Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2023.025983

ARTICLE

New Perspective to Isogeometric Analysis: Solving Isogeometric Analysis Problem by Fitting Load Function

Jingwen Ren[1] **and Hongwei Lin**[1,2,*]

[1]School of Mathematical Sciences, Zhejiang University, Hangzhou, 310058, China

[2]State Key Lab. of CAD&CG, Zhejiang University, Hangzhou, 310058, China

*Corresponding Author: Hongwei Lin. Email: hwlin@zju.edu.cn

Received: 08 August 2022 Accepted: 20 October 2022

ABSTRACT

Isogeometric analysis (IGA) is introduced to establish the direct link between computer-aided design and analysis. It is commonly implemented by Galerkin formulations (isogeometric Galerkin, IGA-G) through the use of nonuniform rational B-splines (NURBS) basis functions for geometric design and analysis. Another promising approach, isogeometric collocation (IGA-C), working directly with the strong form of the partial differential equation (PDE) over the physical domain defined by NURBS geometry, calculates the derivatives of the numerical solution at the chosen collocation points. In a typical IGA, the knot vector of the NURBS numerical solution is only determined by the physical domain. A new perspective on the IGA method is proposed in this study to improve the accuracy and convergence of the solution. Solving the PDE with IGA can be regarded as fitting the load function defined on the NURBS geometry (right-hand side) with derivatives of the NURBS numerical solution (left-hand side). Moreover, the design of the knot vector has a close relationship to the NURBS functions to be fitted in the area of data fitting in geometric design. Therefore, the detected feature points of the load function are integrated into the initial knot vector of the physical domain to construct the new knot vector of the numerical solution. Then, they are connected seamlessly with the IGA-C framework for its great potential combining the accuracy and smoothness merits with the computational efficiency, which we call *isogeometric collocation by fitting load function (IGA-CL)*. In numerical experiments, we implement our method to solve 1D, 2D, and 3D PDEs and demonstrate the improvement in accuracy by comparing it with the standard IGA-C method. We also verify the superiority in the accuracy of our knot selection scheme when employed in the IGA-G method, which we call *isogeometric Galerkin by fitting load function (IGA-GL)*.

KEYWORDS

Isogeometric analysis; collocation methods; feature point detection; knot vector

1 Introduction

Isogeometric analysis (IGA) [1] is a computational mechanics technology that advances the seamless integration between computer-aided design (CAD) and computer-aided engineering (CAE). Although finite element analysis (FEA) [2] gains widespread applications in physical simulation, IGA uses the same nonuniform rational B-spline (NURBS) bases [3] as the physical domain space to

construct the numerical solution domain space. Based on these NURBS basis functions, IGA can directly deal with the NURBS-based CAD model. This condition avoids tedious and time-consuming mesh generation in classical FEA when CAD model simulation is performed. Therefore, IGA saves a considerable amount of computation and greatly improves the numerical accuracy of the solution.

IGA has been initially implemented by Galerkin formulations (isogeometric Galerkin (IGA-G)) [1], thereby raising the efficiency issue due to the numerical integration. Isogeometric collocation (IGA-C) [4] focuses on solving the strong form of partial differential equations (PDEs) [5] rather than the weak form in IGA-G to promote the accuracy-to-computational-cost ratio. Substituting the derivatives of the NURBS numerical solution at some collocation points into the PDE leads to a linear system. The selection of these collocation points is the key to accuracy and convergence rate.

When PDEs are solved over a physical domain defined by a NURBS geometry, typical IGA commonly takes the NURBS as the bases of the approximation solution space, which is the same as the NURBS bases that construct the physical domain space. The NURBS bases are determined totally by the knot vector [3]. This fact indicates that the knot vectors for the NURBS bases of the solution field are directly inherited from the knot vectors of the geometry representation. However, few previous works investigated the influence of the knot vector on the approximating accuracy and convergence rate of the numerical solution. On the other hand, in the area of the data fitting in geometric design, the feature points that depict the shape and characteristics of the NURBS function to be fitted are vital to the knot vector that constructs the NURBS bases [3,6,7]. A theorem stating that the spline numerical solution and its k-order derivative have the same breakpoint sequence has been proposed and proved in [8]. Thus, we can treat the left-hand side and right-hand side of the PDE as a fitting problem, that is, approximating the load function defined by the NURBS geometry (right-hand side) with the derivatives of the NURBS numerical solution (left-hand side).

A new method of IGA is proposed in this study to improve the accuracy when PDEs are solved over the physical domain defined by NURBS geometry. We exploit the idea that the knot vector of the NURBS numerical solution is decided together by the load function and geometry representation. The feature points of the load function over the physical domain are detected in 1D, 2D, and 3D cases. Then, we merge these feature points into the initial knot vector of the spline representation of the physical domain. Given the great potential of the computational efficiency of IGA-C, we focus on the performance of the IGA-C framework with our generated knot vector. Thus, we name it as *IGA-C by fitting load function (IGA-CL)* method. In numerical experiments, we implement our IGA-CL framework to solve 1D, 2D, and 3D PDEs at Greville abscissae [4] and superconvergent (SC) collocation [9]. We demonstrate the efficiency of our method in terms of numerical accuracy in analysis by comparing it with the typical IGA-C method, where the knot vector of the numerical solution is the same as the knot vector of the physical domain. We also show the superiority of the knot selection scheme applied in the IGA-G framework over the typical IGA-G, which we call *IGA-G by fitting load function (IGA-GL)*.

The rest of this paper is structured as follows: Section 1.1 provides the related work. The preliminary of the IGA is provided in Section 2. We present an overview of our IGA-CL framework in Section 3. Section 4 presents the strategy to detect feature points of load functions of the PDEs in 1D, 2D, and 3D cases. Section 4.4 introduces the method to generate the integrated knot vectors and proposes our IGA-CL framework. Section 5 conducts experiments for solving PDEs in 1D, 2D, and 3D cases and applying the IGA-G method. The comparisons exhibit the influences of numerical accuracy in terms of our knot selection. Finally, the conclusions and perspectives on future works are detailed in Section 6.

1.1 Related Work

IGA-G and applications. IGA was first introduced by [1] in 2005 to bridge the gap between CAD and CAE. The core idea of IGA is to utilize the same smooth and high-order NURBS bases for both the geometry parameterization and solution approximation. The finite element method (FEM) [2] in analysis has grown strong sufficiently. However, FEM always suffers from the time-consuming mesh generation and mesh approximation error. IGA possesses some additional merits over the traditional FEM [1,10], such as high continuity, exact representation of geometry, and low computational cost, because IGA directly takes the NURBS-expressed physical domain as the computational domain and computes on the control mesh.

The standard IGA method is implemented by Galerkin formulation (IGA-G) based on the weak form of the problem. The stiffness matrix assembly involves many numerical integrations, such as the commonly used Gauss quadrature. The integration efficiency in IGA has been first considered by [11]. Then, Gauss-Galerkin quadrature rules [12] and optimal and reduced quadrature rules [13] were developed. A promising approach for reusing the basis function evaluations in the numerical integration has been recently presented to improve the accuracy and time cost in [14]. In addition to tensor product B-splines, the isogeometric Galerkin matrices with hierarchical B-splines were efficiently computed through the interpolation, look-up and sum factorization techniques [15,16]. In the past 20 years, IGA has been widely applied in physical simulations such as structural mechanics [17–19], fluid mechanics [20–22], elastic mechanics [23–25], and shape optimization problems [26–28]. Its overview and computer implementation aspects were presented by [29].

IGA-C and applications. Although the mainstream IGA-G method based on the discretization of the weak form has the optimal convergence of $\mathcal{O}(h^{p+1})$ in the L^2 norm [9], the computational efficiency relying on the integration is relatively low. Inspired by this, Auricchio et al. [4] came up with the IGA-C method. This method simply chooses some collocation points in the computational domain, substitutes the NURBS numerical solution at these collocation points into the strong form of the PDE, and finally generates a linear system. Unlike IGA-G, IGA-C considers both the superior accuracy and smoothness of NURBS basis functions and the low computational cost, which is a compromise between the accuracy and the computational time. The assembly of the stiffness matrix becomes simple and easy, and the bandwidth of the matrix becomes narrow [30]. Therefore, IGA-C is more competitive than IGA-G to some extent with respect to the accuracy-to-computational-cost ratio [30,31].

The research of IGA-C is mostly limited by the options of the ideal collocation points. In IGA-C, the commonly used collocation points are Greville abscissae [4] and Demko abscissae [30,32]. Demko abscissae are obtained from the extreme points of Chebyshev splines, and Greville abscissae are acquired from the averaging operation of the knot vector. However, their convergence orders are $\mathcal{O}(h^{p-1})$ for the odd degree p of the NURBS bases and $\mathcal{O}(h^{p})$ for the even degree p. On the basis of superconvergent theory, Anitescu et al. developed SC points [9] to advance the convergence rate. They improved the convergence order of the IGA-C solution to the optimal $\mathcal{O}(h^{p+1})$ for odd degree p. It has been proven that Cauchy-Galerkin points (CG points) exist such that the IGA-C solution can be the same as the IGA-G solution, but the exact locations of CG points are hard to determine [33]. The approximated collocation points are verified to have the convergence order of $\mathcal{O}(h^{p})$ for both odd and even p. Montardini et al. found a subset of SC points and obtained the optimal convergence order $\mathcal{O}(h^{p+1})$ for odd p [34]. Wang et al. established a superconvergent IGA-C method collocating at Greville points that utilized the recursive gradients to replace the conventional gradients of the bases [35]. Moreover, Atroshchenko et al. [36] attempted to decouple the geometry and simulation and yielded

satisfactory results. Timoshenko beam and rod simulations are solved by Beirão et al. with a locking-free collocation approach [37,38]. IGA-C was also applied in fluid-structure interaction problems [39] and poromechanics problems [40]. For nearly-incompressible elasticity problems, Morganti et al. [41] employed IGA-C to achieve the same accuracy results as compressible elasticity via the mixed displacement-pressure method. An overview of IGA-C methods with convergence rates can be found in [42].

Data point sampling and fitting. In geometric design, the quality of curve/surface fitting is related to the quality of data point sampling. Efficient sampling methods enable the reconstruction of a curve/surface with a limited amount of points [43]. Filip et al. [44] estimated the number of isometric parameter points using the bounds on derivatives. Park [45] utilized an offset envelope of a line segment and presented the approximation of error-bounded polygons. This approximation plays an important role in Hausdorff distance between the control segments and the corresponding curves. Razdan [46] discussed the uniform sampling of arc length or total curvature. Then an iteration algorithm was proposed to sample arc length or weighted arc length uniformly and the bending energy [47]. Pagani et al. [43] have acquired certain sampled points in terms of the mean of the arc length and total curvature. The reconstruction of the curve and surface outperforms the uniform sampling method. Instead of the uniform sampling method, Lu et al. [48] focused on the sampling feature points on planar parameter curves to retain the shapes of the original curves and weigh the arc length and the total curvature to sample points. This method is different from uniform sampling but closely related to the feature points.

2 Preliminary

2.1 Isogeometric Galerkin Method

In CAD, the physical domain is expressed by a NURBS mapping from a parametric domain $\hat{\Omega}$, that is, $\mathbf{F}:\hat{\Omega}\rightarrow\Omega$. The physical domain Ω can be written as

$$\mathbf{x}(u)=\sum_{i=0}^{N}N_{i,p}(u)\,\mathbf{P}_i, \tag{1}$$

where $\{\mathbf{P}_i\}_{i=0,\ldots,N}$ are control points, and $\{N_{i,p}\}_{i=0,\ldots,N}$ are the p-degree NURBS basis functions defined on the open knot vector $U=\{u_0,u_1,\ldots,u_{N+p+1}\}$. The u_i are called *knots* satisfying

$$0=u_0=\cdots=u_p\leq u_{p+1}\leq\cdots\leq u_{N+1}=\cdots=u_{N+p+1}=1.$$

We name $\{u_{i_0},u_{i_1},\ldots,u_{i_K}\},u_{i_k}<u_{i_{k+1}},k=0,\ldots,K-1$ the *breakpoint sequence*. Then, the ith NURBS basis function $N_{i,p}(u)$ is defined as ([3])

$$\begin{aligned}
B_i^0(u)&=\begin{cases}1 & u_i\leq u<u_{i+1},\\ 0 & else,\end{cases}\\
B_i^p(u)&=\frac{u-u_i}{u_{i+p}-u_i}B_i^{p-1}(u)+\frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}}B_{i+1}^{p-1}(u),\\
N_{i,p}(u)&=\frac{B_i^p(u)\omega_i}{\sum_{i'=0}^{n}B_{i'}^p(u)\omega_{i'}},
\end{aligned} \tag{2}$$

where $\{\omega_i\}_{i=0,\ldots,N}$ are weights. For high dimensions, the basis functions are calculated by their tensor product. In particular, when all the weights are 1, $\{N_{i,p}\}_{i=0,\ldots,N}$ are B-spline basis functions $\{B_i^p\}_{i=0,\ldots,N}$.

In particular, let the following Laplace equation over the physical domain defined by a NURBS geometry Ω be

$$\begin{aligned} \Delta T + f &= 0 \quad in \quad \Omega, \\ T &= g \quad on \quad \Gamma_D, \\ \nabla T \cdot \mathbf{n} &= h \quad on \quad \Gamma_N, \\ \beta T + \nabla T \cdot \mathbf{n} &= r \quad on \quad \Gamma_R, \end{aligned} \tag{3}$$

where the Dirichlet boundary Γ_D, Neumann boundary Γ_N, and Robin boundary Γ_R form the boundary $\partial\Omega$. f, g, h, r are all given functions, and T is the unknown function. We call Eq. (3) the strong form of the PDE.

The IGA-G method ([1]) focuses on the weak form of the PDE, which is given by

$$\int_\Omega \nabla\omega \cdot \nabla T d\Omega + \beta \int_{\Gamma_R} \omega T d\Gamma = \int_\Omega \omega f \Omega + \beta \int_{\Gamma_N} \omega h d\Gamma + \beta \int_{\Gamma_R} \omega r d\Gamma, \tag{4}$$

where the weighting function $\omega \in \mathscr{V}$, $\mathscr{V} = \{\omega | \omega \in H^1(\Omega), \omega|_{\gamma_D} = 0\}$, the trial solution $T \in \mathscr{S}$, $\mathscr{S} = \{T | T \in H^1(\Omega), T|_{\gamma_D} = g\}$, and H^1 is a Sobolev space. The weak form (4) is then rewritten by a bilinear operator $a(\cdot,\cdot)$ and a linear form $L(\cdot)$ as

$$a(\omega, T) = L(\omega). \tag{5}$$

Galerkin's method aims to construct the finite-dimensional approximations of the spaces $\mathscr{S}$, $\mathscr{V}$, i.e., $\mathscr{S}^h \in \mathscr{S}$, $\mathscr{V}^h \in \mathscr{V}$. In the IGA-G method, these spaces can be spanned by the same NURBS bases as the physical domain Ω. Let the unknown numerical solution T_r be a linear combination of these basis functions with $N+1$ unknown coefficients, that is,

$$T_r(u) = \sum_{i=0}^{N} N_{i,p}(u)\, t_i. \tag{6}$$

Then the weak form can be transformed into

$$a(\omega_r, T_r) = L(\omega_r). \tag{7}$$

By assembling the system of the weak form (7), we finally solve the following linear equations and obtain the solution T_r:

$$\mathbf{K}\mathbf{t} = \mathbf{b}. \tag{8}$$

2.2 Isogeometric Collocation Method

The IGA-C method ([4]) is also proposed to solve the PDE. In general, the boundary value problem (BVP) can be summarized as follows:

$$\begin{cases} \mathfrak{L}T = f \quad in \quad \Omega, \\ \mathfrak{G}T = g \quad on \quad \partial\Omega, \end{cases} \tag{9}$$

where $\mathfrak{L}$ represents a scalar differential operator, T is the unknown solution, and the given right-hand-side f is called the *load function*. The boundary vector operator $\mathfrak{G}T$ represents the boundary conditions.

IGA-C solves the equations by introducing a set of collocation points $\{\bar{\mathbf{u}}_k\}_{k=0,\ldots,N_{int}}$ in the parametric domain $\hat{\Omega}$ and $\{\bar{\mathbf{u}}_k\}_{k=N_{int}+1,\ldots,N_{int}+N_{bd}}$ $(N_{int}+N_{bd} \geq N)$ on the boundary of parametric domain $\partial\hat{\Omega}$. The corresponding points of physical domain are $\bar{\mathbf{x}}_k = \mathbf{F}(\bar{\mathbf{u}}_k), k = 0, \ldots, N_{int}+N_{bd}$. Thus, replacing the unknown T of the BVP with the expression of T_r at $\{\bar{\mathbf{x}}_k\}_{i=0,\ldots,N_{int}+N_{bd}}$ leads to a linear system:

$$\begin{cases} \mathfrak{L}T_r(\bar{\mathbf{x}}_k) = f(\bar{\mathbf{x}}_k) & k = 0, \ldots, N_{int}, \\ \mathfrak{G}T_r(\bar{\mathbf{x}}_k) = g(\bar{\mathbf{x}}_k) & k = N_{int}+1, \ldots, N_{int}+N_{bd}. \end{cases} \tag{10}$$

When $N_{int}+N_{bd} > N$, the linear system can be solved in least-squares fashion. In this study, we only discuss the common case, where $N_{int}+N_{bd} = N$, which has a unique solution of $\{t_i\}_{i=0,\ldots,N}$. The choice of the collocation points is the crucial part of the procedure, affecting directly the accuracy and convergence of the numerical solution. Some possible options exist, such as Greville abscissae ([4]), SC points ([9]), CG points ([33]), and clustered SC points ([34]). The collocation points of these schemes are generated based on the knot vector U. For instance, the typical selection, Greville abscissae in 1D, determines the collocation points as $\bar{u}_i = \frac{1}{p}\left(u_i + u_{i+1} + \cdots + u_{i+p-1}\right), i = 1, \ldots, N+1$. The SC collocation points are generated proportionally from SC points of the spline basis of the degree p on the interval $[-1, 1]$ to the knot interval $[u_i, u_{i+1}], i = p, \ldots, N-p-1$.

3 Algorithm Overview

The IGA-G and IGA-C methods involve the derivatives of the NURBS basis functions when the numerical solution T_r is brought into the PDE. Without loss of generality, a 1D second-order differential equation $-d^2T/dx^2 = f(x)$ is taken as an example. On the one hand, NURBS basis functions and their derivatives have the same breakpoint sequence; thus, the breakpoints of T_r should be the same as the breakpoints of $-d^2T_r/dx^2$. For instance, Fig. 1 demonstrates that a cubic NURBS basis and its derivatives share the same breakpoints. On the other hand, we can treat the left-hand side and right-hand side of the PDE as a fitting problem. In particular, we approximate the load function $f(x)$ on the NURBS-expressed physical domain Ω with the derivatives of the NURBS numerical solution $-d^2T_r/dx^2$. In NURBS curve/surface fitting referring to [3], the knot vector for the basis should apply the technique of *averaging*, which is generated by the averages of the parameters assigned to the data points.

Therefore, the feature points on the load function $f(x)$ are essential to the knot vector for the left-hand-side fitted function and for the numerical solution T_r. The NURBS basis functions constructing the numerical solution are decided by the knot vector; thus, utilizing the information of feature points on the load function to generate the knot vector is helpful in improving the computational accuracy.

Given the great potential of computational efficiency of IGA-C, this study focuses on the performance of IGA-C framework with our generated knot vector. We propose the IGA-CL method, which combines the representation of the physical domain and the load function characteristics to construct the knot vector for the numerical solution in the IGA-C framework. The main idea is illustrated in Fig. 2. First, uniform parameterization of the physical domain defined by the NURBS geometry is done in the parametric domain to obtain a set of parameters, and the corresponding coordinates in the physical domain are calculated by NURBS mapping. Second, via feature point detection operation of the load function presented in Section 4, we choose the parameters assigned to these feature points, which serve as the knots to insert into the initial knot vector for the physical domain. Finally, on the basis of the integrated knot vector, we construct the NURBS bases for the numerical solution that are different from the initial bases for the physical domain. We employ the

IGA-C framework to obtain the numerical solution to the BVP, where both the basis functions and collocation points depend on our integrated knot vector.

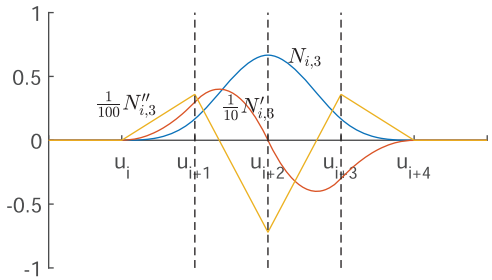


Figure 1: The cubic NURBS basis $N_{i,3}$ and its scaled first-order and scaled second-order derivatives $N'_{i,3}, N''_{i,3}$. The basis functions and their derivatives have the same breakpoints $u_i, u_{i+1}, u_{i+2}, u_{i+3}, u_{i+4}$

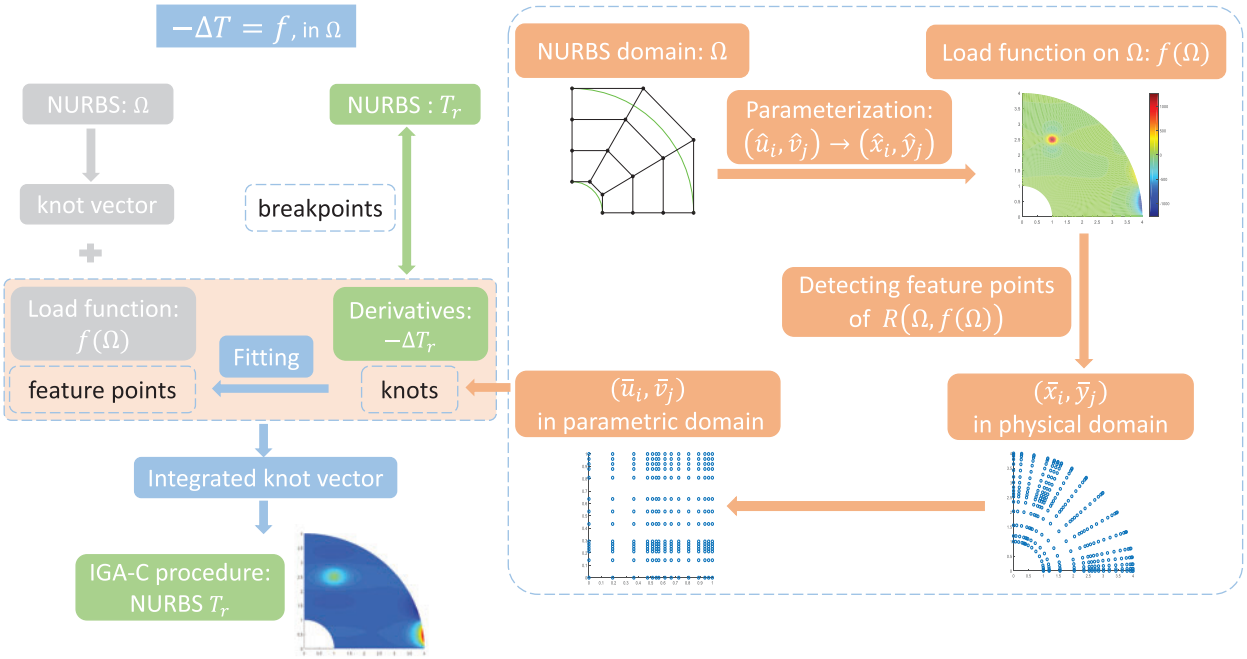


Figure 2: The main idea of IGA-CL: The integrated knot vector combines the initial knot vector of the physical domain Ω and the feature points of the load function $f(\Omega)$. Then, it serves as the knot vector for the NURBS numerical solution T_r. The dotted box on the right presents the detailed flow of the dotted box on the left showing how to obtain the knots

4 Generation of Knot Vectors by Detecting Feature Points of Load Functions

In this section, we introduce the detecting method of feature points from the load functions of the PDEs defined on a physical domain represented by NURBS geometry. We develop a method of knot generation to construct the knot vector once the feature points are detected.

4.1 One-Dimensional Detection

We begin with the detecting method of the load function of the PDE in a 1D case. Comparing with uniform sampling methods using arc length and curvature information ([43,47]), Lu et al. [48] devised a nonuniform point sampling method for fitting the parametric curves with B-spline curves. This method captures the salient features on the curve and produces noticeable fitting results. It begins with the uniform parameterization of the parametric curve; thus, we employ the uniform parameterization to our NURBS-expressed physical domain. Therefore, we obtain the coordinates in the physical domain by NURBS mapping corresponding to the generated parameters and the load function values at these coordinates. Suppose the load function $f(x), x \in \Omega = [a, b]$, where

$$x = \sum_{i'=0}^{N} P_{i'} N_{i',p}(u), u \in [0, 1],$$

and $N_{i',p}$ is NURBS basis defined on the initial knot vector U, then the parametric curve can be written as $\mathbf{R}(u) = (x(u), f(x(u))), u \in \hat{\Omega} = [0, 1]$. *Seed points* are acquired, including endpoints, inflection points, and local extreme curvature points. *Auxiliary points* should be added to generate the detected points until the required number is satisfied, which is realized by a characteristic function, thereby better capturing the feature of the curve.

4.1.1 Seed Points

First, the physical domain $\Omega = [a, b]$ is recognized as an expression of one-to-one NURBS mapping from the parametric domain $\hat{\Omega} = [0, 1]$. We parameterize the domain [0, 1] uniformly to obtain M small intervals and $M + 1$ parameters $\hat{u}_i, i = 0, \ldots, M$. Thus, the corresponding $M + 1$ coordinates $\hat{x}_i$ in domain [*a, b*] are calculated by $\hat{x}_i = \sum_{i'=0}^{N} P_{i'} N_{i',p}(\hat{u}_i)$, where $a = \hat{x}_0 \leq \hat{x}_1 \leq \ldots \leq \hat{x}_M = b$. Then, the point set of the curve at these coordinates is denoted as $\{\mathbf{R}(\hat{u}_i) = (\hat{x}_i, f(\hat{x}_i))\}_{i=0,1,\ldots,M}$. This M is expected to be sufficiently large so that the interior of each small interval has at most one seed point. The curve curvature is denoted as $\hat{k}_i := k(\hat{u}_i)$, defined as

$$k(u) = \frac{\det(\mathbf{R}'(u), \mathbf{R}''(u))}{\|\mathbf{R}'(u)\|^3} = \frac{f''(x)}{(1 + f'^2(x))^{3/2}}. \tag{11}$$

The seed points include endpoints, inflection points, and extreme curvature points, which are determined by referring to [48]. In the present study, we briefly summarize the rules of determination as follows:

Inflection points If $\hat{k}_i = 0$ and $\hat{k}_{i-1}\hat{k}_{i+1} \leq 0$, then point $\mathbf{R}(\hat{u}_i)$ is an inflection point. If $\hat{k}_i \neq 0$ and $\hat{k}_i\hat{k}_{i+1} \leq 0$, then an inflection point exists in the interior of the interval $(\hat{u}_i, \hat{u}_{i+1})$. We denote the parameter of this inflection point as $\overline{u}_i$. Let $\hat{u}_{i+0.5} = \frac{1}{2}(\hat{u}_i + \hat{u}_{i+1})$ and $\hat{k}_{i+0.5} := k(\hat{u}_{i+0.5})$ be the curvature on $\hat{u}_{i+0.5}$. When $\hat{k}_{i+0.5}$ is extremely close to zero, we let $\overline{u}_i = \hat{u}_{i+0.5} = \frac{1}{2}(\hat{u}_i + \hat{u}_{i+1})$. Otherwise, the parabolic function $\hat{k}(u) = a_2u^2 + a_1u + a_0$ is considered to interpolate points $(\hat{u}_i, \hat{k}_i)$, $(\hat{u}_{i+0.5}, \hat{k}_{i+0.5})$, and $(\hat{u}_{i+1}, \hat{k}_{i+1})$. The coefficients of the parabolic function are determined uniquely on the basis of the interpolation conditions. As long as we let $\overline{u}_i \in (\hat{u}_i, \hat{u}_{i+1})$ be the root of function $\hat{k}(u)$, $\mathbf{R}(\overline{u}_i)$ is an approximating inflection point. Then, $\overline{u}_i$ is adopted as a detected point. Note that in our implementation, when $|\hat{k}_i| < 10^{-5}$, $\hat{k}_i$ is taken as 0.

Extreme curvature points If the conditions of $\hat{k}_{i-1} < \hat{k}_i$ and $\hat{k}_{i+1} < \hat{k}_i$ inside the interval $\left(\hat{u}_{i-1}, \hat{u}_{i+1}\right)$ are satisfied, then a local maximum curvature point must lie on it. On the contrary, if $\hat{k}_{i-1} > \hat{k}_i$ and $\hat{k}_{i+1} > \hat{k}_i$, then a local minimum curvature point exists in $\left(\hat{u}_{i-1}, \hat{u}_{i+1}\right)$. $\mathbf{R}\left(\overline{u}_i\right)$ is denoted as the extreme curvature point. We seek $\overline{u}_i$ through parabolic interpolation method, where $\hat{k}\left(u\right) = b_2u^2 + b_1u + b_0$ is used to interpolate points $\left(\hat{u}_{i-1}, \hat{k}_{i-1}\right)$, $\left(\hat{u}_i, \hat{k}_i\right)$, and $\left(\hat{u}_{i+1}, \hat{k}_{i+1}\right)$. Let $\hat{k}'\left(u\right) = 0$, and we obtain the following:

$$\overline{u}_i = -\frac{b_1}{2b_2} = \frac{\left(\hat{u}_i^2 - \hat{u}_{i-1}^2\right)\left(\hat{k}_i - \hat{k}_{i+1}\right) + \left(\hat{u}_{i+1}^2 - \hat{u}_i^2\right)\left(\hat{k}_i - \hat{k}_{i-1}\right)}{2\left(\hat{u}_i - \hat{u}_{i-1}\right)\left(\hat{k}_i - \hat{k}_{i+1}\right) + 2\left(\hat{u}_{i+1} - \hat{u}_i\right)\left(\hat{k}_i - \hat{k}_{i-1}\right)}. \tag{12}$$

Then $\overline{u}_i$ is adopted as a detected point.

The seed point detection is concluded in Algorithm 1. As illustrated, the domain [0, 1] of the parametric curve of the load function $\mathbf{R}(u) = (x(u), f(x(u)))$ is first divided into small intervals, so a set of points in the physical domain [*a, b*] are generated. Given the calculation of the curvatures at these points, the next step is to push the two endpoints into the initial seed point set. Then, we loop all the curvatures to determine the inflection and extreme curvature points according to the schemes above and push them to the current seed point set. Finally, we obtain the corresponding detected point set $\{\overline{u}_i\}_{i=0,\ldots,m}$.

Algorithm 1: Seed point detection algorithm

Require:

The parametric curve of the load function: $\mathbf{R}(u) = (x(u), f(x(u))), x \in [a, b]$;

The NURBS-expressed physical domain: $x = \sum_0^N P_i N_{i,p}(u), u \in [0, 1]$;

Ensure:

Detected point set $\{\overline{u}_i\}_{i=0,\ldots,m}$;

1: Uniform parameterization of the parametric domain [0, 1] : $\left\{\hat{u}_i\right\}_{i=0,\ldots,M}$;
2: Calculate: point set $\left\{\mathbf{R}\left(\hat{u}_i\right)\right\}_{i=0,\ldots,M}$, curvature set $\left\{\hat{k}_i\right\}_{i=0,\ldots,M}$;
3: Initialize: seed point set $\leftarrow \left\{\mathbf{R}\left(\hat{u}_0\right), \mathbf{R}\left(\hat{u}_M\right)\right\}$;
4: **for** $i = 1, 2, \ldots, M-1$ **do**
5: **if** $|\hat{k}_i| < 10^{-5}$ **then**
6: **if** $\hat{k}_{i-1}\hat{k}_{i+1} \leq 0$ **then**
7: Seed point set $\leftarrow$ inflection point $\mathbf{R}\left(\hat{u}_i\right)$;
8: **end if**
9: **else**
10: **if** $\hat{k}_i\hat{k}_{i+1} \leq 0$ **then**
11: Calculate: the root $\overline{u}_i \in \left(\hat{u}_i, \hat{u}_{i+1}\right)$ of $\hat{k} = a_2u^2 + a_1u + a_0$ that interpolates $\left(\hat{u}_i, \hat{k}_i\right)$, $\left(\hat{u}_{i+0.5}, \hat{k}_{i+0.5}\right)$ and $\left(\hat{u}_{i+1}, \hat{k}_{i+1}\right)$;
12: Seed point set $\leftarrow$ inflection point $\mathbf{R}(\overline{u}_i)$;
13: **end if**
14: **end if**

(Continued)

Algorithm 1: (Continued)

```
if (k̂_i > k̂_{i-1} & k̂_i > k̂_{i+1}) || (k̂_i < k̂_{i-1} & k̂_i < k̂_{i+1}) then
    Calculated ū_i by Eq. (12);
    Seed point set ← extreme curvature point R(ū_i);
end if
end for
Obtain the seed point set {R(ū_i)}_{i=0,...,m};
return {ū_i} i = 0, ..., m.
```

4.1.2 Auxiliary Points

The seed point set $\{\mathbf{R}(\overline{u}_i)\}_{i=0,\ldots,m}$ of curve $\mathbf{R}(u)$ is assumed to be already generated, including two endpoints, inflection points, and extreme curvature points. To achieve better detection performance, we need more points to depict the characteristics of the load function.

Define $\lambda(u)$ as the characteristic function of curve $\mathbf{R}(u)$:

$$\lambda(u) = (1-\gamma)\frac{L(u)}{L(1)} + \gamma\frac{K(u)}{K(1)}, \quad u \in [0,1], \tag{13}$$

where γ is the weight, and $L(u)$ and $K(u)$ are the arc length and total curvature from $\mathbf{R}(0)$ to $\mathbf{R}(u)$, respectively.

$$L(u) = \int_0^u \|\mathbf{R}'(s)\| ds,$$

$$K(u) = \int_0^u |k(s)| dL(s) = \int_0^u |k(s)| \|\mathbf{R}'(s)\| ds. \tag{14}$$

Apparently, $\lambda(u)$ is a strictly increasing function, where $\lambda(0) = 0$ and $\lambda(1) = 1$ hold naturally. The integrals Eq. (14) can be numerically computed by discretization. For instance, $L(u) = \sum_{i=0}^{N_L-1} \int_{s_1}^{si+1} ||\mathbf{R}'(s)|| ds$, where $s_i \in \left[0, \frac{u}{N_L}, \frac{2u}{N_L}, \ldots, u\right]$. For this matter, Gauss numerical integration is commonly considered an advanced choice to solve the inner $\int_{s_i}^{s_{i+1}} \|\mathbf{R}'(s)\| ds$ because of the difficulty in computing the integral. The property of the characteristic function depends only on weight γ.

The feature point detection is presented in Algorithm 2. Algorithm 1 returns $m+1$ seed points as the initial feature point set. Then the parametric curve is initially split into m curve segments divided by the acquired seed points. We determine the *salient* curve segment by calculating the difference between the characteristic function values of each segment at the two endpoints, i.e., $\lambda(\overline{u}_{i+1}) - \lambda(\overline{u}_i)$, $i = 0, \ldots, m-1$, to obtain auxiliary detected points. Here, *salient* refers to the curve segment with the largest difference among all curve segments (denoted as j-th curve segment). We seek the $\overline{u}$ that reaches the middle of this salient curve segment, i.e., $\lambda_{mid}(\overline{u}) = \frac{1}{2}\left(\lambda(\overline{u}_j) + \lambda(\overline{u}_{j+1})\right)$. By employing the bisection method through setting a threshold ε, we find the middle sampling point $\overline{u}_{mid}$ such that the characteristic function value at this point is nearest the median characteristic value of the salient curve segment, i.e., $(\lambda(\overline{u}_{mid}) - \lambda_{mid}(\overline{u})) < \varepsilon$. Then the $\overline{u}_{mid}$ is the approximated $\overline{u}$ that we adopt as a detected point. The search for auxiliary points is an iteration process until the required number of detected points is satisfied.

4.2 Two-Dimensional Detection

2D case is regarded as the extension of the 1D case along the $u-$ and $v-$directions. [43] presented the mixed parameterization weighting arc length and curvature to the reconstruction of curves and surfaces similar to [47] rather than the square of the curvature. We exploit the surface sampling of [43] to detect the feature points of the load functions of PDEs here.

A load function $f(x, y)$ with parametric surface is written as $\mathbf{R}(u, v) = (x(u, v), y(u, v), f(x(u, v), y(u, v)))$, $(x, y) \in \Omega$, where Ω is the physical domain represented by NURBS surface

$$S(u, v) = \sum_{i=0}^{N_1} \sum_{j=0}^{N_2} P_{i,j} N_{i,j;p,q}(u, v), u, v \in [0, 1],$$

where $N_{i,j;p,q}(u, v) = \dfrac{B_i^p(u) B_j^q(v) \omega_{ij}}{\sum_{i'=0}^{N_1} \sum_{j'=0}^{N_2} B_{j'}^p(u) B_{j'}^q(v) \omega_{i'j'}}$. Detection is conducted in u-direction and v-direction because NURBS surfaces are tensor product surfaces. In this study, the seed point set of surfaces here only consists of endpoints. We detect feature points on the parametric surface mainly by constructing the characteristic function. The characteristic function contains the terms of area and curvature. We define the area of the parametric surface as below:

$$A = \int_0^1 \int_0^1 \|\mathbf{R}_u(u, v) \times \mathbf{R}_v(u, v)\| du dv. \tag{15}$$

$\mathbf{R}_u(u, v)$ and $\mathbf{R}_v(u, v)$ are partial derivatives in Eq. (15). Thus, the marginal cumulative areas along u- and v-directions are computed as

$$A^u(u) = \int_0^u ds \int_0^1 \|\mathbf{R}_s(s, v) \times \mathbf{R}_v(s, v)\| dv, \quad A^v(v) = \int_0^v ds \int_0^1 \|\mathbf{R}_u(u, s) \times \mathbf{R}_s(u, s)\| du. \tag{16}$$

The mean curvature of the two principle curvatures is good for reflecting the surface shape. Nevertheless, we maintain the curvature computed in each direction without loss of generality. Fix $v = v_0$ and $u = u_0$, respectively, and we obtain the curvatures in u- and v-directions:

$$k^u(u, v_0) = \frac{|f''(u, v_0)|}{(1 + f'^2(u, v_0))^{\frac{3}{2}}}, \quad k^v(u_0, v) = \frac{|f''(u_0, v)|}{(1 + f'^2(u_0, v))^{\frac{3}{2}}}. \tag{17}$$

The curvature of the load function f at point (u_0, v_0) takes the average of the two directional curvatures:

$$k(u_0, v_0) = \frac{k^u(u_0, v_0) + k^v(u_0, v_0)}{2}. \tag{18}$$

Subsequently, we express the marginal cumulative integrals of curvature along the u- and v-directions:

$$k^u(u) = \int_0^u ds \int_0^1 k(s, v) \|\mathbf{R}_s(s, v) \times \mathbf{R}_v(s, v)\| dv,$$

$$k^v(v) = \int_0^v ds \int_0^1 k(u, s) \|\mathbf{R}_u(u, s) \times \mathbf{R}_s(u, s)\| du. \tag{19}$$

Thus, the marginal cumulative characteristic functions along the u- and v-directions are constructed as follows:

$$\lambda^u(u) = (1-\gamma^u)\frac{A^u(u)}{A^u(1)} + \gamma^u\frac{k^u(u)}{k^u(1)}, \quad u \in [0,1],$$

$$\lambda^v(v) = (1-\gamma^v)\frac{A^v(v)}{A^v(1)} + \gamma^v\frac{k^v(v)}{k^v(1)}, \quad v \in [0,1]. \tag{20}$$

Based on $\lambda^u(u)$, we generate the detected points $\{\overline{u}_i\}_{i=0,\ldots,n_1}$ in $u-$direction in accordance with Algorithm 2. Based on $\lambda^v(v)$, we also attain the detected points $\left\{\overline{v}_j\right\}_{j=0,\ldots,n_2}$ in $v-$direction.

Algorithm 2: Feature point detection algorithm

Require:

The parametric curve of the load function: $\mathbf{R}(u) = (x(u), f(x(u))), x \in [a,b]$

The NURBS-expressed physical domain: $x = \sum_0^N P_i N_{i,p}(u), u \in [0,1]$

Seed point set $\{\mathbf{R}(\overline{u}_i)\}_{i=0,\ldots,m}$ by Algorithm 1;

The total number of the feature points: n;

The weight of the characteristic function: γ;

The threshold: ε;

Ensure:

Detected point set $\{\overline{u}_i\}_{i=0,\ldots,n}$;

1: Initialize: feature point set $\leftarrow \{\mathbf{R}(\overline{u}_i)\}_{i=0,\ldots,m}$;

2: **while** $m < n$ **do**

3: Calculate: the differences of characteristic values $\lambda(\overline{u}_{i+1}) - \lambda(\overline{u}_i), i = 0,\ldots,m-1$, and choose the *salient* curve segment j;

4: Calculate: $\lambda_{mid}(\overline{u}) = \frac{1}{2}\left(\lambda\left(\overline{u}_j\right) + \lambda\left(\overline{u}_{j+1}\right)\right)$;

5: Find the approximated $\overline{u}_{mid} \in \left[\overline{u}_j, \overline{u}_{j+1}\right]$ by bisection method to satisfy $(\lambda(\overline{u}_{mid}) - \lambda_{mid}(\overline{u})) < \varepsilon$;

6: Feature point set $\leftarrow$ the auxiliary point $\mathbf{R}(\overline{u}_{mid})$;

7: $m = m + 1$: Adapt the current feature point set $\{\mathbf{R}(\overline{u}_i)\}_{i=0,\ldots,m}$ and ensure $\overline{u}_i$ in an ascending order;

8: **end while**

9: **return** $\{\overline{u}_i\}_{i=0,\ldots,n}$.

4.3 Three-Dimensional Detection

For the 3D case, we construct the characteristic functions in a way similar to that in the 2D case. The load function is $f(x, y, z)$ and the parametric solid is $\boldsymbol{R}(u,v,w) = (x(u,v,w), y(u,v,w), z(u,v,w), f(x,y,z))$, $(x,y,z) \in \Omega$, where Ω is the physical domain defined by

$$S(u,v,w) = \sum_{i=0}^{N_1}\sum_{j=0}^{N_2}\sum_{k=0}^{N_3} P_{i,j,k} N_{i,j,k;p,q,r}(u,v,w), u,v,w \in [0,1],$$

where $N_{i,j,k;p,q,r}(u,v,w) = \dfrac{B_i^p(u)B_j^q(v)B_k^r(w)\omega_{ijk}}{\sum_{i'=0}^{N_1}\sum_{j'=0}^{N_2}\sum_{k'=0}^{N_3} B_{i'}^p(u)B_{j'}^q(v)B_{k'}^r(w)\omega_{i'j'k'}}$. Detected points are computed from u-, v-, and w-directions.

The volume of the parametric solid is presented as below:

$$V = \int_0^1 \int_0^1 \int_0^1 |(\mathbf{R}_u \times \mathbf{R}_v) \cdot \mathbf{R}_w| dudvdw. \tag{21}$$

Thus, the marginal cumulative volumes in three directions can be described as

$$V^u(u) = \int_0^u ds \int_0^1 \int_0^1 |(\mathbf{R}_s \times \mathbf{R}_v) \cdot \mathbf{R}_w| dvdw,$$

$$V^v(v) = \int_0^v ds \int_0^1 \int_0^1 |(\mathbf{R}_u \times \mathbf{R}_s) \cdot \mathbf{R}_w| dudw,$$

$$V^w(w) = \int_0^w ds \int_0^1 \int_0^1 |(\mathbf{R}_u \times \mathbf{R}_v) \cdot \mathbf{R}_s| dudv. \tag{22}$$

On the other hand, the curvatures in u-, v-, and w-directions are

$$k^u(u, v_0, w_0) = \frac{|f''(u, v_0, w_0)|}{\left(1 + f'(u, v_0, w_0)^2\right)^{\frac{3}{2}}},$$

$$k^v(u_0, v, w_0) = \frac{|f''(u_0, v, w_0)|}{\left(1 + f'(u_0, v, w_0)^2\right)^{\frac{3}{2}}},$$

$$k^w(u_0, v_0, w) = \frac{|f''(u_0, v_0, w)|}{\left(1 + f'(u_0, v_0, w)^2\right)^{\frac{3}{2}}}. \tag{23}$$

Then, the curvature of the load function f at point (u_0, v_0, w_0) is

$$k(u_0, v_0, w_0) = \frac{k^u(u_0, v_0, w_0) + k^v(u_0, v_0, w_0) + k^w(u_0, v_0, w_0)}{3}. \tag{24}$$

The marginal cumulative integrals of the curvature in u-, v-, and w-directions are

$$k^u(u) = \int_0^u ds \int_0^1 \int_0^1 k(s, v, w) |(\mathbf{R}_s \times \mathbf{R}_v) \cdot \mathbf{R}_w| dvdw, \tag{25}$$

$$k^v(v) = \int_0^v ds \int_0^1 \int_0^1 k(u, s, w) |(\mathbf{R}_u \times \mathbf{R}_s) \cdot \mathbf{R}_w| dudw,$$

$$k^w(w) = \int_0^w ds \int_0^1 \int_0^1 k(u, v, s) |(\mathbf{R}_u \times \mathbf{R}_v) \cdot \mathbf{R}_s| dudv.$$

Therefore, the marginal cumulative characteristic functions of the load function f in three directions are constructed as follows:

$$\lambda^u(u) = (1 - \gamma^u) \frac{V^u(u)}{V^u(1)} + \gamma^u \frac{k^u(u)}{k^u(1)}, u \in [0, 1], \tag{26}$$

$$\lambda^v(v) = (1 - \gamma^v) \frac{V^v(v)}{V^v(1)} + \gamma^v \frac{k^v(v)}{k^v(1)}, v \in [0, 1],$$

$$\lambda^w(w) = (1 - \gamma^w) \frac{V^w(w)}{V^w(1)} + \gamma^w \frac{k^w(w)}{k^w(1)}, w \in [0, 1].$$

Given that the process is similar to surface detection, we omit the details here.

4.4 Generation of Knot Vector

Many numerical approaches were proposed to solve PDEs, among which IGA-C is an efficient method, which is described in Section 2.2. However, the knot vector of the IGA-C method can influence not only the NURBS bases for the numerical solution but also the collocation points.

Given the 1D scenario, the other two cases are treated similarly. Since the point detection is done, we obtain the feature point set $\{\mathbf{R}(\overline{u}_i)\}_{i=0,\ldots,n}$ and the corresponding parameter set $\{\overline{u}_i\}_{i=0,\ldots,n}$. Determining the collocation points is vital to solve a PDE in IGA-C. To make good use of feature points that depict the shape and characteristics of the load function, the parameter set $\{\overline{u}_i\}_{i=0,\ldots,n}$ of the feature points will be served as knots to be inserted into the knot vector U for the NURBS bases of the physical domain. Then, the knot vector U^* for the NURBS bases of the solution field that we constructed through insertion can be deemed as the derivation of the initial knot vector U of Ω. Hence, assuming the initial open knot vector of the physical domain represented by NURBS is

$$U = \{u_0, u_1, \ldots, u_{N+p+1}\}, \tag{27}$$

we generate additional knots by *averaging* operation ([3]) based on Schoenberg-Whitney theorem ([49]) from the feature points for each direction in curve/surface fitting:

$$u^*_{j+p} = \frac{1}{p}\sum_{i=j}^{j+p-1}\overline{u}_i, \quad j = 1, 2, \ldots, n-p. \tag{28}$$

Moreover, Hosseini et al. summarized three knot placement techniques for geometric construction from the data points to decide the positions of the internal knots of the knot vector ([50]), namely, uniform knot placement, De Boor's algorithm, and Piegl and Tiller's algorithm. Here, the generated knots representing the characteristics of the load function are inserted into the knot vector U. The knots $\{u^*_{j+p}\}_{j=1,\ldots,n-p}$ that are extremely close to the internal knots $\{u_i\}_{i=p+1,\ldots,N}$ in U should be abandoned, because the multiplicity of the knot leads to unnecessary decrease of the continuity here, and the integrated knot vector should be sorted again. These processes are needed to ensure the nondecreasing knot vector and maintain the continuity of the splines. We denote the nondecreasing integrated knot vector as

$$U^* = \{u_0, u_1, \ldots, u_{N^*+p+1}\}. \tag{29}$$

The spline numerical solution is then expressed as $T_r(u) = \sum_0^N t_i N^*_{i,p}(u)$ to connect it with IGA-C and solve the PDE.

Employing collocation schemes such as Greville abscissae, superconvergent points, and Cauchy-Galerkin points yields a set of collocation points. Replacing the unknown function $T(u)$ in the PDE with the numerical solution $T_r(u)$ at collocation points leads to a linear system. Such a knot vector constructed for numerical solution combines the information of the physical domain and the load function. We name it as *IGA-CL*. It reflects how feature points are distributed, and distinguished from the knot vector grabbed directly from the knot vector of geometry expression in typical IGA-C. We only need to calculate the knot vectors U^* and V^* from u- and v-directions for the 2D cases and the knot vector W^* from the additional w-direction for the 3D cases.

5 Experimental Results

In this section, we validate the influence of computing accuracy with our isogeometric analysis by fitting load functions in solving PDEs through numerical experiments. The typical IGA-C and our

IGA-CL are compared in Sections 5.1–5.3. We verify that the IGA-CL method outperforms IGA-C by contrasting relative errors and absolute errors, particularly for the case when the solution has obvious characteristics. In addition, we extend to exploit the IGA-GL method, which displays the same superiority of the accuracy over the typical IGA-G method through the comparisons in Section 5.4. For the following examples, relative error e_r is used, which is defined as

$$e_r = \sqrt{\frac{\int_\Omega (T - T_r)^t (T - T_r) d\Omega}{\int_\Omega T^t T d\Omega}}, \tag{30}$$

where t stands for the transpose, the analytical solution and numerical solution are denoted as T and T_r, respectively. When the solution is a scalar-valued function, $e_r = \sqrt{\frac{\int_\Omega (T - T_r)^2 d\Omega}{\int_\Omega T^2 d\Omega}}$.

5.1 One-Dimensional PDE

Example I: A 1D Poisson problem with Dirichlet boundary condition:

$$\begin{cases} -\frac{d^2T}{dx^2} = f_1(x) & in \quad \Omega = [0, 2], \\ T|_{\partial\Omega(x)} = T_1(x) & on \quad \partial\Omega, \end{cases} \tag{31}$$

where

$$f_1(x) = \frac{1}{3500}\left(\frac{2}{((x - 0.3)^2 + 0.02)^2} - \frac{2(2x - 0.6)^2}{((x - 0.3)^2 + 0.02)^3}\right)$$

and the analytical solution is

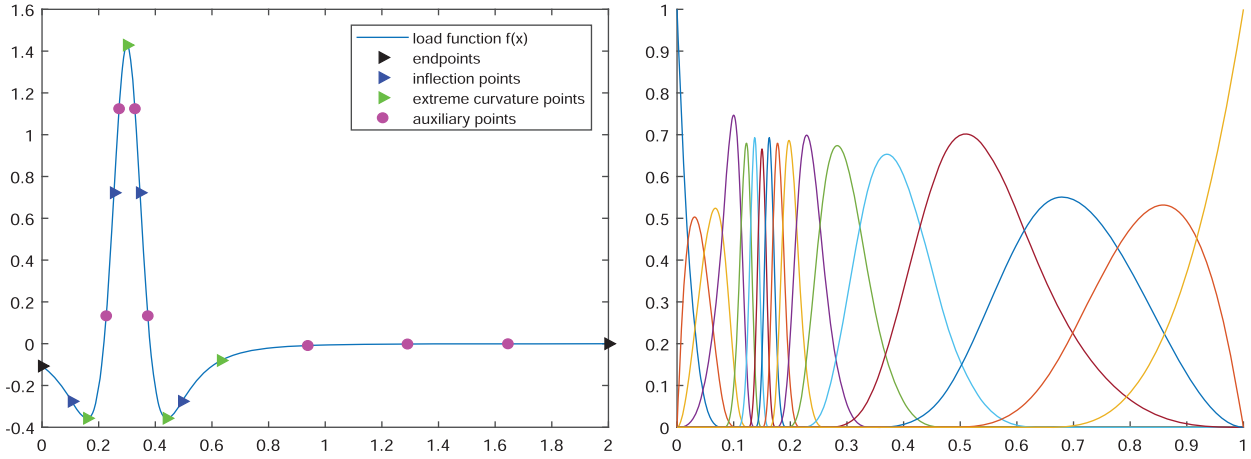
$$T_1(x) = \frac{1}{3500((x - 0.3)^2 + 0.02)}.$$

Given the advantage of our IGA-CL method, we utilize the feature points of the load function and construct an integrated knot vector. To verify the accuracy improvement, we apply our IGA-CL framework to solve Eq. (31) and compare it with the typical IGA-C at Greville abscissae ([4]) and other collocation schemes such as SC points ([9]) of the same degrees of freedom.

Fig. 3a displays the detecting points of the load function $f_1(x)$ according to the point detection. The weight in the characteristic function is set to 0.1, and the number of auxiliary points is set to 7. The knot vector is then constructed with our knot generation method. The NUBRS bases are calculated iteratively based on the knot vector. Fig. 3b shows an instance of cubic NUBRS bases from our integrated knot vector. This instance of cubic NURBS bases is distinguished from the NURBS bases generated only using the knot vector of the physical domain. The distribution of these bases appears dense in the area where the load function is steep; otherwise, the distribution of these bases becomes sparse.

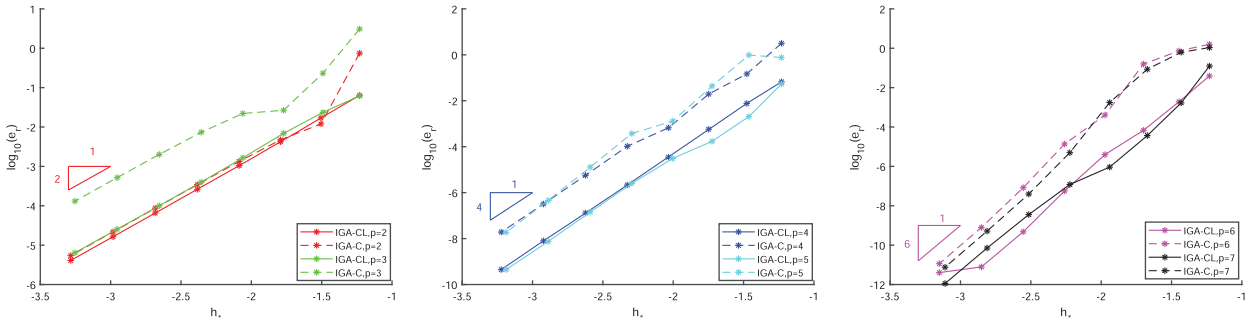
Fig. 4 shows the comparisons of convergence of the relative error by exploiting our IGA-CL framework and the typical IGA-C framework at Greville abscissae with the same degrees of freedom under different degrees of the bases. The convergence of solving this 1D problem by IGA-CL with the integrated knot vector is expressed by solid lines, whereas the convergence by typical IGA-C is displayed by dashed lines. We denote the abscissa as $h_* = \log_{10}(1/dofs)$, where *dofs* is the abbreviation of degrees of freedom, which are obtained by h-refinements that enrich the basis without changing the physical domain geometrically or parametrically. In our implementation, if the original knot vector is $U^* = \{u_0, u_1, \cdots, u_{N^*+p+1}\}$ where $u_0 = u_1 = \cdots = u_p = 0$ and

$u_{N^*+1} = u_{N^*+2} = \cdots = u_{N^*+p+1} = 1$, after once h-refinement, then we obtain the refined knot vector $U_1^* = \left\{u_0, u_1, \ldots, u_p, \frac{1}{2}\left(u_p + u_{p+1}\right), u_{p+1}, \frac{1}{2}\left(u_{p+1} + u_{p+2}\right), u_{p+2}, \ldots, u_{N*}, \frac{1}{2}\left(u_{N*} + u_{N^*+1}\right), u_{N^*+1}, u_{N^*+2}, \ldots, u_{N^*+p+1}\right\}$. In other words, we refer to each "h-refinement" as inserting the midpoint of each nonzero knot interval into the current knot vector each time.



(a) Feature points of f_1 (b) Cubic NURBS bases

Figure 3: Our detected points on the load function and the corresponding cubic NURBS bases with knot generation method



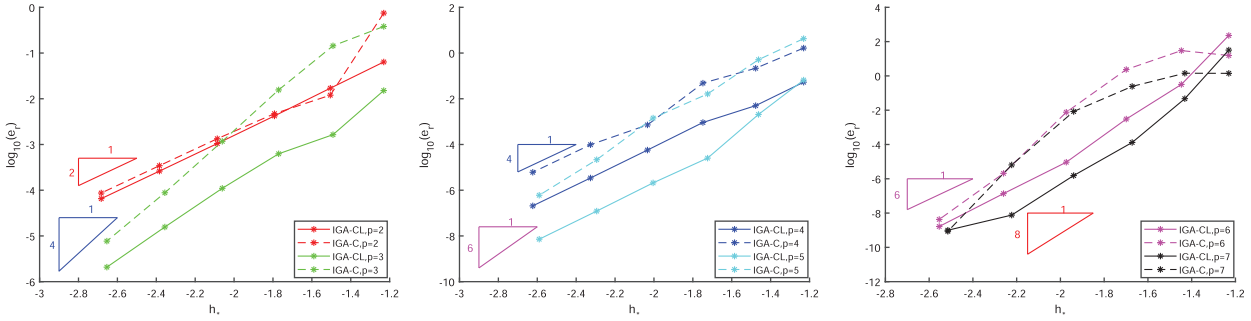
(a) p=2,3 (b) p=4,5 (c) p=6,7

Figure 4: Convergence comparison of IGA-CL/IGA-C at Greville collocation points under different degrees of Example I

Let the ordinate be the $\log_{10}(e_r)$. We describe the convergence rates with the help of triangles, on which the numbers stand for the convergence rates. Besides Greville abscissae, SC collocation points are also applied to IGA-CL in Fig. 5. Notice that the number of SC collocation points is almost twice the degrees of freedom, we solve it in the way of least-squares.

Our method demonstrates its superiority over typical IGA-C in terms of accuracy under all the degrees p of the NURBS basis functions. Because of the knot selection from the load function, the relative errors of small *dofs* at the beginning of IGA-CL are much lower than those of the typical IGA-C that uses the knot vector only from the physical domain. The general convergence rates of

IGA-CL remain unchanged or slightly lower than those of IGA-C. For Greville collocation points, the convergence orders of IGA-C and IGA-CL are both $\mathcal{O}(h^p)$ for even p and $\mathcal{O}(h^{p-1})$ for odd p. For SC collocation points, the convergence orders of IGA-CL and IGA-C are both $\mathcal{O}(h^p)$ for even p; however, the convergence order of IGA-CL is slightly lower than $\mathcal{O}(h^{p+1})$ for a high odd p. Given the desirable accuracy, IGA-CL still outperforms the IGA-C.



(a) p=2,3 (b) p=4,5 (c) p=6,7

Figure 5: Convergence comparison of IGA-CL/IGA-C at SC points under different degrees of Example I

For PDEs under other boundary conditions, we present the following example:

Example II: A 1D ODE with Dirichlet and Neumann boundary conditions:

$$\begin{cases} -\dfrac{d^2T}{dx^2} + T = f_2(x) \quad in \quad \Omega = [0,1], \\ T(0) = T_2(0), \\ T'(1) = T_2'(1). \end{cases} \tag{32}$$

where

$$f_2(x) = \left(101 - 10000\,(x-0.3)^2\right) e^{-\frac{(x-0.3)^2}{0.02}}$$

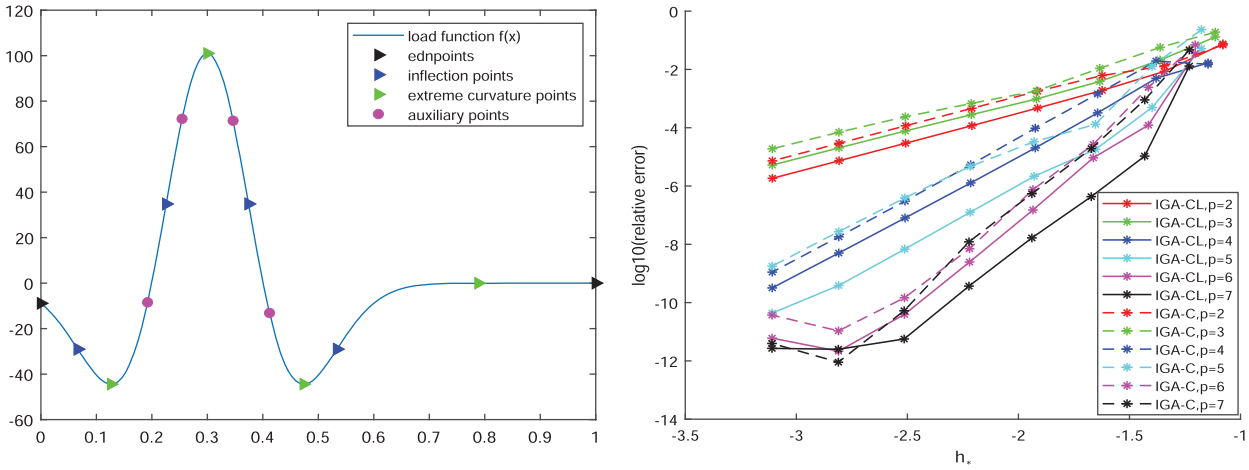
and the analytical solution is

$$T_2(x) = e^{-\frac{(x-0.3)^2}{0.02}}.$$

In this case, we set the parameter γ as 0.1 and the number of auxiliary points as 4. The detection as well as the convergence of relative error at Greville are displayed in Fig. 6.

5.2 Two-Dimensional PDE

We compare IGA-CL with IGA-C at Greville abscissae and SC points in solving the following two 2D Examples III and IV. The main idea of the IGA-CL is detecting the feature points on the load function and then linking them to IGA-C for solving PDEs. Whether the accuracy of the IGA-CL solution outperforms IGA-C principally depends on the quality of the feature points detected. The weights in the characteristic functions can be controlled and adjusted, thereby influencing the detection effect. We show the convergence of examples with a relatively satisfying choice of weight.

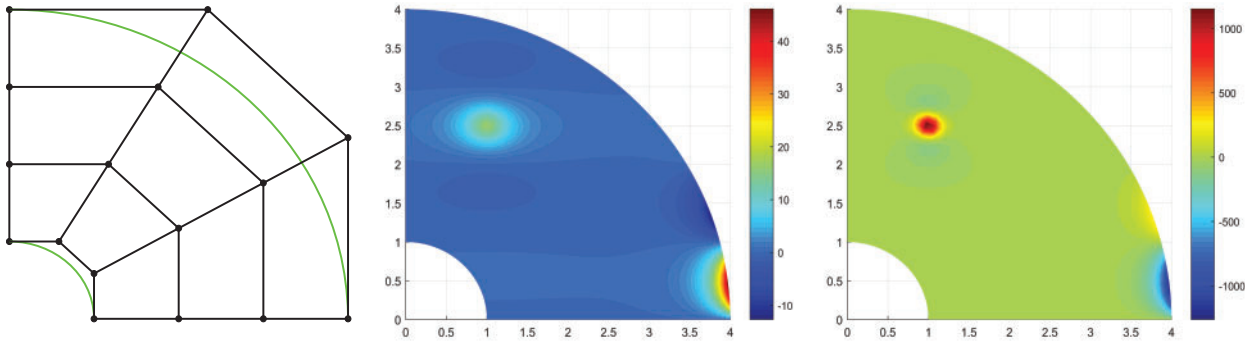


(a) Feature points of f_2 (b) Convergence of relative error

Figure 6: Detected points on the load function and the convergence results of relative error

Example III: A 2D Poisson problem on a quarter of an annulus Ω (Fig. 7a) with Dirichlet boundary condition is given as follows:

$$\begin{aligned} -\Delta T &= f_3(x,y) \quad in \quad \Omega, \\ T|_{\partial\Omega(x,y)} &= T_3(x,y) \quad on \quad \partial\Omega. \end{aligned} \tag{33}$$



(a) Control mesh (b) Analytical solution T_3 (c) Load function f_3

Figure 7: Example III. The analytical solution and the load function have nearly the same characteristics

The load function is

$$\begin{aligned} f_3(x,y) = &\frac{\pi^2 \sin(\pi y)}{A} + \frac{4\sin(\pi y) + 2\pi\cos(\pi y)(2y-5)}{A^2} \\ &- \frac{2\sin(\pi y)((2x-2)^2 + (2y-5)^2)}{A^3} \\ &+ (\pi^2 - 2 - (2x-2)^2)\sin(\pi y) e^{(x-1)^2-5} \end{aligned}$$

where $A = (x-1)^2 + (y-2.5)^2 + 0.06$ such that the analytical solution is

$$T_3(x,y) = \left(e^{(x-1)^2-5} + \frac{1}{A}\right)\sin(\pi y).$$

Example IV: An elliptic PDE on a quarter of a plate with a circular hole Ω (Fig. 8a) with Dirichlet condition is given as follows:

$$\begin{cases} -\Delta T + T = f_4(x,y) & in \quad \Omega, \\ T|_{\partial\Omega(x,y)} = T_4(x,y) & on\ \partial\Omega. \end{cases} \tag{34}$$

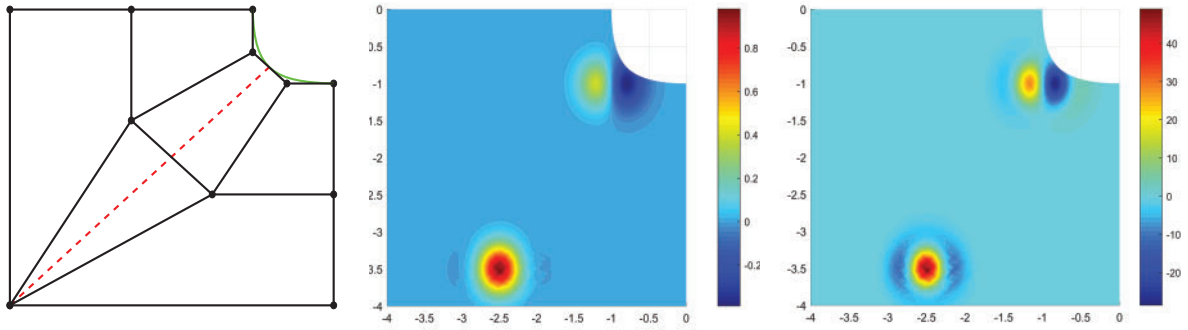
The load function is

$$\begin{aligned} f_4(x,y) =& B[\sin(\pi x)(\pi^2 + 41 - (20x+20)^2 - (20y+20)^2) + 2\pi\cos(\pi x)(20x+20)] \\ &+ C[\sin(\pi x)(-41 - \pi^2 + (20x+50)^2 + (20y+70)^2) - 2\pi\cos(\pi x)(20x+50)] \end{aligned}$$

where $B = e^{-10(x+1)^2-10(y+1)^2}$, $C = e^{-10(x+2.5)^2-10(y+3.5)^2}$ and

$T_4(x,y) = (B-C)\sin(\pi x)$.

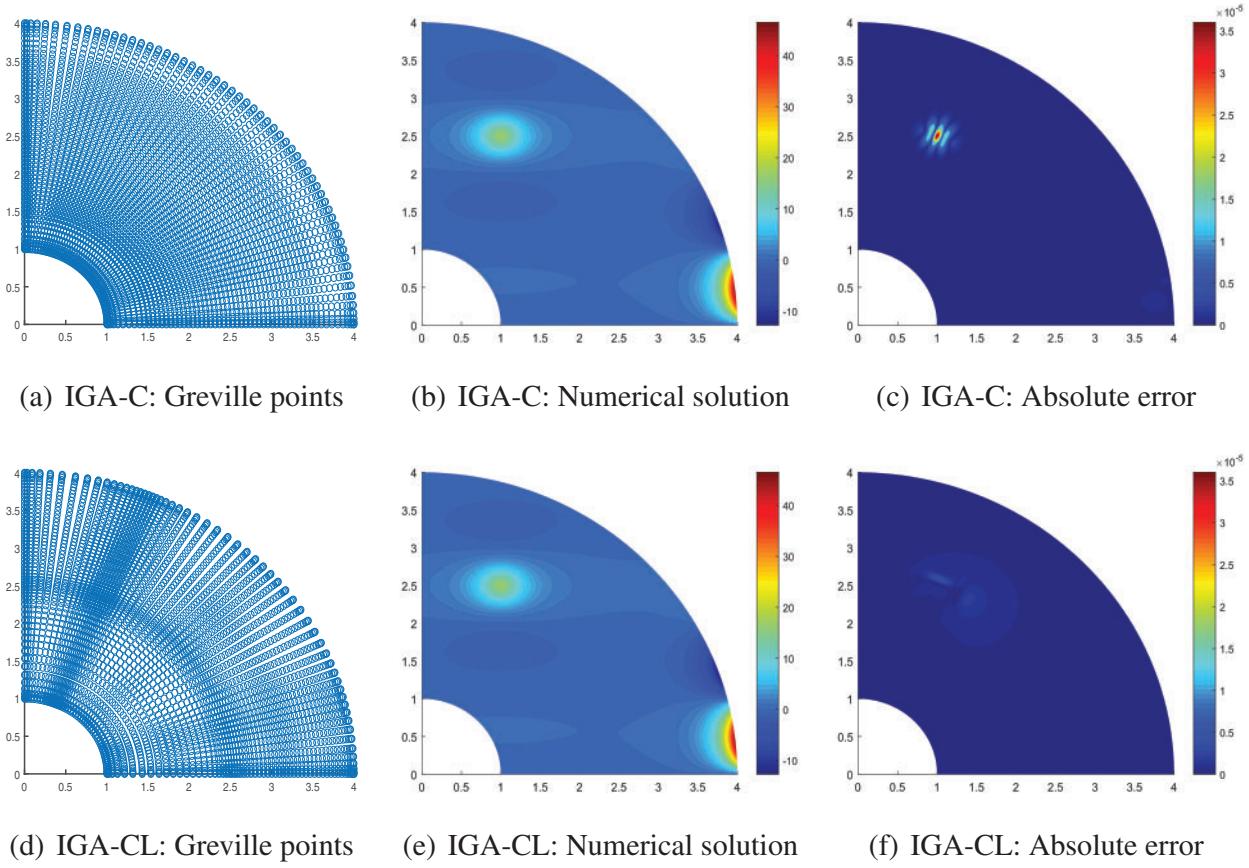
To begin with, we present Figs. 7 and 8 displaying the analytical solutions and the load functions to demonstrate the effectiveness of our method. For the convenience of comparing errors, the analytical solutions for all experiments in this study are assumed to be known. In fact, in "real settings", the "exact" solutions could be computed with a dense mesh. It is observed that the load functions roughly retain the characteristics of the analytical solutions of the PDEs. Consequently, detecting the feature points of the load function is a helpful guidance when solving PDEs.



(a) Control mesh (b) Analytical solution T_4 (c) Load function f_4

Figure 8: Example IV. The analytical solution and the load function have nearly the same characteristics

In order to solve the elliptic equations Examples III and IV, the IGA-CL and IGA-C methods at Greville and SC collocation points are employed and compared in terms of absolute error. The initial number of detected points is 15×15 and 17×15 for Examples III and IV, respectively. Fig. 9 displays the comparison of IGA-C solution and IGA-CL solution of Example III, while Fig. 10 displays the comparison of Example IV. Both figures are shown under the degrees of $p = q = 6$ at Greville. The selection of the weight γ in the characteristic function is vital for detecting. Based on various experiments, if the load function has a curvature distribution that is mostly the same or varies slowly, a large weight γ is suggested; otherwise, a small weight γ is suggested.



(a) IGA-C: Greville points (b) IGA-C: Numerical solution (c) IGA-C: Absolute error

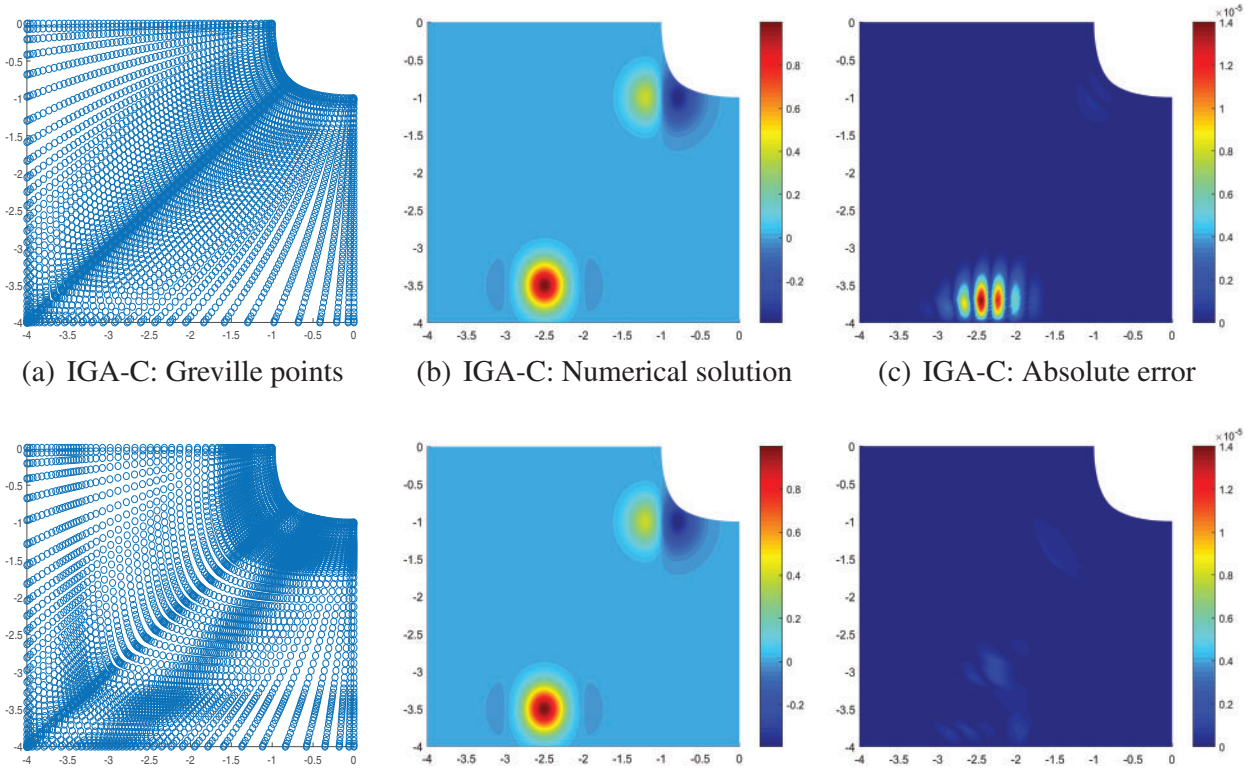
(d) IGA-CL: Greville points (e) IGA-CL: Numerical solution (f) IGA-CL: Absolute error

Figure 9: Comparison of IGA-CL/IGA-C solution of Example III after three times of h-refinement

However, in these two cases, considering the extreme nonuniformity after h-refinements when a small weight is adopted, we take a large weight in practical experiments to ensure that the flat part of the shape of load function has fairly and evenly detected points. For simplicity, the weights we adopted here are $\gamma^u = \gamma^v = 0.8$ in the two directions for Example III and $\gamma^u = \gamma^v = 0.95$ for both directions for Example IV.

Figs. 9d and 10d show that the distribution of the collocation points in the physical domain is very related to the solution. This finding indicates that the initial number of the detected points adopted in the area where the load function reaches the local maximum or the minimum is more than the number of detected points in the area where the load function appears steady. Through the comparisons of absolute error between IGA-C and IGA-CL methods, our IGA-CL method has a much lower error than IGA-C method.

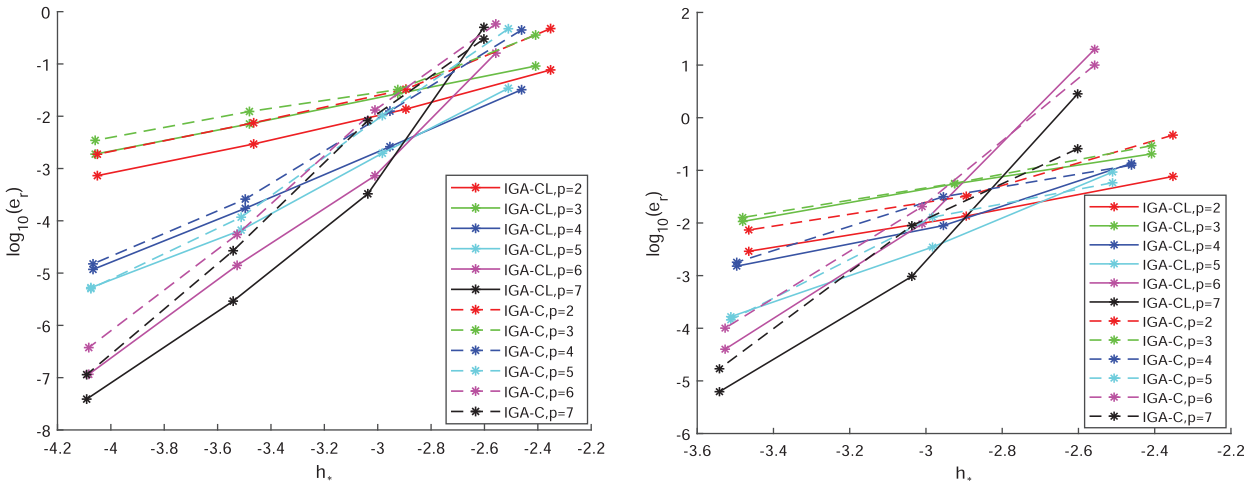
With the appropriate weights, we exhibit the convergence comparisons of the relative error of IGA-CL and IGA-C at Greville abscissae and SC points under different degrees of NURBS bases of the numerical solution of Examples III and IV in Figs. 11 and 12, respectively. Our IGA-CL outperforms IGA-C distinctly under the same *dofs*. The convergence orders of the IGA-CL solution appear a little lower than those of IGA-C because the difference between the size of knot intervals of IGA-C and IGA-CL diminishes as the *dofs* increase. However, IGA-CL solutions still show an obvious increase in accuracy, which is conspicuously considerable.



(a) IGA-C: Greville points (b) IGA-C: Numerical solution (c) IGA-C: Absolute error

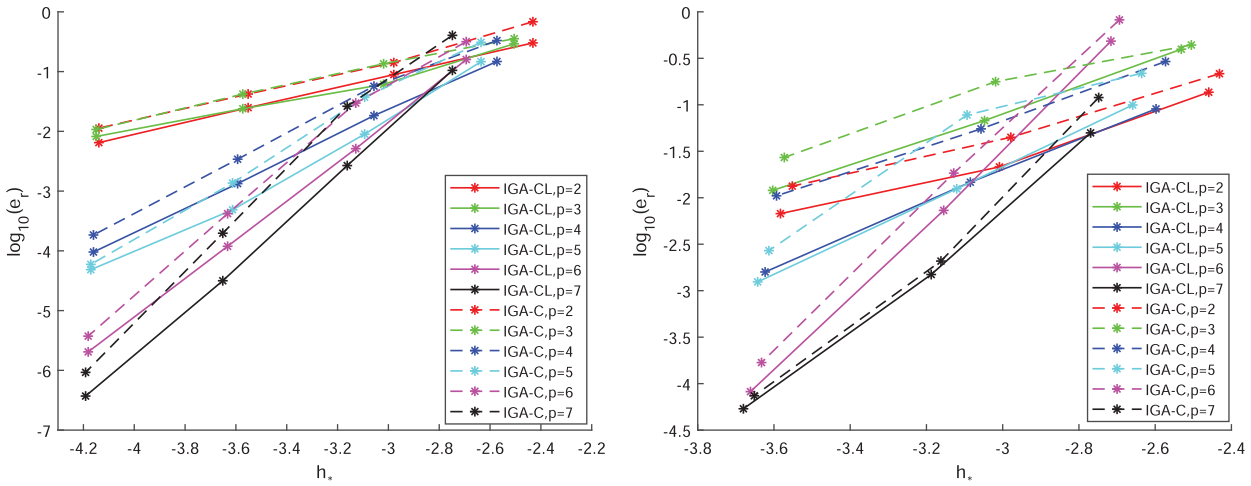
(d) IGA-CL: Greville points (e) IGA-CL: Numerical solution (f) IGA-CL: Absolute error

Figure 10: Comparison of IGA-CL/IGA-C solution of Example IV after three times of h-refinement



(a) Greville points (b) SC points

Figure 11: Convergence comparison of IGA-CL/IGA-C at Greville/SC collocation under different degrees of Example III



(a) Greville points (b) SC points

Figure 12: Convergence comparison of IGA-CL/IGA-C at Greville/SC collocation under different degrees of Example IV

5.3 Three-Dimensional PDE

Example V: A 3D elliptic PDE, on a unit cube with Dirichlet boundary condition is given as follows:

$$\begin{cases} -\Delta T + T = f_5(x,y,z) & in \quad \Omega, \\ T|_{\partial\Omega(x,y,z)} = T_5(x,y,z) & on \quad \partial\Omega, \end{cases} \tag{35}$$

where

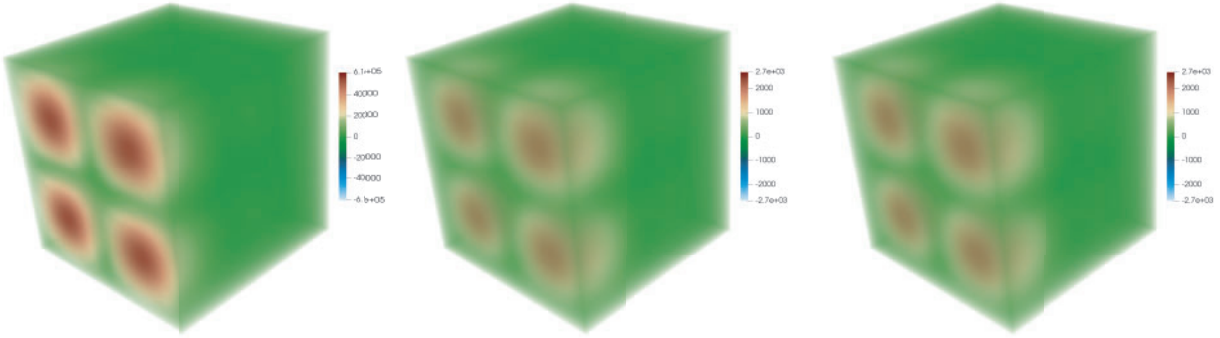
$$f_5(x,y,z) = (12\pi^2 - 4(x+2)^2 - 1)e^{(x+2)^2}\sin(2\pi x)\sin(2\pi y)\sin(2\pi z) - 4\pi(2x+4)e^{(x+2)^2}\cos(2\pi x)\sin(2\pi y)\sin(2\pi z),$$

and the analytical solution is

$$T_5(x,y,z) = e^{(x+2)^2}\sin(2\pi x)\sin(2\pi y)\sin(2\pi z).$$

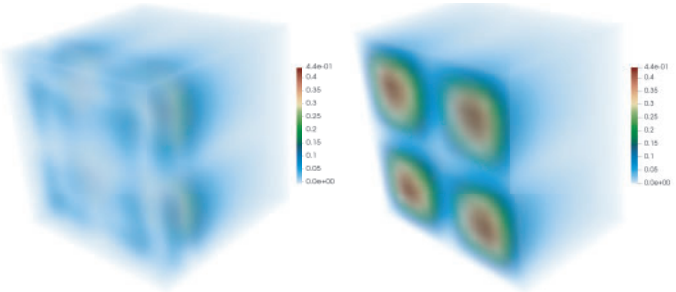
In this case, the initial number of feature points that we sampled in our IGA-CL method is 6 for each direction, and the weights in characteristic functions are $\gamma^u = \gamma^v = \gamma^w = 0.1$. Fig. 13c demonstrates the numerical solution of our IGA-CL method for solving Example V after two times of h-refinements when the degrees are $p = q = r = 4$. These figures are visualized on ParaView 5.9.0-RC3.

Then, we compare the distribution of absolute errors between IGA-CL and IGA-C at Greville abscissae with the same *dofs* at an arbitrary degree of the numerical solution after two times of h-refinements, as shown in Fig. 14. The absolute errors of IGA-CL and IGA-C methods are shown when $p = q = r = 4$, with the same *dofs* of $20 \times 20 \times 20$. The maximum absolute error of IGA-C solving the PDE is 0.4390, and the relative error is 1.3465×10^{-4}. However, the maximum absolute error of our IGA-CL method is 0.0775, and the relative error is 2.1933×10^{-5}.

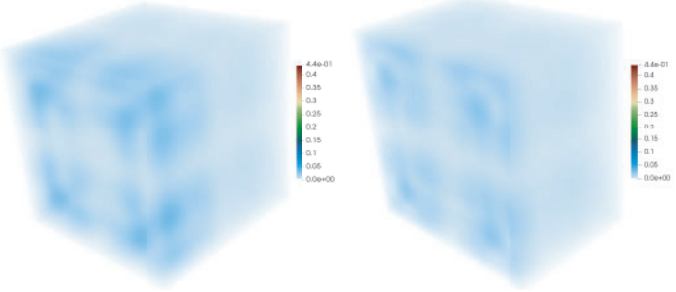


(a) Load function f_5 (b) Analytical solution T_5 (c) Numerical IGA-CL solution

Figure 13: IGA-CL solution of Example V after two times of h-refinements



(a) Absolute error: IGA-C (b) Clip: IGA-C



(c) Absolute error: IGA-CL (d) Clip: IGA-CL

Figure 14: The absolute errors of IGA-CL/IGA-C at Greville abscissae of Example V. The second column is the clip view of the first column. The maximum absolute errors of IGA-C and IGA-CL methods are 0.4390 and 0.0775, respectively

The convergence figures of relative errors are also shown. Fig. 15 presents the comparison of the convergence of relative error between IGA-CL and IGA-C under different degrees of the bases of the numerical solution. Except that the convergence curves of the two methods coincide when the degrees are $p = q = r = 5$ at Greville points, we observe that our method is superior to the IGA-C method.

One concern that should be addressed is that the detection of the feature points as a pre-processing costs time. Here we show the comparison of the time cost and relative error between IGA-CL and IGA-C in Table 1. The detection of feature points includes two parts: the calculation of curvatures of the

sampling points, and the detection using characteristic function. The computation realizes the IGA framework to solve PDEs. We implemented them with MATLAB on an 8G laptop. From the table, we observe that the computation time of IGA-CL and IGA-C is almost the same due to the same framework, while IGA-CL has additional time cost of pre-processing on detection of feature points. Although there is a little extra time for feature point detection in IGA-CL, the precision of IGA-CL is higher than that of IGA-C.

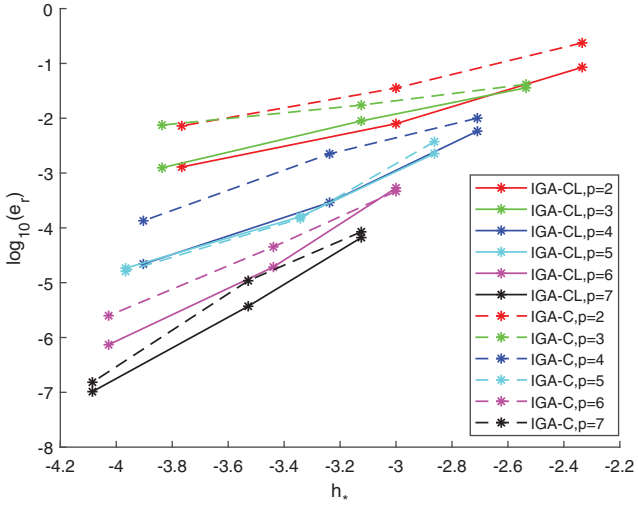


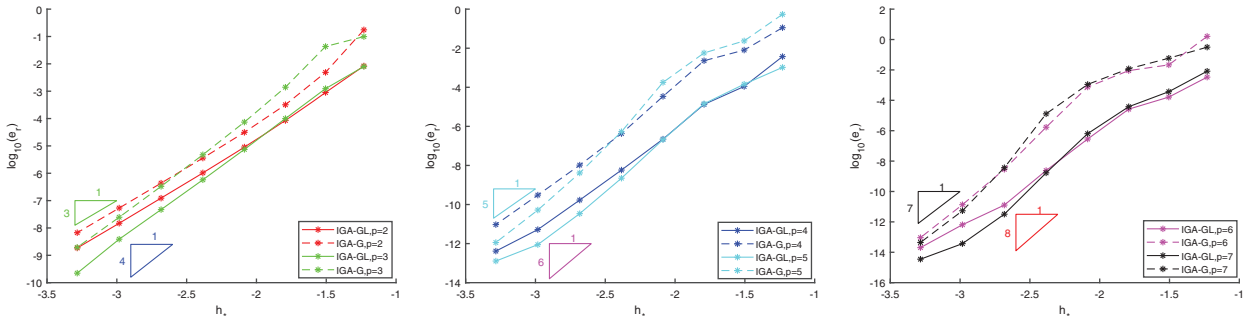
Figure 15: Convergence comparison of IGA-CL/IGA-C at Greville collocation under different degrees of Example V

Table 1: Comparison of time cost of IGA-CL and IGA-C (s)

Examples	DoFs	IGA-CL			IGA-C	
		Detection	Computation	Relative error	Computation	Relative error
1D: Example I	7684	20.099	21.992	9.5477e-11	22.603	1.8522e-10
1D: Example II	8197	6.704	25.399	7.4817e-10	24.973	3.6400e-10
2D: Example III	12100	35.856	150.594	1.1580e-7	151.332	3.7566e-7
2D: Example IV	14161	36.466	293.799	3.7136e-7	292.682	9.2534e-6
3D: Example V	12167	65.214	632.034	8.1771e-8	633.725	1.5276e-7

5.4 Example for IGA-GL

Based on the knot selection scheme, IGA concentrating on load function is applicable not only to the IGA-C method, but also to the IGA-G method. For Example I in Section 5.1, Fig. 16 also shows the superiority of our knot generation by fitting the load function in the IGA-G method over the typical IGA-G method. Since we employ the integrated knot vector to the IGA-G, we call the improved method as *IGA-GL*. From the initial degrees of freedom, our IGA-GL is more accurate than the traditional IGA-G by approximately 1 to 2 orders of magnitude level. The convergence orders are both the optimal $\mathcal{O}(h^{p+1})$ for all degrees p.



(a) p = 2,3 (b) p = 4,5 (c) p = 6,7

Figure 16: Convergence comparison between IGA-GL/IGA-G under different degrees of Example I

6 Concluding Remarks

This study proposes a new perspective on solving IGA problems, which treats the left-hand side and right-hand side of the PDE as fitting the load function with the derivatives of the unknown NURBS-expressed solution. We present a detailed introduction to how to detect feature points on the load function of the PDE in 1D, 2D, and 3D cases. Then, we propose the IGA-CL method in which the knot vector of the NURBS numerical solution combines the characteristics of the load function with the representation of the physical domain. We validate that utilizing the feature points of the load function to construct the integrated knot vector helps improve the accuracy while generally maintaining the convergence rate. The theoretical result of the convergence behavior is beyond the scope of this study. Experimental results demonstrate that the accuracies of both our IGA-CL framework and IGA-GL framework are higher than those of the typical IGA-C and IGA-G methods, particularly in solving problems with analytical solutions with obvious characteristics.

Funding Statement: This work is supported by the National Natural Science Foundation of China under Grant Nos. 61872316, 62272406, 61932018, and the National Key R&D Plan of China under Grant No. 2020YFB1708900.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Hughes, T. J. R., Cottrell, J. A., Bazilevs, Y. (2005). Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics & Engineering, 194(39),* 4135–4195.
2. Zienkiewicz, O. C., Taylor, R. L. (2013). *The finite element method: Its basis and fundamentals*. Oxford: Butterworth-Heinemann.
3. Piegl, L., Tiller, W. (1997). *The NURBS book*. Berlin Heidelberg: Springer.
4. Auricchio, F., Beirão da Veiga, L., Hughes, T. J. R., Reali, A., Sangalli, G. (2010). Isogeometric collocation methods. *Mathematical Models & Methods in Applied Sciences, 20(11),* 2075–2107. https://doi.org/10.1142/S0218202510004878
5. Folland, G. B. (1995). *Introduction to partial differential equations*, vol. 102. Princeton, New Jersey: Princeton University Press.

6. Jupp, D. L. B. (1978). Approximation to data by splines with free knots. *Siam Journal on Numerical Analysis, 15(2),* 328–343. https://doi.org/10.1137/0715022
7. He, X., Shen, L., Shen, Z. (2002). A Data-adaptive knot selection scheme for fitting splines. *IEEE Signal Processing Letters, 8(5),* 137–139.
8. Lin, H., Hu, Q., Xiong, Y. (2013). Consistency and convergence properties of the isogeometric collocation method. *Computer Methods in Applied Mechanics & Engineering, 267(8),* 471–486. https://doi.org/10.1016/j.cma.2013.09.025
9. Anitescu, C., Yue, J., Zhang, Y. J., Rabczuk, T. (2015). An isogeometric collocation method using superconvergent points. *Computer Methods in Applied Mechanics & Engineering, 284,* 1073–1097. https://doi.org/10.1016/j.cma.2014.11.038
10. Quarteroni, A. (2017). Isogeometric analysis. In: *Numerical models for differential problems*. vol. 16. Cham: Springer.
11. Hughes, T. J. R., Reali, A., Sangalli, G. (2015). Efficient quadrature for NURBS-based isogeometric analysis. *Computer Methods in Applied Mechanics & Engineering, 199(5),* 301–313.
12. Barton, M., Calo, V. (2016). Gauss-Galerkin quadrature rules for quadratic and cubic spline spaces and their application to isogeometric analysis. *Computer-Aided Design, 82,* 57–67. https://doi.org/10.1016/j.cad.2016.07.003
13. Hiemstra, R. R., Calabrò, F., Schillinger, D., Hughes, T. J. R. (2017). Optimal and reduced quadrature rules for tensor product and hierarchically refined splines in isogeometric analysis. *Computer Methods in Applied Mechanics & Engineering, 316,* 966–1004. https://doi.org/10.1016/j.cma.2016.10.049
14. Wu, Z., Wang, S., Shao, W., Yu, L. (2020). Reusing the evaluations of basis functions in the integration for isogeometric analysis. *Computer Modeling in Engineering & Sciences, 316(2),* 459–485. https://doi.org/10.32604/cmes.2020.08697
15. Pan, M., Jüttler, B., Giust, A. (2020). Fast formation of isogeometric galerkin matrices via integration by interpolation and look-up. *Computer Methods in Applied Mechanics & Engineering, 366,* 113005. https://doi.org/10.1016/j.cma.2020.113005
16. Pan, M., Jüttler, B., Scholz, F. (2022). Efficient matrix computation for isogeometric discretizations with hierarchical B-splines in any dimension. *Computer Methods in Applied Mechanics & Engineering, 388,* 114210. https://doi.org/10.1016/j.cma.2021.114210
17. Cottrell, J. A., Reali, A., Bazilevs, Y., Hughes, T. J. R. (2006). Isogeometric analysis of structural vibrations. *Computer Methods in Applied Mechanics & Engineering, 195(41),* 5257–5296. https://doi.org/10.1016/j.cma.2005.09.027
18. Morganti, S., Auricchio, F., Benson, D. J., Gambarin, F. I., Hartmann, S. et al. (2015). Patient-specific isogeometric structural analysis of aortic valve closure. *Computer Methods in Applied Mechanics & Engineering, 284,* 508–520. https://doi.org/10.1016/j.cma.2014.10.010
19. Bouclier, R., Passieux, J. C. (2018). A Nitsche-based non-intrusive coupling strategy for global/local isogeometric structural analysis. *Computer Methods in Applied Mechanics & Engineering, 340(1),* 253–277. https://doi.org/10.1016/j.cma.2018.05.022
20. Bazilevs, Y., Calo, V. M., Zhang, Y., Hughes, T. J. R. (2006). Isogeometric fluid-structure interaction analysis with applications to arterial blood flow. *Computational Mechanics, 38(4–5),* 310–322. https://doi.org/10.1007/s00466-006-0084-3
21. Bazilevs, Y., Calo, V. M., Hughes, T. J. R., Zhang, Y. (2008). Isogeometric fluid-structure interaction: Theory, algorithms, and computations. *Computational Mechanics, 43(1),* 3–37. https://doi.org/10.1007/s00466-008-0315-x
22. Seyfaddini, F., Nguyen-Xuan, H., Nguyen, V. H. (2021). A semi-analytical isogeometric analysis for wave dispersion in functionally graded plates immersed in fluids. *Acta Mechanica, 232(3),* 15–32.

23. Auricchio, F., Beirão da Veiga, L., Buffa, A., Lovadina, C., Reali, A. et al. (2007). A fully "locking-free" isogeometric approach for plane linear elasticity problems: A stream function formulation. *Computer Methods in Applied Mechanics & Engineering, 197(1–4),* 160–172. https://doi.org/10.1016/j.cma.2007.07.005
24. Elguedj, T., Bazilevs, Y., Calo, V. M., Hughes, T. J. R. (2010). $\bar{B}$ and $\bar{F}$ projection methods for nearly incompressible linear and non-linear elasticity and plasticity using higher-order NURBS elements. *Computer Methods in Applied Mechanics & Engineering, 197(33),* 2732–2762.
25. Benson, D. J., Bazilevs, Y., Hsu, M. C., Hughes, T. J. R. (2011). A large deformation, rotation-free, isogeometric shell. *Computer Methods in Applied Mechanics & Engineering, 200(13),* 1367–1378. https://doi.org/10.1016/j.cma.2010.12.003
26. Wolfgang, A., Frenzel, M. A., Cyron, C. (2008). Isogeometric structural shape optimization. *Computer Methods in Applied Mechanics & Engineering, 197(33),* 2976–2988.
27. Qian, X. (2010). Full analytical sensitivities in NURBS based isogeometric shape optimization. *Computer Methods in Applied Mechanics & Engineering, 199(29),* 2059–2071. https://doi.org/10.1016/j.cma.2010.03.005
28. Qian, X., Sigmund, O. (2011). Isogeometric shape optimization of photonic crystals via Coons patches. *Computer Methods in Applied Mechanics & Engineering, 200(25),* 2237–2255. https://doi.org/10.1016/j.cma.2011.03.007
29. Nguyen, V. P., Anitescu, C., Bordas, S. P. A., Rabczuk, T. (2015). Isogeometric analysis: An overview and computer implementation aspects. *Mathematics & Computers in Simulation, 117,* 89–116. https://doi.org/10.1016/j.matcom.2015.05.008
30. Schillinger, D., Evans, J. A., Reali, A., Scott, M. A., Hughes, T. J. (2013). Isogeometric collocation: Cost comparison with Galerkin methods and extension to adaptive hierarchical NURBS discretizations. *Computer Methods in Applied Mechanics & Engineering, 13(1),* 170–232.
31. Auricchio, F., Beirão da Veiga, L., Hughes, T. J., Reali, A., Sangalli, G. (2012). Isogeometric collocation for elastostatics and explicit dynamics. *Computer Methods in Applied Mechanics & Engineering, 249–252(1),* 2–14. https://doi.org/10.1016/j.cma.2012.03.026
32. Demko, S. (1985). On the existence of interpolating projections onto spline spaces. *Journal of Approximation Theory, 43(2),* 151–156. https://doi.org/10.1016/0021-9045(85)90123-6
33. Gomez, H., Lorenzis, L. D. (2016). The variational collocation method. *Computer Methods in Applied Mechanics & Engineering, 309,* 152–181. https://doi.org/10.1016/j.cma.2016.06.003
34. Montardini, M., Sangalli, G., Tamellini, L. (2017). Optimal-order isogeometric collocation at Galerkin superconvergent points. *Computer Methods in Applied Mechanics & Engineering, 316,* 741–757. https://doi.org/10.1016/j.cma.2016.09.043
35. Wang, D., Qi, D., Li, X. (2021). Superconvergent isogeometric collocation method with Greville points. *Computer Methods in Applied Mechanics & Engineering, 377,* 113689. https://doi.org/10.1016/j.cma.2021.113689
36. Atroshchenko, E., Gang, X., Tomar, S., Bordas, S. P. A. (2018). Weakening the tight coupling between geometry and simulation in isogeometric analysis: From sub-and super-geometric analysis to Geometry Independent Field approximaTion (GIFT). *International Journal for Numerical Methods in Engineering, 114(10),* 1131–1159. https://doi.org/10.1002/nme.5778
37. Beirão da Veiga, L., Lovadina, C., Reali, A. (2012). Avoiding shear locking for the Timoshenko beam problem via isogeometric collocation methods. *Computer Methods in Applied Mechanics & Engineering, 241–244,* 38–51. https://doi.org/10.1016/j.cma.2012.05.020
38. Auricchio, F., Beirão da Veiga, L., Kiendl, J., Lovadina, C., Reali, A. (2013). Locking-free isogeometric collocation methods for spatial Timoshenko rods. *Computer Methods in Applied Mechanics & Engineering, 263(15),* 113–126. https://doi.org/10.1016/j.cma.2013.03.009

39. Casquero, H., Liu, L., Bona-Casas, C., Zhang, Y., Gomez, H. (2016). A hybrid variational-collocation immersed method for fluid-structure interaction using unstructured T-splines. *International Journal for Numerical Methods in Engineering, 105(11),* 855–880. https://doi.org/10.1002/nme.5004
40. Morganti, S., Callari, C., Auricchio, F., Reali, A. (2018). Mixed isogeometric collocation methods for the simulation of poromechanics problems in 1D. *Meccanica, 53,* 1441–1454. https://doi.org/10.1007/s11012-018-0820-8
41. Morganti, S., Fahrendorf, F., Lorenzis, L. D., Evans, J. A., Hughes, T. et al. (2021). Isogeometric collocation: A mixed displacement-pressure method for nearly incompressible elasticity. *Computer Modeling in Engineering & Sciences, 129(26),* 1125–1150. https://doi.org/10.32604/cmes.2021.016832
42. Maurin, F., Greco, F., Coox, L., Vandepitte, D., Desmet, W. (2018). Isogeometric collocation for kirchhoff-Love plates and shells. *Computer Methods in Applied Mechanics & Engineering, 329,* 396–420. https://doi.org/10.1016/j.cma.2017.10.007
43. Pagani, L., Scott, P. J. (2018). Curvature based sampling of curves and surfaces. *Computer Aided Geometric Design, 59(1),* 32–48. https://doi.org/10.1016/j.cagd.2017.11.004
44. Filip, D., Magedson, R., Markot, R. (1986). Surface algorithms using bounds on derivatives. *Computer Aided Geometric Design, 3(4),* 295–311. https://doi.org/10.1016/0167-8396(86)90005-1
45. Park, H. (2004). An error-bounded approximate method for representing planar curves in B-splines. *Computer Aided Geometric Design, 21(5),* 479–497. https://doi.org/10.1016/j.cagd.2004.03.003
46. Razdan, A. (1999). Knot placement for B-spline curve approximation. In: *Technical report*. Tempe, AZ: Arizona State University.
47. Hernández-Mederos, V., Estrada-Sarlabous, J. (2003). Sampling points on regular parametric curves with control of their distribution. *Computer Aided Geometric Design, 20(6),* 363–382. https://doi.org/10.1016/S0167-8396(03)00079-7
48. Lu, L., Zhao, S. (2019). High-quality point sampling for B-spline fitting of parametric curves with feature recognition. *Journal of Computational & Applied Mathematics, 345(1),* 286–294. https://doi.org/10.1016/j.cam.2018.04.008
49. Schoenberg, I. J., Whitney, A. (1953). On Pólya frequency function. III: The positivity of translation determinants with an application to the interpolation problem by spline curves. *Transactions of the American Mathematical Society, 74(2),* 246–259.
50. Hosseini, S. F., Hashemian, A., Reali, A. (2020). Studies on knot placement techniques for the geometry construction and the accurate simulation of isogeometric spatial curved beams. *Computer Methods in Applied Mechanics & Engineering, 360,* 112705. https://doi.org/10.1016/j.cma.2019.112705