes equations in parametrized domains. Comput Methods Appl Mech Eng 196(7):1244–1260
69. Sirovich L (1987) Turbulence and the dynamics of coherent structures. I. Coherent structures. Q Appl Math 45(3):561–571
70. Amsallem D, Farhat C (2008) Interpolation method for adapting reduced-order models and application to aeroelasticity. AIAA J 46(7):1803–1813
71. Kerfriden P, Goury O, Rabczuk T, Bordas SP (2013) A partitioned model order reduction approach to rationalise computational expenses in nonlinear fracture mechanics. Comput Methods Appl Mech Eng 256:169–188
72. Amsallem D, Zahr M, Choi Y, Farhat C (2015) Design optimization using hyper-reduced-order models. Struct Multidiscip Optim 51(4):919–940
73. Ryckelynck D (2009) Hyper-reduction of mechanical models involving internal variables. Int J Numer Methods Eng 77(1):75–89
74. Scanff R, Néron D, Ladevèze P, Barabinot P, Cugnon F, Delsemme J-P (2022) Weakly-invasive LATIN-PGD for solving time-dependent non-linear parametrized problems in solid mechanics. Comput Methods Appl Mech Eng 396:114999
75. Falcó A, Hackbusch W, Nouy A (2019) On the Dirac–Frenkel variational principle on tensor banach spaces. Found Comput Math 19:159–204
76. Hackbusch W (2012) Tensor spaces and numerical tensor calculus, vol 42. Springer, Berlin
77. Schaback R, Wendland H (2001) Characterization and construction of radial basis functions. In: Multivariate approximation and applications, pp 1–24