SIAC Filters for Discontinuous Galerkin Methods

Xiaozhou Li

ISBN 9789461865007

Smoothness-Increasing Accuracy-Conserving Filters for Discontinuous Galerkin Methods:

Challenging the Assumptions of Symmetry and Uniformity

Xiaozhou Li

Smoothness-Increasing Accuracy-Conserving Filters for Discontinuous Galerkin Methods: Challenging the Assumptions of Symmetry and Uniformity

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof.ir. K.C.A.M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op
donderdag 9 juli 2015 om 10:00 uur

door

XIAOZHOU LI

Bachelor of Science, Mathematics and Applied Mathematics,
University of Science and Technology of China, China
geboren te Chongqing, China

This dissertation has been approved by the

Promotor: Prof.dr.ir. C. Vuik
Copromotor: Dr. J.K. Ryan

Composition of the doctoral committee:

Rector Magnificus,	voorzitter
Prof.dr.ir. C. Vuik,	Technische Universiteit Delft, promotor
Dr. J.K. Ryan,	University of East Anglia, United Kingdom, copromotor

Independent members:

Prof.dr.ir. A.W. Heemink,	Technische Universiteit Delft
Dr.ir. M.I. Gerritsma,	Technische Universiteit Delft
Prof.dr.ir. J.E. Frank,	Universiteit Utrecht
Prof.dr.ir. J.J.W. van der Vegt,	Universiteit Twente
Prof.dr. R.M. Kirby,	University of Utah, United States

Smoothness-Increasing Accuracy-Conserving Filters for Discontinuous Galerkin Methods: Challenging the Assumptions of Symmetry and Uniformity.

Dissertation at Delft University of Technology.

This research was carried out at Delft Institute of Applied Mathematics, Delft University of Technology, and sponsored by the Air Force Office of Scientific Research (AFOSR), Air Force Material Command, USAF, under grant number FA8655-09-1-3017.

ISBN 978-94-6186-500-7

Published by TU Delft Library.

Printed in the Netherlands by Ridderprint.

Summary

SIAC Filters: Challenging the Assumptions of Symmetry and Uniformity

In this dissertation, we focus on exploiting superconvergence for discontinuous Galerkin methods and constructing a superconvergence extraction technique, in particular, Smoothness-Increasing Accuracy-Conserving (SIAC) filtering. The SIAC filtering technique is based on the superconvergence property of discontinuous Galerkin methods and aims to achieve a solution with higher accuracy order, reduced errors and improved smoothness.

The main contributions described in this dissertation are: 1) an efficient one-sided SIAC filter for both uniform and nonuniform meshes; 2) one-sided derivative SIAC filters for nonuniform meshes; 3) the theoretical and computational foundation for using SIAC filters for nonuniform meshes; and 4) the application of SIAC filters for streamline integration.

One-sided SIAC filtering is a technique that enhances the accuracy and smoothness of the DG solution near boundary regions. Previously introduced one-sided filters are not directly useful for most applications since they are limited to uniform meshes, linear equations, and the use of multi-precision packages in the computation. Also, the theoretical proofs relied on a periodic boundary assumption. We aim to overcome these deficiencies and develop a new fast one-sided filter for both uniform and nonuniform meshes. By studying B-splines and the negative order norm analysis, we generalized the structure of SIAC filters from a combination of central B-splines to using more general B-splines. Then, a "boundary shape" B-spline (using multiple knots at the boundary) was used to construct a new one-sided filter. We also presented the first theoretical proof of convergence for SIAC filtering over nonuniform meshes (smoothly-varying meshes).

One purpose of SIAC filtering is to improve the smoothness of DG solutions. Because of the increased smoothness, we can obtain a better approximation for the derivatives of DG solutions. Derivative filtering over the interior region of uniform meshes

was previously studied. However, nonuniform meshes and boundary regions remain a significant challenge. We extended the one-sided filter to a one-sided derivative filter. To deal with nonuniform meshes, we investigated the negative order norm over arbitrary meshes and proposed to scale the one-sided derivative filter with scaling h^{μ}. For arbitrary nonuniform rectangular meshes, we proved that the one-sided derivative filter can enhance the order of convergence for the αth derivative of the DG solution from $k+1-\alpha$ to $\mu(2k+2)$, where $\mu \approx \frac{2}{3}$.

The most challenging part of this project is recovering the superconvergence of the DG solution over nonuniform meshes through SIAC filtering. Typically, most theoretical proofs for SIAC filters are limited to uniform meshes (or translation invariant meshes). The only theoretical investigations for nonuniform meshes were included in our one-sided and derivative filtering studies. Although our earlier research for nonuniform meshes provides good engineering accuracy, we want to do better mathematically. This is not an easy task since unstructured meshes give DG solutions irregular performance under the negative order norm. In our work, we introduced a parameter to measure the unstructuredness of a given nonuniform mesh. Then, by adjusting the scaling of the SIAC filter based on this unstructuredness parameter, we can obtain the optimal filtered approximation (best accuracy) over a given nonuniform mesh.

SIAC filtering for streamline integration is an attempt to use SIAC filters in a realistic engineering application. By using the one-sided filter and one-sided derivative filter, we designed an efficient algorithm: filtering the velocity field along the streamline and then use a backward differentiation formula for integration. Compared to the traditional method of filtering the entire field (multi-dimensional algorithm), the computational cost drops dramatically since its complexity corresponds to a one-dimensional algorithm.

We finally note that most of the work presented originates from published and submitted papers for the past four years of this PhD research.

Samenvatting

SIAC Filters: Symmetrie- en Uniformiteitsaannamen op de proef gesteld

In dit proefschrift focussen we op het ontwikkelen van de theorie achter superconvergentie voor discontinue Galerkinmethoden en het construeren van een superconvergentie-extractietechniek: de Smoothness-Increasing Accuracy-Conserving (SIAC) filter. De SIAC filtertechniek is gebaseerd op de superconvergentie-eigenschap van discontinue Galerkinmethoden en beoogt een oplossing te verkrijgen met een hogere orde van nauwkeurigheid, kleinere fouten en verbeterde gladheid.

De belangrijkste bijdragen die in dit proefschrift beschreven worden zijn: 1) een efficiënte eenzijdige SIAC filter voor zowel uniforme als niet-uniforme roosters; 2) eenzijdige afgeleiden SIAC filters voor niet-uniforme roosters; 3) de theoretische en rekenkundige basis voor het gebruik van SIAC filters voor niet-uniforme roosters; en 4) de toepassing van SIAC filters voor stroomlijnintegratie.

Eenzijdige SIAC filtering is een techniek die de nauwkeurigheid en gladheid van de DG oplossing in de buurt van grensregio's verbetert. Eerder geïntroduceerde eenzijdige filters zijn niet onmiddellijk toepasbaar voor de meeste toepassingen, aangezien zij beperkt zijn tot uniforme roosters, lineaire vergelijkingen en het gebruik van meervoudige precisie in de rekenpakketten. Daarbij steunen de theoretische bewijzen op periodieke randvoorwaarden. Onze bedoeling was om deze gebreken te overwinnen en om een nieuwe snelle eenzijdige filter voor zowel uniforme als niet-uniforme roosters te ontwikkelen. Door het bestuderen van B-splines en de negatieve-ordenormanalyse hebben we de structuur van SIAC filters gegeneraliseerd van een combinatie van centrale B-splines naar het gebruik van meer algemene B-splines. Vervolgens is een 'grensvorm' B-spline (gebruikmakend van meerdere knopen aan de rand) gebruikt om een nieuwe eenzijdige filter te construeren. Ook presenteren we het eerste theoretische bewijs van convergentie voor SIAC filtering over niet-uniforme roosters (gelijkmatig variërende roosters).

Een doel van SIAC filtering is het verbeteren van de gladheid van DG benaderingen.

Vanwege de verbeterde gladheid kunnen we een betere benadering voor de afgeleiden van DG oplossingen verkrijgen. Afgeleidefiltering over het inwendige gebied van uniforme roosters is al eerder bestudeerd. Echter, niet-uniforme roosters en grensregio's blijven een grote uitdaging. Wij hebben de eenzijdige filter uitgebreid naar een eenzijdige afgeleidefilter. Om niet-uniforme roosters te kunnen behandelen onderzochten we de negatieve-ordenorm voor willekeurige roosters, en stelden voor om de eenzijdige afgeleidefilter te schalen met schaalfactor h^μ. Voor willekeurige niet-uniforme rechthoekige roosters hebben we bewezen dat de eenzijdige afgeleidefilter de convergentieorde voor de afgeleide van orde α van de DG oplossing kan verhogen van $k+1-\alpha$ naar $\mu(2k+2)$, waarbij $\mu \approx 2/3$.

Het meest uitdagende deel van dit project is het terugvinden van de superconvergentie van de DG oplossing over niet-uniforme roosters met behulp van SIAC filtering. Over het algemeen zijn theoretische bewijzen voor SIAC filters begrensd tot uniforme roosters (of translatie-invariante roosters). De enige theoretische onderzoeken voor niet-uniforme roosters zijn inbegrepen in onze studies naar eenzijdige filters en afgeleidefilters. Hoewel ons eerder onderzoek naar niet-uniforme roosters ons voorziet van een goede nauwkeurigheid voor ingenieurs willen we wiskundig gezien een hogere nauwkeurigheid bewijzen. Dit is geen eenvoudige opgave, aangezien DG oplossingen afwijkend presteren in de negatieve-ordenorm indien ongestructureerde roosters gebruikt worden. In ons werk hebben we een parameter geïntroduceerd die de mate van ongestructureerdheid van een gegeven niet-uniform rooster bepaalt. Door het aanpassen van de schaalfactor gebaseerd op deze ongestructureerdheidsparameter kunnen we de optimale gefilterde benadering (hoogste nauwkeurigheid) over een gegeven niet-uniform rooster bepalen.

SIAC filtering voor stroomlijnintegratie is een poging om SIAC filters in een realistische technische toepassing te gebruiken. Door gebruik te maken van de eenzijdige filter en de eenzijdige afgeleidefilter hebben we een efficiënt algoritme ontworpen: hierbij wordt het snelheidsveld langs de stroomlijn geïntegreerd en daarna wordt een achterwaartse differentieformule voor integratie gebruikt. Vergeleken met de traditionele methode waarbij het volledige veld gefilterd wordt (multidimensionaal algoritme) neemt de rekentijd enorm af omdat de complexiteit overeenkomt met een eendimensionaal algoritme.

Tenslotte merken we op dat het grootste deel van dit gepresenteerde werk voortkomt uit gepubliceerde en ingestuurde artikelen over de afgelopen vier jaar van dit promotieonderzoek.

Contents

0 Introduction

In the last decades, discontinuous Galerkin (DG) methods have been under rapid development and attracted considerable attention from diverse areas. Since DG methods allow discontinuities in the approximate solutions of general finite element methods, the DG method also can be considered as a generalization of finite volume methods. As a consequence, DG methods incorporate the features of finite element methods and finite volume methods in a very natural way. The main advantages of the DG method are:

- High order accuracy. DG schemes of arbitrary high order of accuracy can be obtained by suitably choosing the degree of the approximation polynomials.

- DG methods are suited to handling complicated geometries and boundary conditions.

- DG methods are highly parallelizable and can easily handle adaptive strategies.

Of course, the increased accuracy of DG methods requires additional degrees of freedom compared to finite element methods. Later, as DG methods have matured, researchers concentrate on more interesting aspects of the method. In recent research, the additional degrees of freedom of DG methods, which was considered as a disadvantage, allows for recovery of hidden accuracy (superconvergence) of DG methods. As a consequence, a robust, accurate, and efficient method for theoretically and numerically extracting this hidden accuracy is of considerable importance and, as expected, has attracted the interest of many researchers. In this dissertation, research that concentrates on exploiting superconvergence for discontinuous Galerkin methods and a superconvergence extraction technique, Smoothness-Increasing Accuracy-Conserving (SIAC) filtering, is discussed. The new contributions of our work are: an efficient one-sided filter for both uniform and nonuniform meshes; derivative filters for nonuniform meshes and near boundaries; the theoretical and computational foundation for using SIAC filters for nonuniform meshes; and the applications of SIAC filters in the visualization areas.

0.1 A Brief Historical Perspective

0.1.1 Discontinuous Galerkin Methods

The original discontinuous Galerkin method was introduced by Reed and Hill [54] in 1973 for the neutron transport equation

$$\sigma u + \nabla \cdot (\mathbf{a}u) = \mathbf{f},$$

where σ is a real number and $\mathbf{a}$ is a constant vector. This method was referred to as the discontinuous Galerkin method by Lesaint and Raviart [45] in 1974. In the same publication [45], Lesaint and Raviart presented the first mathematical analysis of the DG method and proved a convergence rate of k in the L^2 norm for general triangulations and $k+1$ for rectangular grids. In the later part of the 1990s, Cockburn and Shu successfully extend the DG methods to hyperbolic problems in a series of papers [26, 20, 27, 28, 21] and proposed using Runge-Kutta methods for time discretization. This so-called Runge-Kutta discontinuous Galerkin (RKDG) method incorporated the ideas of numerical flux and slope limiter into the finite element framework to produce high-order accurate, nonlinearly stable schemes. Beginning with Cockburn and Shu's efforts, in the recent decades, the DG method finally steps into a rapid evolution. For applying DG methods for high order equations, Cockburn and Shu [27, 29] proposed the local discontinuous Galerkin methods (LDG) for the convection-diffusion problem. A series of studies of diverse high order equations were then been made, to name a few [70, 71, 73, 47]. The DG method has found its use very quickly transitioned in the applied sciences and engineering as diverse as aeroacoustics, turbulent flows, modeling of shallow water, image processing, among many others. A more detailed overview of the evolution of the discontinuous Galerkin method can be found in [24, 36].

0.1.2 Superconvergence

Along with the development of finite element methods, the superconvergence of the finite element methods also becomes a dynamically developing area of research. The superconvergence of the finite element methods is a phenomenon where the order of convergence, under certain measures, is higher than the accuracy order under the standard L^2 norm. In general, these measures include the negative order norm, point-wise, average over on element, special projections, etc. In the literature, the term superconvergence was first used by Douglas and Dupont in [33]. Superconvergence has been extensively studied, up to now there are more than thousands of research paper concentrating on this subject, to name a few [13, 34, 59, 62, 63, 67]. A bibliography (before 1998) includes 600 references given in [44].

Superconvergence in DG methods is gaining an increasing amount of attention in recent years [15, 37, 69, 77, 76]. Superconvergence of DG methods is mainly divided into the following three types: 1) superconvergence of DG errors in the negative order norm, which in the ideal situation gives a superconvergence rate of $2k+1$, see [13, 25, 62, 51, 40, 41, 39]; 2) superconvergence of DG errors at particular points (Radau points) or the average over an element, contributed (to name a few) by Adjerid et al. [4, 2, 6, 3, 5, 16], Bacouch et al. [11, 10, 8, 9] and Zhang et al. [15, 69, 77]; 3) the

superconvergence between the DG solution and a special projection, see [17, 18, 74]. The focus of this thesis, SIAC filtering, is developed mainly on the studies of the superconvergence on the negative order norm, and also involving the superconvergence at Radau points.

0.1.3 SIAC Filters

As a superconvergence extracting technique, SIAC filtering developed from a post-processing technique for enhancing the accuracy of solutions of finite element methods introduced by Bramble and Schatz [13] in 1977. The work of Bramble and Schatz [13] demonstrated that the superconvergence in the negative order norm can be extracted and recovery of higher-order approximations in the L^2 norm can be obtained. A fundamental relation between the negative order norm and the L^2 norm was established. In the same year, Thomée extended this technique to approximate the derivatives in the finite element method and presented further investigation of the relation between these two norms. Then 1978, Mock and Lax [51] deduced the post-processing technique from another point of view by studying the discontinuous solutions of linear hyperbolic equations.

The first extension of this post-processing technique to discontinuous Galerkin methods was given by Cockburn et al. [25]. In [25], they applied DG methods to linear hyperbolic equations with periodic boundary conditions [25]. The superconvergence rate of $2k + 1$ is proven in the negative order norm and after post-processing in the L^2 norm. Later, Ryan and Shu [57] proposed the idea of a one-sided post-processing technique, which can be applied to boundary regions, discontinuities of solutions and interfaces of elements. This one-sided idea was modified in [65] and renamed as a position-dependent filter. The respect error estimates were presented in [39]. In 2008, the first numerical exploration of the post-processing over nonuniform meshes was given in [30] and numerically obtained the superconvergence rate of $2k+1$ for some particular nonuniform meshes. There are a wide variety of studies using this post-processing technique, such as applied to an aeroacoustic problem [58], derivatives in the DG approximation [56], convection-diffusion equations [40] and streamline visualization [61, 68]. The name Smoothness-Increasing Accuracy-Conserving filtering was first used in [61], and nowadays refers to the generalized post-processing technique based on the negative order norm.

0.2 Contributions

The main purpose of SIAC filtering is twofold: extracting useful information within the DG solution to improve the accuracy of the solution; removing the oscillations within the DG error and improve the smoothness of the solution. The particular contributions of this thesis are the following:

- **One-Sided SIAC Filtering Over Uniform and Nonuniform Meshes.**

Typically, most of the studies of SIAC filtering are confined to the interior of the underlying domain. For boundary regions, a one-sided filter is needed. The existing one-sided filters are not directly useful for most applications since they were limited to

uniform meshes, linear equations, using multi-precision packages in the computation. Also, the theoretical proof relied on the periodic boundary assumption. We aimed to overcome these deficiencies and develop a new fast one-sided filter for both uniform and nonuniform meshes. By studying B-splines and the negative order norm analysis, we generalized the structure of SIAC filters from a combination of central B-splines to using more general B-splines. Then, a "boundary shape" B-spline (using multiplicity knots at the boundary) was used to construct a new one-sided filter. We also presented the first theoretical proof of convergence for SIAC filtering over nonuniform meshes (smoothly-varying meshes). Details are given in Chapter 2.

• **Derivative Filtering Over Nonuniform Meshes and Near Boundaries**

One advantage of SIAC filtering is that it improves the smoothness of DG solutions. Because of the increased smoothness, we can obtain a better approximation of the derivatives of DG solutions. The derivative filtering over the interior region of uniform meshes was previously studied. However, nonuniform meshes and boundary regions still remain a big challenge. We extended the one-sided filter to a one-sided derivative filter. Nonuniform meshes are a difficult area, by investigating negative order norm over arbitrary meshes, we proposed to scale the one-sided derivative filter with scaling h^{μ}. For arbitrary nonuniform rectangle meshes, we proved that the one-sided derivative filter can enhance the order of convergence for αth derivative of DG solution from $k+1-\alpha$ to $\mu(2k+2)$, where $\mu \approx \frac{2}{3}$. Details are in Chapter 3.

• **Superconvergence Extraction Over Nonuniform Meshes**

The most challenging part of this project is recovering the superconvergence of a DG solution over nonuniform meshes through SIAC filtering. Typically, most theoretical proofs for the SIAC filter are limited to uniform meshes (or translation invariant meshes). The few theoretical investigations for nonuniform meshes were given in the one-sided and derivative filtering studies. Although our early research for nonuniform meshes was able to provide good engineering accuracy, we want to do better mathematically. This is not an easy task since unstructured meshes give DG solutions irregular performance under the negative order norm. In our work, we introduced a parameter to measure the "unstructuredness" of a given nonuniform mesh. Then by adjusting the scaling of SIAC filter based on this "unstructuredness" parameter, we are able to obtain the optimal filtered approximation (best accuracy) over a given nonuniform mesh. Details are in Chapter 4.

• **Application to Streamline Integration**

After introducing the new one-sided filter, we aimed to verify its usage in realistic engineering applications. The topic we choose was streamline integration. By taking advantage of the one-sided property of the new filter, we designed an efficient algorithm which filters the velocity field along the streamline, then uses a backward differentiation formula (BDF) for integration. Compared to the traditional method that filters the entire field (multi-dimensions algorithm), the computational cost drops dramatically since it is only a one-dimensional algorithm. Details can be found in Chapter 5.

• **Further Topics of SIAC Filters**

After studying SIAC filters for a broad range of applications, we retured to further investigations of SIAC filters themselves. Further topics such the uniqueness of the structure SIAC filters, the effects of the order of B-splines to SIAC filters and the

compressed SIAC filters are included in Chapter 6. These topics give us in-depth insight into SIAC filters and reveal some future directions for the development of SIAC filters.

1 Background

This chapter briefly introduces Discontinuous Galerkin (DG) methods and Smoothness-Increasing Accuracy-Conserving (SIAC) filters.

1.1 Notations of Function Spaces

Let us recall the norms of function spaces that will be used in the following. Consider a domain $\Omega \subset \mathbb{R}^d$, the standard L^2-norm over Ω is defined as

$$\|u\|_{0,\Omega} = \left\{ \int_\Omega u^2 dx \right\}^{\frac{1}{2}}.$$

For any nonnegative integer ℓ, the norm and seminorm of the Sobolev space $H^\ell(\Omega)$ are given by

$$\|u\|_{\ell,\Omega} = \left\{ \sum_{|\alpha|\leq\ell} \|D^\alpha u\|_{0,\Omega}^2 \right\}^{\frac{1}{2}}, \qquad |u|_{\ell,\Omega} = \left\{ \sum_{|\alpha|=\ell} \|D^\alpha u\|_{0,\Omega}^2 \right\}^{\frac{1}{2}}.$$

Then, we can define the negative order norm on the domain Ω as

$$\|u\|_{-\ell,\Omega} = \sup_{\phi\in\mathcal{C}_0^\infty(\Omega)} \frac{(u,\phi)_\Omega}{\|\phi\|_{\ell,\Omega}},$$

where $(\cdot,\cdot)$ represents an inner product. The negative order norm is the norm of the dual space of $H^\ell(\Omega)$. It was claimed in [25] that the negative order norm can be used to detect the oscillations of a function round zero.

Lastly, we introduce the notation for the divided differences. In the one-dimension case,

$$\partial_h u(x) = \frac{1}{h}\left(u\left(x+h/2\right) - u\left(x+h/2\right)\right),\ \partial_h^\alpha u = \partial_h\left(\partial_h^{\alpha-1}u\right),\ \alpha > 1,$$

and the multi-dimensional notation is defined analogously by using a tensor product.

1.2 Discontinuous Galerkin Methods

Although Reed and Hill [54] introduced the original DG method 40 years ago, it was only the last decade that DG methods have rapidly evolved for various applications. DG methods can be viewed as a combination of finite element methods and finite volume methods. It allows for discontinuities in the approximation space and introduces numerical fluxes.

More specially, consider a multi-dimensional linear hyperbolic equation for a domain $\Omega = [a_1, b_1] \times \cdots \times [a_d, b_d] \subset \mathbb{R}^d$,

$$\begin{aligned} u_t + \sum_{i=1}^{d} a_i u_{x_i} + a_0 u = 0, \qquad (\mathbf{x}, \mathbf{t}) \in \mathbf{\Omega} \times [\mathbf{0}, \mathbf{T}], \\ u(\mathbf{x}, \mathbf{0}) = u_0(\mathbf{x}), \end{aligned} \tag{1.1}$$

where u_0 is sufficiently smooth. To create the DG approximation, we first introduce a mesh tessellation. A rectangular mesh $\mathcal{T}_h$ of Ω is a finite collection of disjoint rectangles K, $K = \prod_{i=1}^{d} [x^{(i)}_{j-\frac{1}{2}}, x^{(i)}_{j+\frac{1}{2}}]$. Each $K \in \mathcal{T}_h$ is called a mesh element, and the mesh size is defined as

$$h = \max_{K \in \mathcal{T}_h} h_K,$$

where h_K denotes the diameter of the element K. The DG method seeks an approximation in the space of piecewise polynomials of degree $\leq k$,

$$V_h^k = \left\{ \varphi \in L^2(\Omega) : \varphi|_K \in \mathbb{P}^k, \forall K \in \mathcal{T}_h \right\}.$$

To find the DG approximation for solving Equation (1.1), we look for a function $u_h \in V_k$ such that, for each element K and all test function $v_h \in V_h^k$, we have

$$\begin{aligned} \int_K (u_h)_t v_h dK - \sum_{i=1}^{d} \int_K a_i u_h (v_h)_{x_i} dK \\ + \sum_{i=1}^{d} \int_{\partial K} a_i \hat{u}_h v_h n_i ds + \int_K a_0 u_h v_h dK = 0, \end{aligned}$$

or

$$\begin{aligned} ((u_h)_t, v_h)_K - \sum_{i=1}^{d} (a_i u_h, (v_h)_{x_i})_K \\ + \sum_{i=1}^{d} \int_{\partial K} a_i \hat{u}_h v_h n_i ds + (a_0 u_h, v_h) = 0, \end{aligned} \tag{1.2}$$

where $\hat{u}_h$ is the numerical flux. The numerical fluxes are chosen according to the partial differential equations and Finite Volum principals [46]. For the linear hyperbolic equation (1.1), we usually choose the standard upwind flux.

In this simple linear example, we can see the main components of the DG method, namely,

- the use of a discontinuous piecewise polynomial basis,
- the enforcement of the PDE by means of a Galerkin weak formulation,
- the introduction of the so-called numerical flux $\hat{u}_h$.

The choice of the numerical flux is the most import aspect of the DG methods since it affects the consistency, stability and accuracy. For questions of how to choose the numerical fluxes and a detailed analysis of DG methods is referred to [20, 27, 28, 21, 23, 29, 22] for details.

1.2.1 Superconvergence of DG Methods

Another crucial aspect of the DG methods are the error estimates of the DG solutions in different norms and the so-call superconvergence property. The superconvergence of the DG methods is a phenomenon where the order of convergence, under certain norms (or measures), is higher than the accuracy order under the L^2 norm. The relevant studies of superconvergence for DG methods include: Adjerid and Baccouch et al. [4, 3, 5, 11, 10] proved that the DG approximations have superconvergence order of $k+2$ at Radau points (roots of right Radau polynomials in the interior); the study of Celiker and Cockburn [16] showed that the numerical flux approximates the exact flux with convergence of order $2k+1$ at the element interfaces, etc. The techniques that extraction the superconvergence from the DG approximations are usually referred as superconvergence extract or post-processing techniques, one of these techniques that has obtain increased interests is the so-called smoothness-increasing accuracy-conserving (SIAC) filtering. The focus of this thesis, the SIAC filter, is developed mainly on the studies of the superconvergence of the DG approximation and its divided differences on the negative order norm. For uniform meshes, the main theorem is given below.

Theorem 1.2.1 (Cockburn et al. [25]). *Let u be the exact solution of equation (1.1) with periodic boundary conditions, and u_h the DG approximation derived by scheme (1.2). For a uniform mesh, the approximation and its divided differences in the L^2 norm, we have the following error estimate:*

$$\|\partial_h^\alpha (u - u_h)\|_{0,\Omega} \leq C h^{k+1}, \tag{1.3}$$

and in the negative order norm:

$$\|\partial_h^\alpha (u - u_h)\|_{-(k+1),\Omega} \leq C h^{2k+1}, \tag{1.4}$$

where $\alpha = (\alpha_1, \ldots, \alpha_d)$ is an arbitrary multi-index.

The relation between the L^2 norm and the negative order norm was given by

Lemma 1.2.2 (Bramble and Schatz [13]). *Let $\Omega_0 \subset\subset \Omega_1$ and s be an arbitrary but fixed nonnegative integer. Then for $u \in H^s(\Omega_1)$, there exist a constant C such that*

$$\|u\|_{0,\Omega_0} \leq C \sum_{|\alpha| \leq s} \|D^\alpha u\|_{-s,\Omega_1}.$$

Theorem 1.2.1 and Lemma 1.2.2 construct the theoretical foundation of applying the SIAC filter to DG solutions. Before we discuss the details, we first introduce SIAC filters.

1.3 Smoothness-Increasing Accuracy-Conserving Filters

The original SIAC filter we will be using was sourced from the accuracy enhance technique designed by Bramble and Schatz [13], Thomée [62] and Mock and Lax [51]. It was extended to DG methods by Cockburn et al. in [25]. The name "Smoothness-Increasing Accuracy-Conserving" was first used in [61].

1.3.1 Symmetric SIAC Filter

The symmetric SIAC filter used in [13, 25] is the archetype of SIAC filters. It is described below.

Assume the DG approximation is given over a uniform mesh, the SIAC filter is applied only at the final time T of the DG approximation, and the filtered solution $u_h^\star$, in the one-dimension case, is formed through convolution with the SIAC filter:

$$\begin{aligned} u_h^\star(x,T) &= \left(K_h^{(2k+1,k+1)} \star u_h(\cdot,T)\right) \\ &= \int_{-\infty}^{\infty} K_h^{(2k+1,k+1)}(x-\xi)u_h(\xi,T)d\xi. \end{aligned} \tag{1.5}$$

The symmetric SIAC filter, $K^{(2k+1,k+1)}$, is a linear combination of $2k+1$ central B-splines of order $k+1$,

$$K^{(2k+1,k+1)}(x) = \sum_{\gamma=0}^{2k} c_\gamma^{(2k+1,k+1)} \psi^{(k+1)}\left(x+k-\gamma\right), \tag{1.6}$$

and the scaled filter

$$K_h^{(2k+1,k+1)}(x) = \frac{1}{h} K^{(2k+1,k+1)}\left(\frac{x}{h}\right),$$

uses the scaling h, which is the diameter of uniform mesh. The $k+1$ order central B-spline, $\psi^{(k+1)}(x)$, can be constructed recursively by

$$\begin{aligned} \psi^{(1)}(x) &= \chi_{[-1/2,1/2]}(x), \\ \psi^{(\ell+1)}(x) &= \frac{1}{\ell}\left(\left(\frac{\ell+1}{2}+x\right)\psi^{(\ell)}\left(x+\frac{1}{2}\right)\right) \\ &\quad + \frac{1}{\ell}\left(\left(\frac{\ell+1}{2}-x\right)\psi^{(\ell)}\left(x-\frac{1}{2}\right)\right), \quad \ell \geq 1. \end{aligned} \tag{1.7}$$

For example:

$$\psi^{(2)}(x) = \begin{cases} 1+x, & x \in [-1,0), \\ 1-x, & x \in [0,1), \\ 0 & \text{else;} \end{cases}$$

$$\psi^{(3)}(x) = \begin{cases} \frac{1}{2}x^2 + \frac{3}{2}x + \frac{9}{8}, & x \in [-\frac{3}{2}, -\frac{1}{2}), \\ -x^2 + \frac{3}{4}, & x \in [-\frac{1}{2}, \frac{1}{2}), \\ \frac{1}{2}x^2 - \frac{3}{2}x + \frac{9}{8}, & x \in [\frac{1}{2}, -\frac{3}{2}) \\ 0 & \text{else.} \end{cases}$$

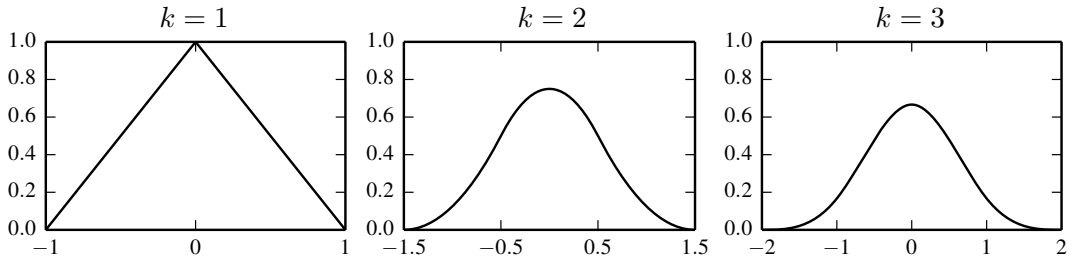


Figure 1.1: Central B-spline $\psi^{(k+1)}$ with $k = 1, 2, 3$.

B-splines have special properties that aid in the proofs of higher order accuracy in the negative order norm. One of these properties is differentiation:

Property 1.3.1 (Differentiation of Central B-spline)**.** *The αth derivative of a central B-spline is given by*

$$D^\alpha \psi_h^{(\ell)} = \partial_h^\alpha \psi_h^{(\ell-\alpha)},$$

where $\psi_h^{(\ell)}$ is the central B-spline with scaling h.

This shows that the derivatives of a central B-spline can be express simply by its divided differences.

The coefficients of the SIAC filter, $c_\gamma^{(2k+1,k+1)}$, are decided by implementing the property that the filter reproduces polynomials by convolution up to degree $2k$,

$$K^{(2k+1,k+1)} \star p = p, \quad p = 0, x, ..., x^{2k}. \tag{1.8}$$

For example, the symmetric SIAC filter (1.6) with $k = 2$ is given by

$$K^{(3,2)}(x) = -\frac{1}{12}\psi^{(2)}(x+1) + \frac{7}{6}\psi^{(2)}(x) - \frac{1}{12}\psi^{(2)}(x-1).$$

In the multi-dimensional case, the filter is a tensor product of the one-dimensional filters (1.6)

$$K_h^{(2k+1,k+1)}(\mathbf{x}) = \prod_{i=1}^{d} K_h^{(2k+1,k+1)}(x_i), \qquad \mathbf{x} = (x_1, \ldots, x_d) \in \mathbb{R}^d,$$

with the scaled filter $K_h^{(2k+1,k+1)}(\mathbf{x}) = \frac{1}{h^d} K^{(2k+1,k+1)}\left(\frac{\mathbf{x}}{h}\right)$.

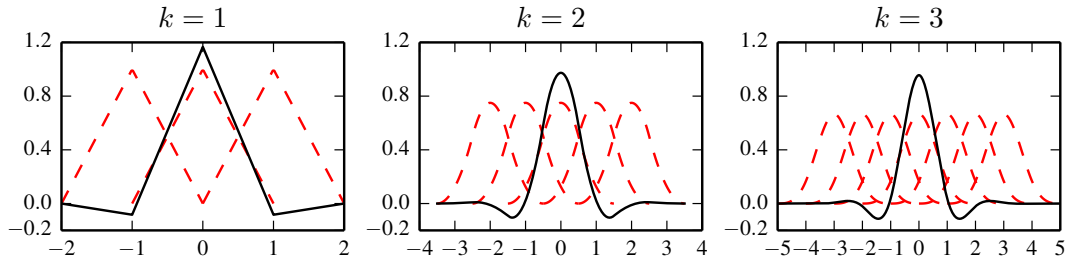


Figure 1.2: Solid black lines represent the symmetric filter $K^{(2k+1,k+1)}(x)$ with $k = 1, 2, 3$, dashed red lines represent the respective central B-splines. The filtered point is $x = 0$.

The Properties of SIAC Filter

In the above examples, we can see that the main features of the symmetric SIAC filter $K^{(2k+1,k+1)}$ are:

- Compact support, the support size is $3k + 1$;
- Symmetry with respect to the filtered point ($x = 0$);
- The filter satisfies
$$\int_{-\infty}^{\infty} K^{(2k+1,k+1)}(x)dx = 1;$$
- The filter is a $\mathcal{C}^{k-1}$ function and therefore so is the filtered solution $u_h^\star$.

Property 1.3.2. *The symmetric SIAC filter* $K^{(2k+1,k+1)}$ (1.6), *which satisfies* (1.8) *reproduces polynomial by convolution until degree of* $2k + 1$,

$$K^{(2k+1,k+1)} \star p = p,\ p = 1, x, \ldots, x^{2k+1}. \tag{1.9}$$

Proof. c.f. [64] □

Property 1.3.3 (Differential)**.** *As a consequence of the filter constructed using central B-splines (Property 1.3.1), one can express derivatives of the convolution with the filter in terms of simple difference quotients. It is trivial to verify that*

$$D^\alpha(K_h^{(2k+1,k+1)} \star u_h) = \tilde{K}_h^{(2k+1,k+1-\alpha,\alpha)} \star \partial_h^\alpha u_h,$$

where $\alpha = (\alpha_1, \ldots, \alpha_d)$ *is an arbitrary multi-index* ($\alpha_i < k + 1$) *and*

$$\tilde{K}_h^{(2k+1,k+1-\alpha,\alpha)} = \sum_{\gamma=0}^{2k} c_\gamma^{(2k+1,k+1)} \psi^{(k+1-\alpha)}(x + k - \gamma).$$

Properties 1.3.2 and 1.3.3 are the key to extract superconvergence from DG solutions, together with Theorem 1.2.1 and Lemma 1.2.2 we obtain the error estimates for the filtered solution $u_h^\star$.

Theorem 1.3.4 (Cockburn et al. [25])**.** *Under the same conditions in Theorem 1.2.1, denote* $\Omega_0 + 2supp(K_h^{(2k+1,k+1)}) \subset\subset \Omega_1 \subset \Omega$*, then*

$$\|u - K_h^{(2k+1,k+1)} \star u_h\|_{0,\Omega_0} \leq Ch^{2k+1}.$$

Remark 1.3.1. *The error estimates for the filtered solution in the* L^∞ *norm were proven in [39] under the same conditions of Theorem 1.3.4.*

Example 1.3.5. *As a simple example of the DG method and filtered solution, consider a linear hyperbolic equation*

$$\begin{aligned} u_t + u_x &= 0, \quad (x,t) \in [0,1] \times (0,T] \\ u(x,0) &= \sin(2\pi x) \end{aligned}$$

with final time $T = 1$ *over uniform meshes. The* L^2 *and* L^∞ *norm errors and respective accuracy order are given in Table 1.1, and Figure 1.3 shows the point-wise errors in log scale.*

Table 1.1: L^2- and $L^\infty-$errors for the DG approximation u_h and the filtered solution $u_h^\star$ for a linear advection equation.

	DG error				After filtering			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\mathbb{P}^1$								
20	4.60E-03	–	1.13E-02	–	1.97E-03	–	2.80E-03	–
40	1.09E-03	2.08	3.21E-03	1.82	2.44E-04	3.02	3.46E-04	3.02
80	2.67E-04	2.02	8.49E-04	1.92	3.02E-05	3.01	4.28E-05	3.01
160	6.65E-05	2.01	2.18E-04	1.96	3.76E-06	3.01	5.33E-06	3.01
$\mathbb{P}^2$								
20	1.07E-04	–	3.67E-04	–	4.11E-06	–	5.82E-06	–
40	1.34E-05	3.00	4.62E-05	2.99	9.49E-08	5.44	1.34E-07	5.44
80	1.67E-06	3.00	5.78E-06	3.00	2.49E-09	5.25	3.52E-09	5.26
160	2.09E-07	3.00	7.23E-07	3.00	7.75E-11	5.00	1.10E-10	5.00
$\mathbb{P}^3$								
20	2.06E-06	–	6.04E-06	–	6.97E-08	–	9.86E-08	–
40	1.29E-07	4.00	3.80E-07	3.99	2.83E-10	7.95	4.00E-10	7.95
80	8.07E-09	4.00	2.38E-08	4.00	1.23E-12	7.85	1.73E-12	7.85
160	5.04E-10	4.00	1.49E-09	4.00	1.59E-14	6.27	2.25E-14	6.27

Table 1.1 shows that the DG approximation has accuracy order of $k+1$, and it has been improved to $2k+1$ by applying SIAC filter. More importantly, we can see the accuracy of the DG solution has been significantly improved after filtering, which achieves the goal of extracting the "hidden accuracy".

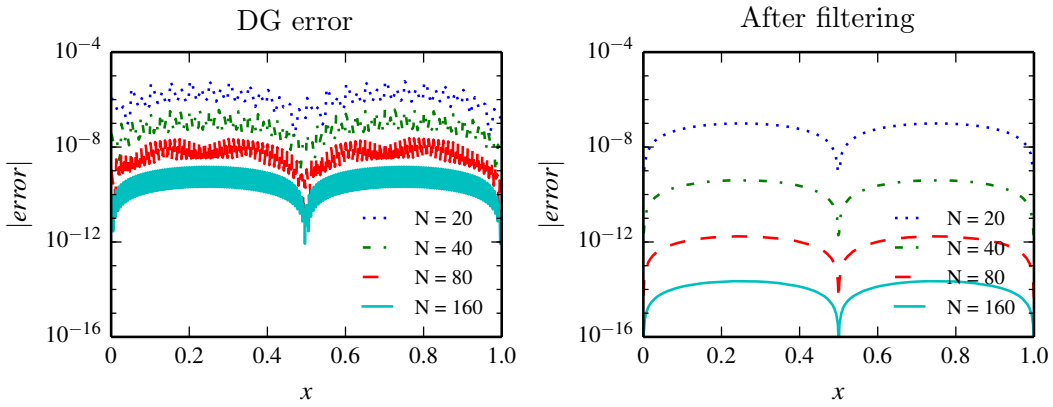


Figure 1.3: Comparison of the point-wise errors in log scale of the DG approximation together the filtered solution with polynomial $\mathbb{P}^3$ for a linear advection equation.

Figure 1.3 reveals another important feature of the SIAC filter as its name suggests, smoothness-increasing. The DG solution has weak continuity at the element interfaces, and the piecewise approximation represents as oscillations in the point-wise error plot in Figure 1.3. After filtering, due to the continuity of the symmetric filter $K^{(2k+1,k+1)}$, the filtered solution $u_h^\star$ is also a $\mathcal{C}^{k-1}$ function. It follows that the filtered solution is smoother compared to the original DG solution, and oscillations in the point-wise error plot have been eliminated.

Although Example 1.3.5 is quite simple, it has fully demonstrates the main purposes of SIAC filtering:

- Extract useful information from the DG solution and improve the accuracy of the solution;
- Remove the oscillations within the DG error and improve the smoothness of the solution.

1.3.2 Symmetric Derivative Filter

As mentioned before, one important feature of the filtered solution is the higher continuity compared to the original DG solution. This leads to a natural extension, the symmetric derivative filter, which aims to improve the accuracy of derivatives of DG solutions. The first derivative post-processing technique was introduced by Thomée [62], which generalized the results in [13] to derivatives in the finite element method. The symmetric derivative filter for DG methods was introduced by Ryan and Cockburn [56]. In the previous work, the authors identified two ways to calculate derivatives. The first method is a direct calculation of derivatives of filtered solution (1.5). By applying this method, the convergence rate of derivatives of filtered solutions is higher than derivatives of DG approximation itself, but the accuracy order decreases and oscillations in the error increase with each successive derivative. The second method

is employed to maintain the same $2k+1$ accuracy order as Theorem 1.3.4 regardless of the derivative order. In order to calculate the αth derivative of the DG solution without losing any accuracy order, we have to use higher order central B-splines to construct the symmetric derivative filter,

$$K^{(2k+1,k+1+\alpha)}(x) = \sum_{\gamma=0}^{2k} c_\gamma^{(2k+1,k+1+\alpha)} \psi^{(k+1+\alpha)}(x+k-\gamma). \tag{1.10}$$

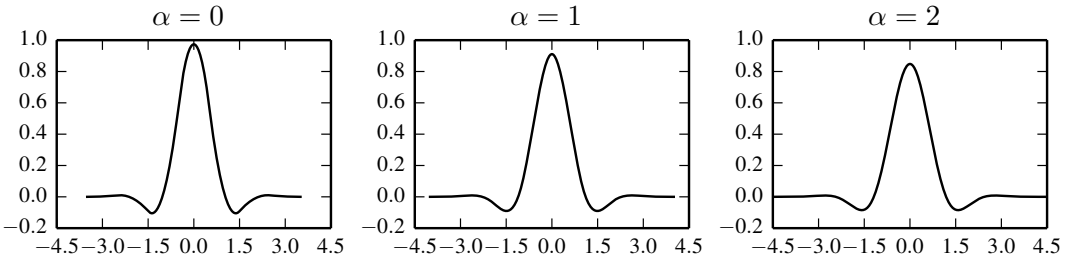


Figure 1.4: The symmetric derivative filter $K^{(2k+1,k+1+\alpha)}(x)$ given in (1.10) with $k=2$ and $\alpha = 0, 1, 2$. The filtered point is $x = 0$.

We note that the order of the B-splines is now $k+1+\alpha$ instead of $k+1$ in (1.6), and then the filtered solution becomes a $\mathcal{C}^{k-1+\alpha}$ function. Property 1.3.3 implies that one can write the αth derivative of the symmetric filter as $\frac{d^\alpha}{dx^\alpha} K_h^{(2k+1,k+1+\alpha)}(x) = \partial_h^\alpha \tilde{K}_h^{(2k+1,k+1,\alpha)}$, where

$$\tilde{K}_h^{(2k+1,k+1,\alpha)} = \sum_{\gamma=0}^{2k} c_\gamma^{(2k+1,k+1+\alpha)} \psi_h^{(k+1)}(x+k-\gamma).$$

By the property of convolution,

$$\begin{aligned} \partial_x^\alpha u_h^\star &= \partial_x^\alpha \left(K_h^{(2k+1,k+1+\alpha)} \star u_h \right) = \left(\frac{d^\alpha}{dx^\alpha} K_h^{(2k+1,k+1+\alpha)} \right) \star u_h \\ &= \left(\partial_h^\alpha \tilde{K}_h^{(2k+1,k+1,\alpha)} \right) \star u_h = \tilde{K}_h^{(2k+1,k+1,\alpha)} \star \partial_h^\alpha u_h. \end{aligned} \tag{1.11}$$

For uniform meshes, [56] showed filtered solution (1.11) has $2k+1$ superconvergence rate regardless of the derivative order α.

1.3.3 One-Sided SIAC Filters

The symmetric SIAC filter (1.6) takes a symmetric amount of information around the point being filtered. It means that the symmetric filter can not be applied near the domain boundaries. More precisely, within a distance of $\frac{3k+1}{2}h$ of the boundaries. In order to use the SIAC filter near the boundaries, Ryan and Shu [57] extended the idea

of the symmetric filter and developed a concept of the one-sided SIAC filter. This one-sided filter can be applied near boundaries or discontinuities in the exact solution, referred to the RS filter. The formula for the RS filter is given by

$$K^{(2k+1,k+1)}(x) = \sum_{\gamma=0}^{2k} c_\gamma^{(2k+1,k+1)} \psi^{(k+1)} \left(x - x_\gamma(\bar{x})\right), \tag{1.12}$$

where x_γ depends on the location of the evaluation point $\bar{x}$ and is given by

$$x_\gamma(\bar{x}) = -k + \gamma + [\lambda](\bar{x}),$$

with discrete shift

$$[\lambda](\bar{x}) = \begin{cases} \min\{0, -\frac{3k+1}{2} + \lfloor \frac{\bar{x}-x_L}{h} \rfloor\}, & \bar{x} \in [x_L, \frac{x_L+x_R}{2}), \\ \max\{0, \frac{3k+1}{2} + \lceil \frac{\bar{x}-x_R}{h} \rceil\}, & \bar{x} \in [\frac{x_L+x_R}{2}, x_R]. \end{cases} \tag{1.13}$$

Here x_L and x_R are the left and right boundaries, respectively. An example of the RS filter (for the left boundary) with $k = 2$ is given in Figure 1.5.

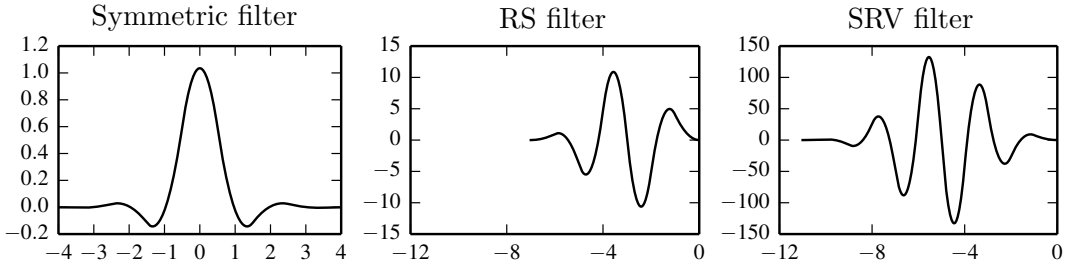


Figure 1.5: Comparison of symmetric filter (1.6), RS filter (1.12), and SRV filter (1.14) with $k = 2$. The filtered point is $x = 0$.

However, the performance of the RS filter was not very satisfactory as the errors had a stair-stepping-type structure, and the errors themselves were not reduced when the RS filter was applied to some DG solutions over coarse meshes, see Example 1.3.6. Later, van Slingerland, Ryan and Vuik [65] recast this formulation as a position-dependent SIAC filter, referred as SRV filter, by introducing a smooth shift function $\lambda(\bar{x})$ that aided in redefining the filter nodes and helped to ease the errors from the stair-stepping-type structure. In an attempt to reduce the errors, the authors doubled to $4k+1$ the number of central B-splines used in the filter when near a boundary. The SRV filter for filtering near the boundaries can then be written as

$$K^{(4k+1,k+1)}(x) = \sum_{\gamma=0}^{4k} c_\gamma^{(4k+1,k+1)} \psi^{(k+1)} \left(x - x_\gamma(\bar{x})\right), \tag{1.14}$$

where x_γ depends on the location of the evaluation point $\bar{x}$ and is given by

$$x_\gamma(\bar{x}) = -2k + \gamma + \lambda(\bar{x}),$$

with smooth shift

$$\lambda(\bar{x}) = \begin{cases} \min\{0, -\frac{5k+1}{2} + \frac{\bar{x}-x_L}{h}\}, & \bar{x} \in [x_L, \frac{x_L+x_R}{2}), \\ \max\{0, \frac{5k+1}{2} + \frac{\bar{x}-x_R}{h}\}, & \bar{x} \in [\frac{x_L+x_R}{2}, x_R]. \end{cases} \tag{1.15}$$

Here x_L and x_R are the left and right boundaries, respectively. In the interior, the symmetric filter uses $2k+1$ central B-splines is implemented. In order to provide a smooth transition between the SRV filter and the symmetric filter, a convex combination was used:

$$\begin{aligned} u_h^\star(x) =& \theta(x) \left(K_h^{(2k+1,k+1)} \star u_h \right)(x) \\ &+ (1-\theta(x)) \left(K_h^{(4k+1,k+1)} \star u_h \right)(x), \end{aligned} \tag{1.16}$$

where $\theta(x) \in \mathcal{C}^{k-1}$ such that $\theta = 1$ in the interior and $\theta = 0$ in the boundary regions. An example of the SRV filter (for the left boundary) with $k = 2$ is given in Figure 1.5.

Comparing the structures of the RS filter (1.12) and the SRV filter (1.14), there are two differences:

- the SRV filter uses many more B-splines $(4k + 1)$ than the RS filter $(2k + 1)$;
- by introducing a smoothly-varying shift (1.15) and convex combination (1.16) the SRV filter is smoother than the RS filter.

The error estimates of applying one-sided SIAC filters are similar to the symmetric filter, using a periodic boundary assumption, the filters solutions have an accuracy order of $2k + 1$ [39].

The performances of these two one-sided filters is done in the following example.

Example 1.3.6. *Consider the same problem in Example 1.3.5, we apply RS filter* (1.12) *and SRV filter* (1.14) *to the DG approximation u_h. Table 1.2 presents the L^2 and L^∞ errors, and the point-wise error plots are given in Figure 1.6. The results of original DG approximation and applying symmetric filter can be found in Table 1.1 and Figure 1.3.*

Through Example 1.3.6, it seems that the performance of the SRV filter is better than the RS filter. However, the true story is more complicated than this example shows. In Chapter 2, we will reveal more details of one-sided filters.

1.3.4 Implementation of SIAC Filter

As an additional remark, in this section, we briefly describe the implementation issues and strategies of applying the SIAC filter for DG solutions. For more details of efficient implementation of the SIAC filter, one can refer to the work of Mirzaee, Ryan and Kirby [50].

Table 1.2: L^2- and $L^\infty-$errors for the filtered solutions with RS filter (1.12) and SRV filter (1.14) for the linear advection equation.

	After RS filtering				After SRV filtering			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
				$\mathbb{P}^1$				
20	6.75E-03	–	2.24E-02	–	1.98E-03	–	2.80E-03	–
40	7.29E-04	3.21	3.13E-03	2.84	2.44E-04	3.02	3.46E-04	3.02
80	7.02E-05	3.38	4.01E-04	2.96	3.02E-05	3.01	4.28E-05	3.01
160	6.80E-06	3.37	5.05E-05	2.99	3.76E-06	3.01	5.33E-06	3.01
				$\mathbb{P}^2$				
20	8.41E-04	–	3.35E-03	–	3.73E-06	–	5.82E-06	–
40	3.53E-05	4.57	1.65E-04	4.35	9.42E-08	5.31	1.34E-07	5.44
80	8.87E-07	5.32	5.66E-06	4.86	2.48E-09	5.24	3.52E-09	5.26
160	2.02E-08	5.46	1.81E-07	4.97	7.75E-11	5.00	1.10E-10	5.00
				$\mathbb{P}^3$				
20	4.23E-05	–	2.32E-04	–	1.53E-07	–	1.02E-06	–
40	1.88E-06	4.49	8.98E-06	4.69	2.70E-10	9.15	4.00E-10	11.32
80	1.36E-08	7.11	8.72E-08	6.69	1.22E-12	7.79	1.73E-12	7.85
160	7.99E-11	7.41	7.16E-10	6.93	1.59E-14	6.26	2.25E-14	6.27

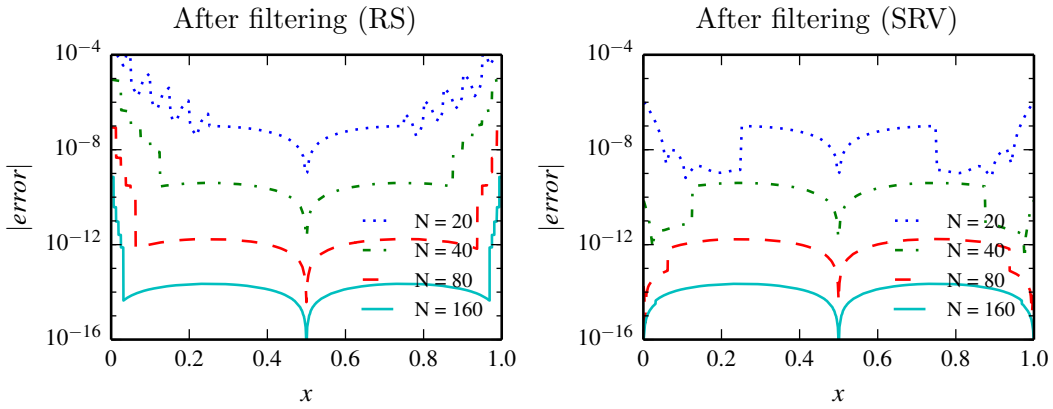


Figure 1.6: Comparison of the point-wise errors in log scale of the filtered solutions with RS filter (1.12) and SRV filter (1.14). The approximation polynomial is $\mathbb{P}^3$.

Construction of SIAC Filter

We remind the reader that the SIAC filter is formulated as

$$K^{(r+1,\ell)}(x) = \sum_{\gamma=0}^{r} c_\gamma^{(r+1,\ell)} \psi^{(\ell)}(x - x_\gamma(\bar{x})),$$

where $x_\gamma(\bar{x}) = -\frac{r}{2} + \gamma + \lambda(\bar{x})$ represent the positions of the filter nodes. Here, $\lambda(\bar{x})$ is defined as a shift function that depends upon the evaluation point $\bar{x}$. However, for convenience we focus on the symmetric filter, $K^{r,\ell}$ with $\lambda(\bar{x}) = 0$. The implementation of one-sides filters is similar.

The components of the filter, central B-splines, which can be constructed using the recursion relation (1.7). In Chapter 2, we will introduce the generalized definition of B-splines, and then we can use the efficient algorithm given by de Boor [31] to construct the filter. Since the B-splines are always the same, one can also calculate the polynomial coefficients, store them and then use polynomial evaluation scheme to evaluate the B-spline at arbitrary points.

The filter coefficients, c_γ, remain to be defined. The coefficients are decided by the property that the filter reproduces polynomials up to degree r, where $r+1$ is the number of B-splines. Using the monomials as in (1.8) we can obtain the following linear system for the filter coefficients:

$$\sum_{\gamma=0}^{r} c_\gamma^{(r+1,\ell)} \int_{-\infty}^{\infty} \psi^{(\ell)}(\xi - x_\gamma)(x-\xi)^m d\xi = x^m, \ m = 0, 1, \ldots, r. \tag{1.17}$$

In order to calculate the integration exactly, we use Gaussian quadrature with $\lceil \frac{l+m+1}{2} \rceil$ quadrature points. As an example for $k = 1$ ($r = 2k, \ell = k+1$), we have

$$\begin{bmatrix} 1 & 1 & 1 \\ x-1 & x & x+1 \\ x^2+2x+\frac{7}{6} & x^2+\frac{1}{6} & x^2-2x+\frac{7}{6} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}. \tag{1.18}$$

Since linear system (1.18) holds for all x, we can simply set $x = 0$ and obtain the coefficients $[c_0, c_1, c_2]^T = [-\frac{1}{12}, \frac{7}{6}, -\frac{1}{12}]^T$. The linear system (1.17) for the coefficients is a non-singular system, so the coefficients are exist and unique, see [13, 25].

Remark 1.3.2. *For one-sided filters, such as the SRV filter, the linear system* (1.17) *will have a large condition number, which causes computational issues. We will discuss these issues in the following chapter.*

After implementation of the filter, we continue by demonstrating how to implement the convolution operator in SIAC filtering.

Evaluation of the Convolution Operator

The basic operation used in SIAC filtering is convolution of the DG solution against a B-spline based filter. Here, we explicitly point out the steps to efficient evaluation of the convolution operator.

In the one-dimensional case, denote $\{I_j\}_{j=1}^N$ be the mesh. To evaluate the filtered solution at a point $x \in I_j$, we have

$$\begin{aligned} u^\star(x) &= \frac{1}{h} \int_{-\infty}^{\infty} K^{(2k+1,k+1)} \left(\frac{x-\xi}{h} \right) u_h(\xi) d\xi \\ &= \frac{1}{h} \int_{x-\frac{3k+1}{2}h}^{x+\frac{3k+1}{2}h} K^{(2k+1,k+1)} \left(\frac{x-\xi}{h} \right) u_h(\xi) d\xi \end{aligned} \tag{1.19}$$

The integration in (1.19) is calculated by Gauss quadrature with $k+1$ quadrature points. However, both the DG solution and the filter are piecewise polynomials. Therefore, we have to divide the support, $supp(K(x)) = \left[x - \frac{3k+1}{2}h, x + \frac{3k+1}{2}h\right]$ into many subintervals, such that both the DG solution and the filter are polynomials on each subinterval. First, consider the discontinuities of the DG solution. We can write (1.19) as

$$u^\star(x) = \frac{1}{h} \sum_{I_{i+j} \cap supp(K(x)) \neq \varnothing} \int_{I_{i+j}} K^{(2k+1,k+1)}\left(\frac{x-\xi}{h}\right) u_h(\xi) d\xi. \tag{1.20}$$

Then we divide the elements I_{i+j} into several subintervals that $I_{i+j} = \bigcup_{\alpha=1}^{n_{i+j}} I_{i+j}^{\alpha}$ according to the breaks of the filter such that on each subinterval I_{i+j}^{α} the filter is a polynomial,

$$\begin{aligned} &\int_{I_{i+j}} K^{(2k+1,k+1)}\left(\frac{x-\xi}{h}\right) u_h(\xi) d\xi \\ &= \sum_{\alpha=1}^{n_{i+j}} \int_{I_{i+j}^{\alpha}} K^{(2k+1,k+1)}\left(\frac{x-\xi}{h}\right) u_h(\xi) d\xi. \end{aligned} \tag{1.21}$$

Finally, we can apply the Gauss quadrature to calculate the integration on subintervals I_{i+j}^{α}. Usually, for uniform meshes we divide each element I_{i+j} into two subintervals, but for nonuniform meshes the number of subintervals is dependent on the mesh. To speed up the filtering process, sometimes it is possible to use inexact integration, see [50]. However, step (1.20) is necessary. The step, which divides the integration region into subintervals according to the filter breaks, is also needed for calculating the linear system (1.17).

In multi-dimensions, the filter is a tensor product of the one-dimensional filters. The implementation of the multi-dimensional SIAC filter over rectangular meshes is the same. For triangular meshes, the principles are the same and one can find the details in [48].

2 Position-Dependent SIAC Filters

2.1 Introduction

When judging the value of numerical methods, the practical usage and computational considerations are always a criteria. As introduced in Chapter 1, due to its symmetric property, the symmetric SIAC filter (1.6) can not be applied near domain boundaries or discontinuities of the exact solution, which is impractical in practice. To overcome this disadvantage, two one-sided filters, the RS filter (1.12) and the SRV filter (1.14), were introduced. However, these still have some deficiencies which are discussed in the following.

2.1.1 The Deficiencies of the RS and SRV Filters

- **Theoretical Considerations**

As mentioned in Chapter 1, the main difference between the RS filter (1.12) and the SRV filter (1.14) is the number of B-splines. In order to reduce the errors of the RS filtered solution, the SRV filter increases the number of B-splines from $2k+1$ to $4k+1$. The strategy of using $4k+1$ B-splines seems work well as in [65] and Example 1.3.6, however, the actual story is more complicated. One can easily find a counterexample that demonstrates using $4k+1$ B-splines makes filtered solutions worse than using only $2k+1$ B-splines for some examples. A simple one of these is the L^2 projection of the wave functions $\sin(2\lambda\pi x)$ over a uniform mesh with N elements. For large λ using the SRV filter leads to a worse result compared to using the RS filter, see Figure 2.1. The details of dealing wave functions are presented in Chapter 6.

To explain this occurrence, one has to check the error estimate of the filtered solutions. First, we write the generalized formula of SIAC filters as

$$K^{(r+1,\ell)}(x) = \sum_{\gamma=0}^{r} c_{\gamma}^{(r+1,\ell)} \psi^{(\ell)}(x - x_{\gamma}(\bar{x})), \tag{2.1}$$

where x_γ depends on the location of the evaluation point $\bar{x}$. Formula (2.1) can be used to represent the symmetric filter (1.6), the RS filter (1.12) and the SRV filter (1.14).

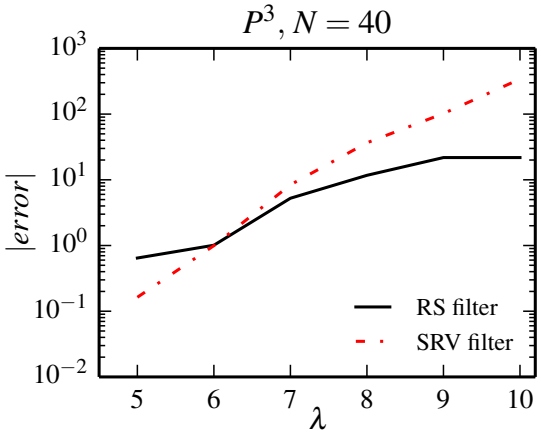


Figure 2.1: Comparison of the RS filtered errors (black) and the SRV filtered errors (red) for the L^2 projection of $\sin(2\pi\lambda x)$ over a uniform mesh.

Similar to the proof of Theorem 1.3.4, for uniform meshes, we have

$$\begin{aligned}\|u - u_h^\star\|_{0,\Omega_0} &\leq \|u - K_h^{(r+1,\ell)} \star u\|_{0,\Omega_0} + \|K_h^{(r+1,\ell)} \star (u - u_h)\|_{0,\Omega_0} \\ &\leq \Theta_1 + \Theta_2,\end{aligned}$$

where

$$\Theta_1 = \|u - K_h^{(r+1,\ell)} \star u\|_{0,\Omega_0} \leq \frac{h^{r+1}}{(r+1)!} C_1 |u|_{r+1}, \quad \text{(Equation (1.8))}$$

and

$$\begin{aligned}\Theta_2 &= C_0 \sum_{|\alpha|\leq\ell} \|D^\alpha K_h^{(r+1,\ell)} \star (u - u_h)\|_{-\ell,\Omega_{1/2}}, \quad \text{(Lemma 1.2.2)} \\ &\leq C_0 C_1 \sum_{|\alpha|\leq\ell} \|\partial_h^\alpha (u - u_h)\|_{-\ell,\Omega_1} \\ &\leq C_1 C_2 h^{2k+1}, \quad \text{(Theorem 1.2.1)}\end{aligned}$$

here $\Omega_0 + supp(K_H^{(r+1,\ell)}) \subset \Omega_{1/2}$ and $\Omega_{1/2} + supp(K_H^{(r+1,\ell)}) \subset \Omega_1$.

Now, we have

$$\|u - u_h^\star\|_{0,\Omega_0} \leq \frac{h^{r+1}}{(r+1)!} C_1 |u|_{r+1} + C_1 C_2 h^{2k+1}, \tag{2.2}$$

where C_2 is a constant related to the DG approximation and

$$C_1 = \sup_{\bar{x}\in\Omega} \kappa(\bar{x}), \quad \text{where } \kappa(\bar{x}) = \sum_{\gamma=0}^{r} |c_\gamma^{(r+1,k+1)}| \tag{2.3}$$

is determined by the filter coefficients. In addition, we note that the filter coefficients are dependent on the location of the evaluation point $\bar{x}$.

One can see that increasing the number of B-splines can increase the order of the first term in (2.2), but it has no effect on the second term. Another important factor is the constant C_1 (or κ), which depends on the filter coefficients. Figure 2.2 shows the values of κ with respect to the location of the evaluation point.

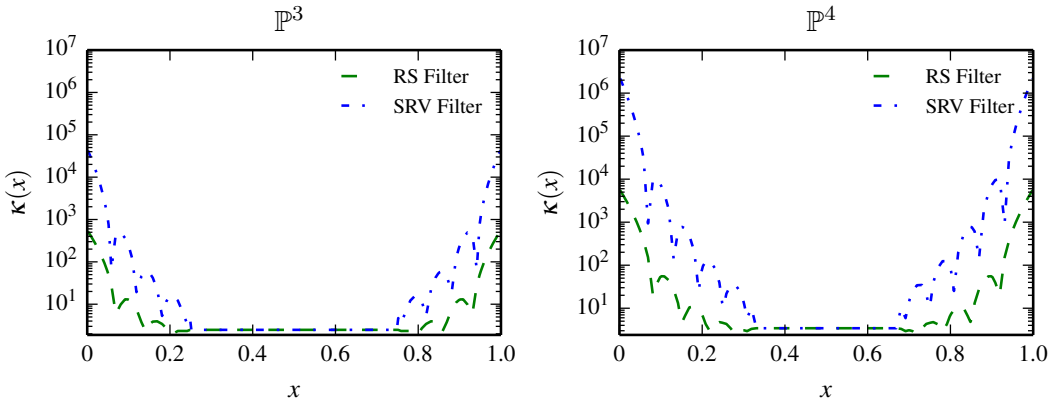


Figure 2.2: $\kappa(x)$ in (2.3) for: the RS filter (1.12) and the SRV filter (1.14) in error estimate (2.2) with respect the location of the evaluation point. Left: $\mathbb{P}^3$ polynomials. Right: $\mathbb{P}^4$ polynomials.

The two components of the above error estimate are the error constant and the accuracy order. Comparing to the RS filter, the SRV filter maintains the same accuracy order and the error constant is significantly increased, which are the theoretical deficiencies of the SRV filter compared to the RS filter. However, if filtering an exact solution that is sufficiently smooth, using the SRV filter leads to a better accuracy than using the RS filter, see [39].

- **Computational Considerations**

In addition to the theoretical estimates, computational considerations are important to consider when applying a technique to real world problems.

First, the SRV filter is constructed with $4k+1$ central B-splines, which increased both the width of the stencil generated and the computational cost (in terms of functions evaluations) a disproportionate amount compared to the symmetric filter. Also, when calculating the filter coefficients using the linear system (1.17), one has to use Gaussian quadrature with $\lceil \frac{5k}{2} + 1 \rceil$ quadrature points.

Second, the SRV filter requires the use of multiple precision (at least quadruple) for $\mathbb{P}^3$ and higher degree polynomials to obtain consistent and meaningful results, which makes it highly unsuitable for practical CPU-based computations and certainly GPU computing. Figure 2.3 shows the significant round-off error near the boundaries when using double precision for filtering the L^2 projection of a sine function. The round-off error is due to the huge filter coefficients $c_\gamma^{(4k+1,k+1)}$, and the enormous condition number of the linear system (1.17).

Third, the numerical performances of the former filters are not satisfactory for nonlinear equations and nonuniform meshes, see numerical examples in Section 2.4.

Lastly, in practical applications, such as streamline integration [68], suggest that the RS filter does not place enough weight at the boundary point. This can lead to dissatisfied results. Since the SRV filter is developed based on the RS filter, it also has the same problem (even worse, see Figures 5.6 and 5.7 in Chapter 5).

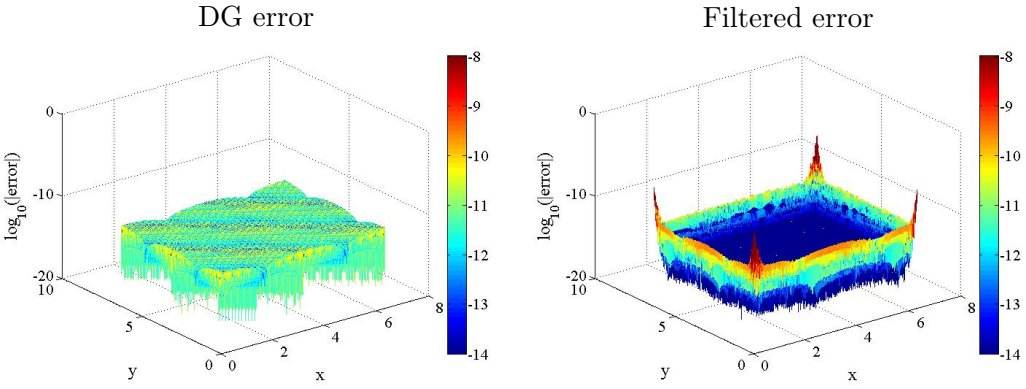


Figure 2.3: The point-wise errors in log scale of the original L^2 projection solution and the SRV filtered solution with polynomial $\mathbb{P}^4$, mesh 80×80. Double precision was used in these computations.

2.2 Modification of Position-Dependent Filter

In order to overcome the principle deficiencies of the former one-sided filters, we have to consider a new position-dependent filter for filtering near boundaries. If we consider the error estimate (2.2), we can see that its components are the error constant C_1 in (2.3) and the accuracy order. Since the accuracy order of the former filters is already optimal, in this chapter, we focus on reducing the value of C_1 to design the new filter. Also, preliminary results suggest that changing the number and position of the B-splines is not enough to overcome the deficiencies. Therefore, to complete the task, we have to add a general B-spline into the central B-spline filter. In this section, we first review the generalized definition of B-splines. Then, we propose a new position-dependent filter that ameliorates the deficiencies of the former filters.

2.2.1 A Review of B-splines

First, we recall the definition of B-splines given by de Boor [31].

Definition 2.2.1 (B-spline)**.**

Let $\mathbf{t} := (t_j)$ *be a nondecreasing sequence of real numbers that create a so-called knot sequence. The* j*th* **B-spline** *of order* ℓ *for the knot sequence* $\mathbf{t}$ *is denoted by* $B_{j,\ell,\mathbf{t}}$

and is defined, for $\ell = 1$*, by the rule*

$$B_{j,1,\mathbf{t}}(x) = \begin{cases} 1, & t_j \leq x < t_{j+1}; \\ 0, & \textit{otherwise.} \end{cases}$$

In particular, $t_j = t_{j+1}$ *leads to* $B_{j,1,\mathbf{t}} = 0$*. For* $\ell > 1$*,*

$$B_{j,\ell,\mathbf{t}}(x) = \omega_{j,k,\mathbf{t}} B_{j,\ell-1,\mathbf{t}} + (1 - \omega_{j+1,\ell,\mathbf{t}}) B_{j+1,\ell-1,\mathbf{t}},$$

with

$$\omega_{j,\ell,\mathbf{t}}(x) = \frac{x - t_j}{t_{j+\ell-1} - t_j}.$$

This notation will be used to create a new filter near the boundaries.

A central B-spline of order ℓ has a knot sequence that is uniformly spaced and symmetrically distributed

$$\mathbf{t} = -\frac{\ell}{2}, -\frac{\ell-2}{2}, \cdots, \frac{\ell-2}{2}, \frac{\ell}{2}.$$

For convenience, we denote $\psi_{\mathbf{t}}^{(\ell)}(x)$ to be the 0^{th} B-spline of order ℓ for the knot sequence $\mathbf{t}$,

$$\psi_{\mathbf{t}}^{(\ell)}(x) = B_{0,\ell,\mathbf{t}}(x).$$

Remark 2.2.1. *The knot sequence* $\mathbf{t}$ *also represents the so-called breaks of the B-spline. The B-spline in the region* $[t_i, t_{i+1})$*,* $i = 0, \ldots, \ell - 1$ *is a polynomial of degree* $\ell - 1$*, but in the entire support* $[t_0, t_\ell]$*, the B-spline is a piecewise polynomial. When the knots* (t_j) *are sampled in a symmetric and equidistant fashion, the B-spline is called a central B-spline. Notice that a central B-spline* (1.7) *is a subset of this definition where the knots are equally-spaced. This new notation provides more flexibility than the previous central B-spline notation.*

2.2.2 New Position-Dependent SIAC Filter

We begin by restating the definition of the SIAC filter through the definition of the knots defining the B-splines used in the filter. We recall that the generalized definition of the filter relied on $r + 1$ *central* B-splines of order ℓ. B-splines were then defined automatically through a knot sequence $\mathbf{t} := (t_j)$. Before we deduce the new boundary filter, we introduce a new definition: *knot matrix.*

Definition 2.2.2 (Knot matrix)**.**

A **knot matrix, T**, *is an* $n \times m$ *matrix such that the* $\gamma-$*th row,* $\mathbf{T}(\gamma)$*, of the matrix* $\mathbf{T}$ *is a knot sequence with* $\ell + 1$ *elements (i.e.,* $m = \ell + 1$*) that are used to create the B-spline* $\psi_{\mathbf{T}(\gamma)}^{(\ell)}(x)$*. The number of rows* n *is specified based on the number of B-splines used to construct the filter.*

For example, the knot matrix for the symmetric filter (1.6) has components given by

$$T(i,j) = -\frac{\ell}{2} + j + i - \frac{r}{2}, \qquad i = 0, \ldots, r; \text{ and } j = 0, \ldots, \ell.$$

More specifically, consider the filter for DG solutions of degree $k = 1$. For the symmetric filter ($\ell = 2$ and $r = 2$), the elements of the knot matrix $\mathbf{T}_{sym}$ are given by

$$\mathbf{T}_{sym} = \begin{pmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{pmatrix}.$$

For the RS filter (1.12), which uses only $2k+1$ central B-splines at the left boundary, the knot matrix $\mathbf{T}_{RS}$ is given by

$$\mathbf{T}_{RS} = \begin{pmatrix} -4 & -3 & -2 \\ -3 & -2 & -1 \\ -2 & -1 & 0 \end{pmatrix}.$$

For the SRV filter (1.14), which uses $4k+1$ central B-spline at the left boundary, the knot matrix $\mathbf{T}_{SRV}$ is given by

$$\mathbf{T}_{SRV} = \begin{pmatrix} -6 & -5 & -4 \\ -5 & -4 & -3 \\ -4 & -3 & -2 \\ -3 & -2 & -1 \\ -2 & -1 & 0 \end{pmatrix}.$$

Therefore, we can use Definition 2.2.2 to rewrite the symmetric filter (1.6) in terms of a knot matrix as follows

$$K^{(2k+1,k+1)}_{\mathbf{T}_{sym}}(x) = \sum_{\gamma=0}^{2k} c_\gamma^{(2k+1,k+1)} \psi^{(k+1)}_{\mathbf{T}_{sym}(\gamma)}(x).$$

Now we can define the new filter by generating a knot matrix.

Definition 2.2.2 alone is not enough to create the boundary filter we wish to propose. We must impose further restrictions on the knot matrix. First, for convenience we require

$$T(\gamma,0) \le T(\gamma,1) \le \cdots \le T(\gamma,\ell), \quad \text{for} \quad \gamma = 0,\dots,r,$$

and

$$T(\gamma+1,0) \le T(\gamma,\ell), \quad \text{for} \quad \gamma = 0,\dots,r-1.$$

Second, the knot matrix, $\mathbf{T}$, should satisfy

$$T(0,0) \ge \frac{\bar{x}-x_R}{h} \quad \text{and} \quad T(r,\ell) \le \frac{\bar{x}-x_L}{h},$$

where h is the element size for a uniform mesh. This requirement is derived from the support of the B-spline as well as the support of the filter needing to remain inside the domain. Recall that the support of the B-spline $\psi^{(\ell)}_{\mathbf{T}(\gamma)}$ is $[T(\gamma,0), T(\gamma,\ell)]$, and the support of the filter is $[T(0,0), T(r,\ell)]$. For any $\bar{x} \in [x_L, x_R]$, the filtered solution at point $\bar{x}$ can then be written as

$$\begin{aligned} u^\star(\bar{x}) = K^{(r+1,\ell)}_{h\mathbf{T}} \star u_h(\bar{x}) &= \int_{-\infty}^{\infty} K^{(r+1,\ell)}_{h\mathbf{T}}(\bar{x}-\xi) u_h(\xi) d\xi \\ &= \int_{\bar{x}-hT(r,\ell)}^{\bar{x}-hT(0,0)} K^{(r+1,\ell)}_{h\mathbf{T}}(\bar{x}-\xi) u_h(\xi) d\xi, \end{aligned}$$

where $h\mathbf{T}$ represents the scaled knot matrix. For the boundary regions, we force the interval $[\bar{x} - hT(r,\ell), \bar{x} - hT(0,0)]$ to remain inside the domain $\Omega = [x_L, x_R]$. This implies that

$$x_L \le \bar{x} - hT(r,\ell), \qquad \bar{x} - hT(0,0) \le x_R,$$

and hence the requirement of $T(0,0) \ge \frac{\bar{x}-x_R}{h}$ and $T(r,\ell) \le \frac{\bar{x}-x_L}{h}$. Finally, we require that the filter remain as symmetric as possible. This means the knots should be chosen as

$$\begin{aligned}
&\text{Left} \ : T \leftarrow T - \left(T(r,\ell) - \frac{\bar{x}-x_L}{h}\right), && \text{for } \frac{\bar{x}-x_L}{h} < \frac{3k+1}{2},\\
&\text{Right} : T \leftarrow T - \left(T(0,0) - \frac{\bar{x}-x_R}{h}\right), && \text{for } \frac{x_R-\bar{x}}{h} < \frac{3k+1}{2}.
\end{aligned}$$

This shifting will increase the error and it is therefore still necessary to increase the number of B-splines used in the filter.

Because the symmetric filter yields superconvergence results, we wish to retain the original form of the filter as much as possible. Near the boundary, where the symmetric filter cannot be applied, we keep the $2k+1$ shifted central B-splines and add only one general B-spline. We keep the notation $r+1 = 2k+1$ associated with the number of central B-splines. To avoid increasing the spatial support of the filter, we will choose the knots of this general B-spline dependent upon the knots of the $2k+1$ central B-splines in the following way: near the left boundary, we let the first $2k+1$ B-splines be central B-splines whereas the last B-spline will be a general spline. The elements of knot matrix are then given by

$$T(i,j) = \begin{cases} -\ell - r + j + i + \frac{\bar{x}-x_L}{h}, & 0 \le i \le 2k, 0 \le j \le \ell;\\ \frac{\bar{x}-x_L}{h} - 1, & i = 2k+1, j = 0;\\ \frac{\bar{x}-x_L}{h}, & i = 2k+1, j = 1,\dots,\ell. \end{cases}$$

The filter coefficients are decided by the linear system (1.17), which reproducing polynomials up to degree $r+1$. For the left one-sided filter with scaling h, we have

$$K_{h\mathbf{T}}^{(r+1,\ell)}(x) = \sum_{\gamma=0}^{r+1} c_\gamma^{(r+1,\ell)} \psi_{h\mathbf{T}(\gamma)}^{(\ell)}(x),$$

where $r+1 = 2k+1$ is the number of central B-splines and $\mathbf{T}(\gamma)$ represents the γ-th row of the knot matrix $\mathbf{T}$. For the central B-splines, $\gamma = 0,\dots,2k$ and

$$\psi_{\mathbf{hT}(\gamma)}^{(\ell)}(x) = \frac{1}{h}\psi_{\mathbf{T}(\gamma)}^{(\ell)}\left(\frac{x}{h}\right).$$

The added B-spline is a monomial defined as

$$\psi_{h\mathbf{T}(r+1)}^{(\ell)}(x) = \frac{1}{h} x_{\mathbf{T}(r+1)}^{\ell-1}\left(\frac{x}{h}\right),$$

where

$$x_{\mathbf{T}(r+1)}^{\ell-1} = \begin{cases} (x - T(r+1,0))^{\ell-1}, & T(r+1,0) \le x \le T(2k+1,\ell);\\ 0, & \text{otherwise.} \end{cases}$$

Therefore near the left boundary, the filter can be rewritten as

$$K_{h\mathbf{T}}^{(r+1,\ell)}(x) = \underbrace{\sum_{\gamma=0}^{r} c_{\gamma}^{(r+1,\ell)} \psi_{\mathbf{hT}(\gamma)}^{(\ell)}(x)}_{r+1=2k+1 \text{ central B-splines}} + \underbrace{c_{r+1}^{(r+1,\ell)} \psi_{h\mathbf{T}(\mathbf{r+1})}^{(\ell)}(x)}_{\text{General B-spline}}. \tag{2.4}$$

Similarly, we can design the new filter near the right boundary, where the general B-spline is given by

$$\psi_{\mathbf{T}(\mathbf{0})}^{(\ell)}(x) = x_{\mathbf{T}(0)}^{\ell-1} = \begin{cases} (T(0,\ell)-x)^{\ell-1}, & T(0,0) \le x \le T(r+1,\ell); \\ 0, & \text{otherwise.} \end{cases}$$

The elements of the knot matrix for the right boundary filter are defined as

$$T(i,j) = \begin{cases} \frac{\bar{x}-x_R}{h}, & i=0, j=0,\dots,\ell-1; \\ \frac{\bar{x}-x_R}{h}+1, & i=0, j=\ell; \\ j+i-1+\frac{\bar{x}-x_R}{h}, & 1 \le i \le r+1, 0 \le j \le \ell, \end{cases}$$

and the form of the filter is then

$$K_{h\mathbf{T}}^{(r+1,\ell)}(x) = c_0^{(r+1,\ell)} \psi_{h\mathbf{T}(\mathbf{0})}^{(\ell)}(x) + \sum_{\gamma=1}^{r+1} c_{\gamma}^{(r+1,\ell)} \psi_{\mathbf{hT}(\gamma)}^{(\ell)}(x).$$

We note that this "extra" B-spline is used only when $\frac{\bar{x}-x_L}{h} < \frac{3k+1}{2}$ or $\frac{x_R-\bar{x}}{h} < \frac{3k+1}{2}$, otherwise the coefficient of the "extra" B-spline becomes zero when solving the linear system (1.17), and then the filter becomes the symmetric central B-spline filter.

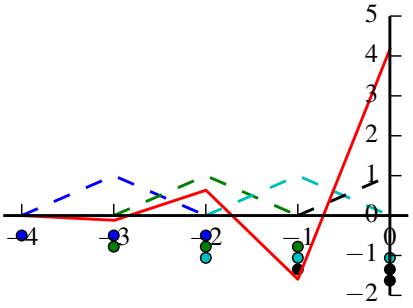
Example 2.2.1. *We present a concrete example for the $\mathbb{P}^1$ case with $\ell = 2$. In this case, the knot matrices for our newly proposed filter at the left and right boundaries are*

$$\mathbf{T}_{Left} = \begin{pmatrix} -4 & -3 & -2 \\ -3 & -2 & -1 \\ -2 & -1 & 0 \\ -1 & 0 & 0 \end{pmatrix}, \qquad \mathbf{T}_{Right} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 2 \\ 1 & 2 & 3 \\ 2 & 3 & 4 \end{pmatrix}.$$

The following plot illustrates how to use the knot matrix to construct the filter. The knots and respective B-spline are in same color, the filter is in red.

$$\mathbf{T}_{Left} = \begin{pmatrix} -4 & -3 & -2 \\ -3 & -2 & -1 \\ -2 & -1 & 0 \\ -1 & 0 & 0 \end{pmatrix},$$

In the left figure, the three equally tributed blue (green, cyan) points re sent the central B-spline in color (green, cyan), and the three black p represent the general B-spline (two m ple points at 0).



These new knot matrices are 4×3 matrices where, in the case of the filter for the left boundary, the first three rows express the knots of the three central B-splines and the last row expresses the knots of the general B-spline. For the filter applied to the right boundary, the first row expresses the knots of the general B-spline and the last three rows express the knots of the central B-splines.

Comparing the new knot matrix with the one used to obtain the SRV filter, we can see that they have the same number of columns, which indicates that they use the same order of B-splines. There are fewer rows in the new matrix $(2k + 2)$ than the number of rows from the SRV filter $(4k + 1)$. This indicates that the new filter uses fewer B-splines than the SRV filter.

To compare all existing one-sided filters, we plot these filters used at the left boundary for $k = 2$. Figure 2.4 illustrates that the new position-dependent SIAC filter places more weight on the evaluation point than the former filters, and the SRV filter has a significantly larger magnitude and support which we observed to cause problems, especially for higher-order polynomials (such as $\mathbb{P}^3$ or $\mathbb{P}^4$). For this example, using the filter for quadratic approximations, the scaling of the SRV filter has a range from -150 to 150 versus -5 to 5 for the newly proposed filter.

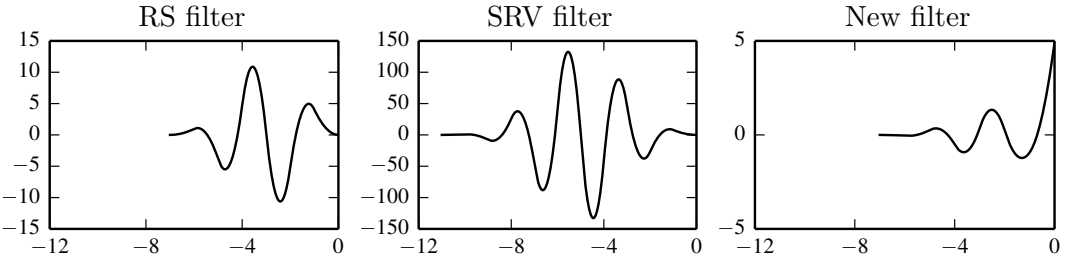


Figure 2.4: (Left) RS filter (1.12), (Center) SRV filter (1.14) and (Right) the newly proposed filter with $k = 2$. The filtered point is at boundary $x = 0$.

2.3 Theoretical Results

The previous section introduced a new filter to reduce the errors of dG approximations while attempting to ameliorate the issues concerning the former filters. In this section, we discuss the theoretical results of the newly defined boundary filter.

2.3.1 Local Error Estimate in the Negative Order Norm

First of all, we point out there is a minor flaw in the theoretical foundation of one-sided filters. The error estimate in the negative order norm, given in Theorem 1.2.1, assumes periodic boundary conditions. It follows that the error estimate of the SRV filter given in [39] is under the same periodic boundary assumption. The periodic boundary assumption is unnatural for one-sided filters since with the periodic boundary assumption we can use the symmetric filter directly. To ameliorate this minor

flaw, we present an alternative error estimate of the DG solution in the negative order norm.

Lemma 2.3.1. *Let u be the exact solution of a linear hyperbolic equation* (1.1)*, and let u_h be the DG approximation. Then the negative order norm estimate of $u - u_h$ satisfies*

$$\|(u - u_h)(T)\|_{-(k+1),\Omega} \leq Ch^{2k+1}. \tag{2.5}$$

Note: comparing to Theorem 1.2.1, the periodic boundary condition was removed.

Proof. First we give a dual problem of equation (1.1) by

$$\varphi_t + \sum_{i=1}^{d} a_i \varphi_{x_i} - a_0 \varphi = 0, \qquad \varphi(\mathbf{x}, T) = \Phi(\mathbf{x}).$$

The DG approximation satisfies scheme (1.2)

$$\begin{aligned}((u_h)_t, v_h)_K - \sum_{i=1}^{d} (a_i u_h, (v_h)_{x_i})_K \\ + \sum_{i=1}^{d} \int_{\partial K} a_i \hat{u}_h v_h n_i ds + (a_0 u_h, v_h) = 0,\end{aligned}$$

By applying the dual problem, we obtain

$$\frac{d}{dt}(u, \varphi)_K = -\sum_{i=1}^{d} \int_{\partial K} a_i u \varphi n_i ds.$$

and

$$(u, \varphi)_K(T) = (u, \varphi)_K(0) - \int_0^T \left(\sum_{i=1}^{d} \int_{\partial K} a_i u \varphi n_i ds \right) dt.$$

Note: the original proof in [25] assumed periodic boundary conditions. Then the term $\int_0^T \left(\sum_{i=1}^{d} \int_{\partial K} a_i u \varphi n_i ds \right) dt$ is counteracted by summing up for all $K \in \mathcal{T}_h$ in [25]. Without assuming the periodic boundary conditions the term remains in the analysis. Then we have

$$\begin{aligned}&((u - u_h)(T), \Phi)_K = (u - u_h, \varphi)(T) \\ =&(u - u_h, \varphi)_K(0) - \int_0^T \left(\sum_{i=1}^{d} \int_{\partial K} a_i u \varphi n_i ds \right) dt - \int_0^T \frac{d}{dt}(u_h, \varphi)_K dt.\end{aligned}$$

Considering $\frac{d}{dt}(u_h, \varphi)_K$ in the third term,

$$\begin{aligned}
&\frac{d}{dt}(u_h, \varphi)_K = ((u_h)_t, \varphi)_K + (u_h, \varphi_t)_K, \\
&= ((u_h)_t, \varphi - v_h)_K + ((u_h)_t, v_h)_K + (u_h, \varphi_t)_K \qquad (v_h \in V_h^k) \\
&= ((u_h)_t, \varphi - v_h)_K + (u_h, \sum_{i=1}^{d} a_i (v_h)_{x_i})_K \\
&\quad - \sum_{i=1}^{d} \int_{\partial K} a_i \hat{u}_h v_h n_i ds - (a_0 u_h, v_h)_K + (u_h, \varphi_t)_K \\
&= ((u_h)_t + a_0 u_h, \varphi - v_h)_K \\
&\quad - \left(u_h, \sum_{i=1}^{d} a_i (\varphi - v_h)_{x_i} \right)_K - \sum_{i=1}^{d} \int_{\partial K} a_i \hat{u}_h v_h n_i ds, \\
&= \left((u_h)_t + \sum_{i=1}^{d} a_i (u_h)_{x_i} + a_0 u_h, \varphi - v_h \right)_K \\
&\quad - \sum_{i=1}^{d} \int_{\partial K} a_i u_h (\varphi - v_h) n_i ds - \sum_{i=1}^{d} \int_{\partial K} a_i \hat{u}_h v_h n_i ds, \\
&= - \sum_{i=1}^{d} \int_{\partial K} a_i u_h (\varphi - v_h) n_i ds - \sum_{i=1}^{d} \int_{\partial K} a_i \hat{u}_h v_h n_i ds,
\end{aligned}$$

substituting above formula back, we have

$$\begin{aligned}
&((u - u_h)(T), \Phi)_K \\
&= (u - u_h, \varphi)_K(0) - \int_0^T \left(\sum_{i=1}^{d} \int_{\partial K} a_i u \varphi n_i ds \right) dt \\
&\quad + \int_0^T \left(\sum_{i=1}^{d} \int_{\partial K} a_i \left(u_h (\varphi - v_h) n_i + \hat{u}_h v_h n_i \right) ds \right) dt \\
&= (u - u_h, \varphi)_K(0) - \int_0^T \left(\sum_{i=1}^{d} \int_{\partial K} a_i (u - \hat{u}_h) \varphi n_i ds \right) dt \\
&\quad + \int_0^T \left(\sum_{i=1}^{d} \int_{\partial K} a_i (u_h - \hat{u}_h)(\varphi - v_h) n_i ds \right) dt
\end{aligned}$$

Since the first and the third term are identical to the proof in [25], we need only consider the second term. According to [4, 15], the DG solution has superconvergence property for its numerical flux

$$|\sum_{i=1}^{d} \int_{\partial K} (u - \hat{u}_h)| \leq C h^{2k+1},$$

which does not rely on the periodic boundary condition. Then, for the second term, we have

$$\int_0^T \left(\sum_{i=1}^{d} \int_{\partial K} a_i (u - \hat{u}_h) \varphi n_i ds \right) dt \leq \int_0^T C h^{2k+1} \|\varphi\|_{k+1,\Omega} dt.$$

Summing up over all $K \in \mathcal{T}_h$, we obtain (2.5). □

Remark 2.3.1. *Compared to the original theorem, Lemma 2.3.1 theoretically confirms that the periodic boundary conditions are not necessary. It reveals the fact that the filter has compact support and needs only local information of the DG approximation. The error estimates for divided differences of the DG approximation $\partial_h^\alpha u_h$ are similar, and we will address the details in Chapter 4.*

2.3.2 Theoretical Results in the Uniform Case

First, we discuss the theoretical results of the new one-sided filter for uniform meshes. Specifically, for $k = 1$ it is globally superconvergent of order three. For higher degree polynomials, it is possible to obtain superconvergence only in the interior of the domain.

Recall that the new scaled filter has the form

$$K_{h\mathbf{T}}^{(r+1,\ell)}(x) = \sum_{\gamma=0}^{r+1} c_\gamma^{(r+1,\ell)} \psi_{h\mathbf{T}(\gamma)}^{(\ell)}(x).$$

In the interior of the domain the symmetric filter is used. It consists of $2k+1$ central B-splines,

$$K_{h\mathbf{T}}^{(2k+1,\ell)}(x) = \sum_{\gamma=0}^{2k} c_\gamma^{(2k+1,\ell)} \psi_{\mathbf{hT}(\gamma)}^{(\ell)}(x),$$

and, near the left boundary the new one-sided filter can be written as

$$K_{h\mathbf{T}}^{(2k+1,\ell)}(x) = \left(\sum_{\gamma=0}^{2k} c_\gamma^{(2k+1,\ell)} \psi_{h\mathbf{T}(\gamma)}^{(\ell)}(x) \right) + c_{2k+1}^{(2k+1,\ell)} \psi_{h\mathbf{T}(2k+1)}^{(\ell)}(x),$$

where $2k+1$ central B-splines are used together with one general B-spline. The scaled filter $K_{h\mathbf{T}}^{(r+1,\ell)}$ has the property that the convolution $K_{h\mathbf{T}}^{(r+1,\ell)} \star u_h$ only uses information inside the domain Ω.

Theorem 2.3.2. *Under the same conditions in Theorem 1.3.4, let $u_h^\star(\bar{\mathbf{x}}) = (K_{h\mathbf{T}}^{(r+1,\ell)} \star u_h)(\bar{\mathbf{x}})$ be the solution obtained by applying the newly proposed filter which uses $r+1 = 2k+1$ central B-splines of order $\ell = k+1$ and one general B-spline in boundary regions. Then the filtered solution has the following properties:*

(i) $\|(u - u_h^\star)(\bar{\mathbf{x}})\|_{0,\Omega} \leq C h^3$ *for $k = 1$. That is, $u_h^\star(\bar{\mathbf{x}})$ is globally superconvergent of order three for linear approximations.*

(ii) $\|(u - u_h^\star)(\bar{\mathbf{x}})\|_{0,\Omega \setminus supp\{K_s\}} \leq C h^{r+1}$ *when $r+1 \leq 2k+1$ central B-splines are used in the filter. Here $supp\{K_s\}$ represents the support of the symmetric filter. Thus, $u_h^\star(\bar{\mathbf{x}})$ is superconvergent in the interior of the domain.*

(iii) $\|(u - u_h^\star)(\bar{\mathbf{x}})\|_{0,\Omega} \leq Ch^{k+1}$ *globally.*

Proof. We neglect the proof of properties (i) and (ii) as they are similar to the proofs in [25] and [39]. Also, we note that the periodic boundary conditions in the origin of proofs can be removed due to Lemma 2.3.1.

Consider the one-dimensional case ($d = 1$). Then the error can be written as

$$\|u - K_{h\mathbf{T}}^{(r+1,\ell)} \star u_h\|_{0,\Omega} \leq \Theta_{h,1} + \Theta_{h,2},$$

where

$$\Theta_{h,1} = \|u - K_{h\mathbf{T}}^{(r+1,\ell)} \star u\|_{0,\Omega} \quad \text{and} \quad \Theta_{h,2} = \|K_{h\mathbf{T}}^{(r+1,\ell)} \star (u - u_h)\|_{0,\Omega}.$$

The proof of higher order convergence for the first term, $\Theta_{H,1}$, is the same as in [25] as the requirement on $K_{h\mathbf{T}}$ does not change (reproduction polynomials of degree $r + 1$). This means that

$$\Theta_{h,1} \leq \frac{h^{r+1}}{(r+1)!} C_1 |u|_{r+1,\Omega}.$$

Now consider the second term, $\Theta_{h,2}$. Without loss of generality, we consider the filter for the left boundary in order to estimate $\Theta_{h,2}$. The proofs for the filter in the interior and right boundary are similar. We use the form of the filter given in (2.4), which decomposes the new filter into two parts: $2k + 1$ central B-splines and one general B-spline. That is, we write

$$K_{h\mathbf{T}}^{(r+1,\ell)}(x) = \underbrace{\left(\sum_{\gamma=0}^{r} c_\gamma^{(r+1,\ell)} \psi_{h\mathbf{T}(\gamma)}^{(\ell)}(x) \right)}_{\text{central B-splines}} + \underbrace{c_{r+1}^{(r+1,\ell)} \psi_{h\mathbf{T}(r+1)}^{(\ell)}(x)}_{\text{general B-spline}}.$$

Setting $e(x) = u(x) - u_h(x)$, then

$$\Theta_{h,2} = \|K_{h\mathbf{T}}^{(r+1,\ell)} \star e\|_{0,\Omega_1} \leq \|K_{h\mathbf{T}}^{(r+1,\ell)}\|_{L^1} \|e\|_0 \leq \sup_{x\in\Omega} (\kappa(x)) \, \|e\|_0.$$

where $\kappa(x) = \sum_{\gamma=0}^{r} |c_\gamma^{(r+1,\ell)}| + \frac{|c_{2k+1}^{(r+1,\ell)}}{\ell}|$. Hence

$$\Theta_{h,2} \leq \mathsf{C} \sup_{x\in\Omega} (\kappa(x)) \, h^{k+1}.$$

Remark 2.3.2. *Note that in this analysis we steered away from the negative order norm argument. Technically, the terms involving the central B-splines have a convergence rate of $r + 1 \leq 2k + 1$ as given in [25, 39]. It is the new addition, the term involving the general B-spline that presents the limitation and reduces the convergence rate to that of the dG approximation itself.*

To extend this to the multidimensional case ($d > 1$), given an arbitrary $\mathbf{x} = (x_1, \ldots, x_d) \in \mathbb{R}^d$, we set

$$\psi_{\mathbf{T}(\gamma)}^{(\ell)}(\mathbf{x}) = \prod_{i=1}^{d} \psi_{\mathbf{T}(\gamma)}^{(\ell)}(x_i).$$

The filter for the multidimensional space considered is of the form

$$K_{h\mathbf{T}}^{(r+1,\ell)}(\mathbf{x}) = \sum_{\gamma=0}^{r+1} \mathbf{c}_{\gamma}^{(r+1,\ell)} \psi_{h\mathbf{T}(\gamma)}^{(\ell)}(\mathbf{x}),$$

where the coefficients $\mathbf{c}_{\gamma}^{(\ell)}$ are tensor products of the one-dimensional coefficients. To emphasize the general B-spline used near the boundary, we assume, without loss of generality, that in the $x_{k_1}, \ldots, x_{k_{d_0}}$ directions we need the general B-spline, where $0 \leq d_0 \leq d$. Then

$$\psi_{h\mathbf{T}(2k+1)}^{(\ell)} = \prod_{i=1}^{d_0} \psi_{h\mathbf{T}(2k+1)}^{(\ell)}(x_{k_i}).$$

By applying the same idea we used for the one-dimensional case, the theorem is also true for multi-dimensional case. □

We note that the constant in the final estimate is a product of two other constants, one of them is decided by the filter, $\sum_{\gamma=0}^{r} |c_{\gamma}^{(r+1,\ell)}| + \frac{|c_{r+1}^{(r+1,\ell)}|}{\ell}$, and the other one is decided by the DG approximation. To further illustrate the necessity of examining the constant in the error term which contributed to the filter, we provide Figure 2.5. This figure demonstrates the difference between $\sum_{\gamma=0}^{r} |c_{\gamma}^{(r+1,\ell)}|$ for the previously introduced filters and our new filter in which the constant is modified to $\sum_{\gamma=0}^{r} |c_{\gamma}^{(r+1,\ell)}| + \frac{|c_{r+1}^{(r+1,\ell)}|}{\ell}$. In Figure 2.5, one can clearly see that by adding a general spline to the $r+1$ central B-splines we are able to reduce the constant in the error term significantly.

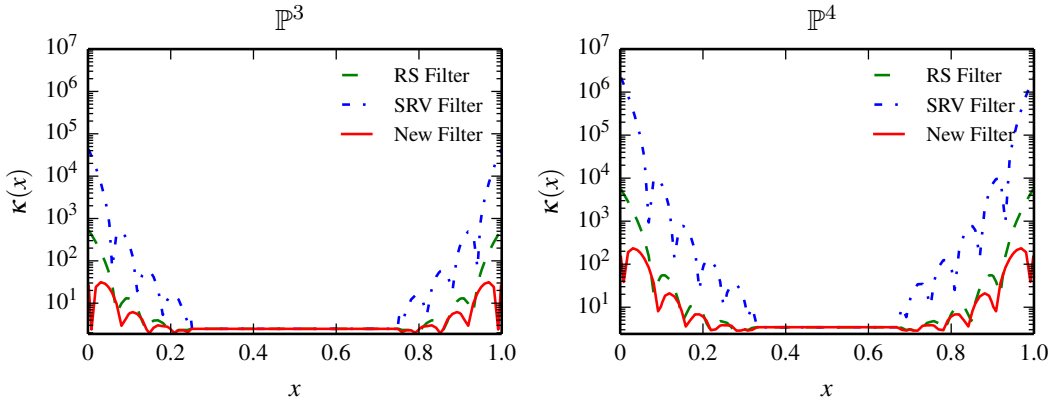


Figure 2.5: Plots demonstrating the effect of the coefficients on the error estimate for $\mathbb{P}^3$ and $\mathbb{P}^4$ polynomials. Shown is $\kappa(x)$ for: the RS filter, the SRV fitler and the new filter.

2.3.3 Theoretical Results in the Nonuniform Case

In this section, we give a theoretical interpretation to the computational results presented in [30]. This is done by using the newly proposed filter for nonuniform meshes and showing that the new position-dependent filter maintains the superconvergence property in the interior of the domain for smoothly-varying meshes and is accuracy order conserving near the boundaries for nonuniform meshes. We begin by defining what we mean by a smoothly-varying mesh.

Definition 2.3.1 (Smoothly-Varying Mesh)**.**
Let ξ be a function defined over a uniform mesh on domain $\Omega \subset \mathbb{R}$, then a smoothly-varying mesh defined over Ω is a nonuniform mesh whose variable x satisfies

$$x = \xi + f(\xi), \tag{2.6}$$

where f is a sufficiently smooth function and satisfies

$$f'(\xi) > -1, \qquad \xi \in \partial\Omega \Longleftrightarrow \xi + f(\xi) \in \partial\Omega.$$

For example, we can choose $f(\xi) = 0.5\sin(\xi)$ over $[0, 2\pi]$. The multi-dimensional definition can be defined in the same way.

Lemma 2.3.3. *Under the same conditions in Theorem 1.2.1, denote ξ to be the variable for the uniform mesh defined on Ω with size h, and x be the variable of a smoothly-varying mesh defined in (2.6). Let $u_h(\xi)$ be the numerical solution to linear hyperbolic equation (1.1) over uniform mesh ξ, and $u_h(x)$ be the approximation over smoothly-varying mesh x, both of them obtained by using the discontinuous Galerkin scheme (1.2). Then the filtered solution obtained by applying SIAC filter $K_h(\xi)$ for $u_h(\xi)$ and $K_H(x)$ for $u_h(x)$ with a proper scaling H, are related by*

$$\|u(x) - K_H \star u_h(x)\|_{0,\Omega} \leq C\|u(\xi) - K_h \star u_h(\xi)\|_{0,\Omega}.$$

Here, the filter K can be any filter we mentioned in the previous section (symmetric filter, RS filter, SRV filter, and newly proposed position-dependent filter).

This lemma will be important for demonstrating superconvergence for smoothly-varying meshes.

Proof. The proof is straightforward. If the scaling H is properly chosen, a simple mapping can be done from the smoothly-varying mesh to the corresponding uniform mesh. The result holds if the Jacobian is bounded (from the definition of smoothly-varying mesh).

$$\begin{aligned}
\|u(x) - K_H \star u_h(x)\|^2_{0,\Omega} &= \int_\Omega \left(u(x) - K_H \star u_h(x)\right)^2 dx \\
&\overset{x\to\xi}{=} \int_{\tilde{\Omega}} \left(u(\xi) - u_h^\star(\xi)\right)^2 (1 + f'(\xi)) d\xi \\
&\leq \|u(\xi) - K_h \star u_h(\xi)\|^2_{0,\tilde{\Omega}} \cdot \max|1 + f'(\xi)|.
\end{aligned}$$

According to the definition of smoothly-varying mesh, $\Omega = \tilde{\Omega}$, we have

$$\|u(x) - K_H \star u_h(x)\|_{0,\Omega} \leq C\|u(\xi) - K_h \star u_h(\xi)\|_{0,\Omega},$$

where $C = \left(\max_{\Omega} |1 + f'|\right)^{\frac{1}{2}}$. □

Remark 2.3.3. *The proof seems obvious, but it is important to choose a proper scaling for H in the computations. Due to the smoothness and computational cost requirements, we need to keep H constant when treating points within the same element. Under this condition, the best choice is $H = \Delta x_j$ when post-processing the element I_j. It is now easy to see that there exists a c in the element I_j, such that*

$$H = \Delta x_j = h(1 + f'(c)).$$

Theorem 2.3.4. *Under the same conditions in Theorem 2.3.2. Let $u_h^\star(\bar{\mathbf{x}}) = (K_{h\mathbf{T}}^{(r+1,\ell)} \star u_h)(\bar{\mathbf{x}})$ be the solution obtained by applying our newly proposed filter which uses $r+1 = 2k+1$ central B-splines of order $\ell = k+1$ and one general B-spline in boundary regions. Then the filtered solution has the following properties:*
For smoothly-varying meshes *(Definition 2.3.1),*

(i) $\|(u - u_h^\star)(\bar{\mathbf{x}})\|_{0,\Omega} \leq Ch^3$ *for $k = 1$. That is, $u_h^\star(\bar{\mathbf{x}})$ is globally superconvergent of order three for linear approximations.*

(ii) $\|(u - u_h^\star)(\bar{\mathbf{x}})\|_{0,\Omega\setminus supp\{K_s\}} \leq Ch^{r+1}$ *when $r + 1 \leq 2k + 1$ central B-splines are used in the filter. Here $supp\{K_s\}$ represents the support of the symmetric filter. Thus, $u_h^\star(\bar{\mathbf{x}})$ is superconvergent in the interior of the domain.*

For nonuniform meshes,

(iii) $\|(u - u_h^\star)(\bar{\mathbf{x}})\|_{0,\Omega} \leq C\ h^{k+1}$ *globally.*

Proof. Lemma 2.3.3 allows us to use the result over uniform meshes for smoothly-varying meshes as well, then we know the properties (i) and (ii) are true by Theorem 2.3.2. For property (iii), we can use the same proof in Theorem 2.3.2 since the proof does not depend on the mesh type. □

Remark 2.3.4. *For nonuniform meshes, all one-sided filters have the same accuracy order in boundary regions.*

We have now shown that superconvergence can be achieved for interior solutions over smoothly-varying meshes. In the following section, we present numerical results that confirm these findings on uniform and nonuniform (smoothly-varying) meshes.

2.4 Numerical Results

The previous section introduced a new position-dependent filter by adding a general B-spline to a modified central B-spline filter. The addition of a general B-spline helps to maintain a consistent support size for the filter throughout the domain and eliminates

the need for a multi-precision package. Also, due to Property 1.3.2 of the symmetric filter, we know that when the evaluation point moves gradually from the boundary to the interior region, the new one-sided filter smoothly transforms to the symmetric filter. This section illustrates the performance of the new position-dependent SIAC filter on uniform and nonuniform (smoothly-varying and random) meshes.

We compare the results to the SRV filter [65]. In order to provide a fair comparison between the SRV and new filters, we mainly show the results using quadruple precision for mostly one-dimensional cases. We also provide one result using double precision to show that quadruple precision is necessary(unnecessary) to use the SRV(new) filter. Furthermore, in order to reduce the computational cost of the filter that uses $4k+1$ central B-splines, we neglect to implement the convex combination described in (1.16). This convex combination is not necessary for the new filter, and it has no effect on the accuracy.

Remark 2.4.1. *The SRV filter requires using quadruple precision in the computations to eliminate round-off error, which is more computationally expensive than using double precision. The new filter only requires double precision. In order to give a fair comparison between the SRV filter and the new filter, for the one-dimensional examples we have used quadruple precision to maintain a consistent computational environment. However, for the two-dimensional examples we have used double precision since quadruple precision is too expensive (approximately ten times slower) for multi-dimensions.*

2.4.1 Uniform Meshes

Linear hyperbolic equation

The first example we consider is a linear hyperbolic equation,

$$\begin{aligned} u_t + u_x &= 0, \qquad (x,t) \in [0,1] \times (0,T] \\ u(x,0) &= \sin(2\pi x), \qquad x \in [0,1]. \end{aligned} \tag{2.7}$$

with $T = 1$ over uniform meshes. The exact solution is a translation of the sine function, $u(x,t) = \sin(2\pi(x-t))$.

The DG solution error and the position-dependent SIAC filtered (SRV and new filter) error are shown in Table 2.1 for both quadruple precision and double precision. Using quadruple precision, both filters reduce the errors in filtered solution, although the new filter has only a minor reduction in the quality of the error. However, using double precision only the new filter can maintain this error reduction for $\mathbb{P}^3$ and $\mathbb{P}^4$polynomials. We note that we concentrate on the results for $\mathbb{P}^3$ and $\mathbb{P}^4$ polynomials as there is no noticeable difference between double and quadruple precision for $\mathbb{P}^1$ and $\mathbb{P}^2$ polynomials in the one-dimension.

The point-wise error plots are given in Figures 2.6. When using quadruple precision, the SRV filter can reduce the error of the DG solution better than the new filter for sufficiently fine meshes. However, it uses $2k-1$ more B-splines than the newly generated filter. This difference is noticeable when using double precision, which is almost ten times faster than using quadruple precision for $\mathbb{P}^3$ and $\mathbb{P}^4$. For such examples

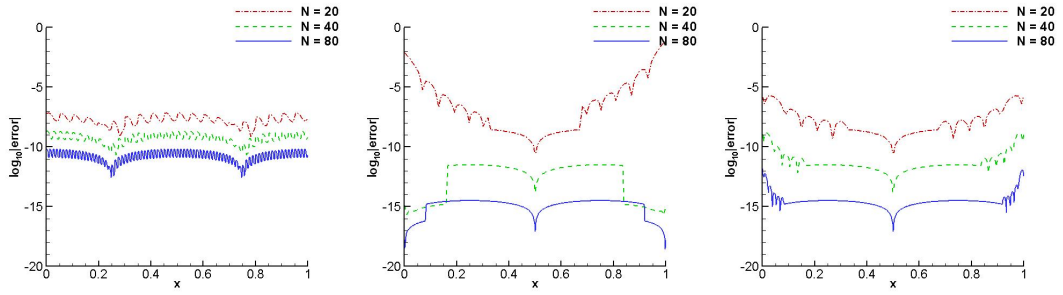
the new filter performs better both computationally and numerically (in terms of error). Table 2.1 shows that the former filter can only reduce the error for fine meshes when using $\mathbb{P}^4$ piecewise polynomials. The new filter performs as good as when using quadruple precision and reduces the error magnitude at a reduced computational cost.

Additionally, we point out that the accuracy of the SRV filter depends on (1) having higher regularity of $\mathcal{C}^{4k+1}$, (2) a well-resolved DG solution, and (3) a wide enough support (at least $5k+1$ elements). The same phenomenon will also be observed in the following tests such as for a nonlinear equation. For the new filter, the support size remains the same throughout the domain – $3k+1$ elements – and a higher degree of regularity is not necessary.

Note: In the following examples, all one-dimensional examples are using quadruple precision due to the SRV filter, and all two-dimensional examples are using double precision due to the computational cost.

DG SRV Filter New filter

Quadruple precision:



Double precision:

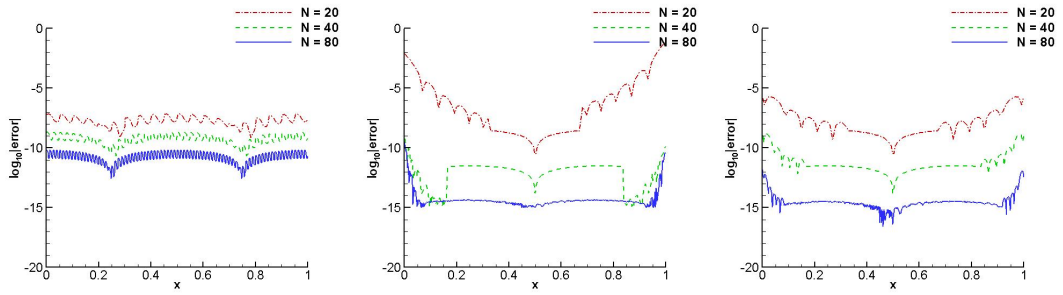


Figure 2.6: Comparison of the point-wise errors of the DG solution, the SRV filter and the new filter for linear hyperbolic equation (2.7) over uniform meshes with polynomials $\mathbb{P}^4$.

Table 2.1: L^2- and $L^\infty-$errors for the DG approximation together with the SRV and new filters for linear hyperbolic equation (2.7) over uniform meshes.

	DG				SRV Filter				New Filter			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
Quadruple precision												
$\mathbb{P}^1$												
20	4.02E-03	–	1.45E-02	–	1.98E-03	–	2.80E-03	–	1.98E-03	–	2.80E-03	–
40	1.02E-03	1.97	3.82E-03	1.92	2.44E-04	3.02	3.46E-04	3.02	2.44E-04	3.02	3.46E-04	3.02
80	2.58E-04	1.99	9.79E-04	1.96	3.02E-05	3.01	4.28E-05	3.01	3.03E-05	3.01	4.28E-05	3.01
$\mathbb{P}^2$												
20	1.07E-04	–	3.67E-04	–	3.73E-06	–	5.82E-06	–	1.21E-05	–	8.27E-05	–
40	1.34E-05	3.00	4.62E-05	2.99	9.42E-08	5.31	1.34E-07	5.44	5.52E-07	4.45	5.31E-06	3.96
80	1.67E-06	3.00	5.78E-06	3.00	2.48E-09	5.24	3.52E-09	5.26	4.79E-08	3.53	6.19E-07	3.10
$\mathbb{P}^3$												
20	2.06E-06	–	6.04E-06	–	1.53E-07	–	1.02E-06	–	2.30E-06	–	8.71E-06	–
40	1.29E-07	4.00	3.80E-07	3.99	2.70E-10	9.15	4.00E-10	11.32	4.14E-09	9.12	2.27E-08	8.58
80	8.07E-09	4.00	2.38E-08	4.00	1.22E-12	7.79	1.73E-12	7.85	8.18E-12	8.98	1.20E-10	7.56
$\mathbb{P}^4$												
20	3.19E-08	–	7.02E-08	–	7.53E-03	–	7.33E-02	–	5.31E-07	–	1.99E-06	–
40	1.00E-09	4.99	2.25E-09	4.97	1.99E-12	31.82	3.12E-12	34.45	2.97E-10	10.80	1.58E-09	10.30
80	3.14E-11	5.00	7.14E-11	4.98	2.23E-15	9.80	3.19E-15	9.93	1.37E-13	11.08	1.55E-12	9.99
Double precision												
$\mathbb{P}^3$												
20	2.06E-06	–	6.04E-06	–	1.53E-07	–	1.02E-06	–	2.30E-06	–	8.71E-06	–
40	1.29E-07	4.00	3.80E-07	3.99	2.70E-10	9.15	4.00E-10	11.32	4.14E-09	9.12	2.27E-08	8.58
80	8.07E-09	4.00	2.38E-08	4.00	1.25E-12	7.75	3.85E-12	6.70	8.18E-12	8.98	1.20E-10	7.56
$\mathbb{P}^4$												
20	3.19E-08	–	7.02E-08	–	7.53E-03	–	7.33E-02	–	5.31E-07	–	1.99E-06	–
40	1.00E-09	4.99	2.25E-09	4.97	3.97E-11	27.50	6.14E-10	26.83	2.97E-10	10.80	1.58E-09	10.30
80	3.14E-11	5.00	7.14E-11	4.98	1.48E-11	1.42	3.28E-10	0.90	1.37E-13	11.08	1.55E-12	9.99

For the two-dimensional version linear hyperbolic equation,

$$\begin{aligned} u_t + u_x + u_y = 0, \quad (x, y) \in [0, 2\pi] \times [0, 2\pi], \\ u(x, y, 0) = \sin(x + y), \end{aligned} \tag{2.8}$$

at $T = 2\pi$, due to the computational cost issue to obtain the filtered solution, we only calculate the 2D results using double precision. Table 2.2 shows that the accuracy is affected by the round-off error, especially for the SRV filter. Such significant round-off error appears to destroy the accuracy. Although the error magnitude near the boundaries is larger than the regions where the symmetric filter is used, the new filter reduces the error and improves smoothness of the DG solution, see Figure 2.7.

Table 2.2: L^2- and $L^\infty-$errors for the DG approximation together with the SRV and new filters for 2D linear hyperbolic equation (2.8) using polynomials of degree $k = 3, 4$. Double precision was used in the computations.

	DG				SRV Filter				New Filter			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\mathbb{P}^3$												
20×20	3.30E-06	–	1.21E-05	–	2.60E-07	–	1.12E-04	–	2.39E-06	–	1.80E-05	–
40×40	2.06E-07	4.00	7.60E-06	3.99	4.69E-10	9.11	3.11E-09	5.17	7.01E-09	8.41	5.11E-08	8.46
80×80	1.29E-08	4.00	4.76E-08	4.00	1.74E-11	4.75	5.50E-09	-0.82	7.97E-11	6.46	1.02E-09	5.65
$\mathbb{P}^4$												
20×20	4.71E-08	–	1.41E-07	–	2.77E-08	–	1.35E-06	–	5.25E-07	–	3.77E-06	–
40×40	1.46E-09	5.01	4.50E-09	4.97	2.55E-08	0.12	2.84E-06	-1.07	3.83E-10	10.42	3.40E-09	10.11
80×80	4.44E-11	5.04	1.43E-10	4.98	2.73E-08	-0.10	7.86E-06	-1.47	3.00E-13	10.31	3.12E-12	10.09

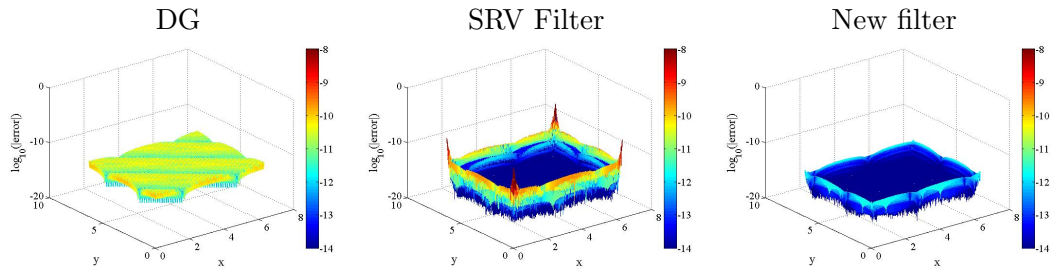


Figure 2.7: Comparison of the point-wise errors of the DG solution, the SRV filter and the new filter for linear hyperbolic equation (2.8) over uniform meshes with polynomials $\mathbb{P}^4$.

Nonlinear Hyperbolic Equation

In the previous linear examples, we can see that both the SRV and new filter are able to reduce the errors of the original DG solutions. Also, we notice that the SRV filter has better performance than the new filter if we do not consider computational issues. However, we point out that the SRV filter is not suitable for dealing with nonlinear equations because its theoretical foundation heavily replies on the linearity. It is also one motivation in developing the new filter. To illustrate the statement, we consider a nonlinear conservation law

$$\begin{aligned} u_t + (e^u)_x &= 0, \quad x \in [0, 2\pi], \\ u(x, 0) &= \sin(x), \end{aligned} \tag{2.9}$$

with $T = 0.2$. Note that due to the flux e^u, the exact solution of this nonlinear problem does not contain any shocks, and the theoretical analysis holds for the solution. In this example, the SRV filter no longer performs well, even under quadruple-precision, see Figure 2.8 and Table 2.3. We can see that the performance of the new filter near the boundaries is better than the SRV filter. However, we point out that for this problem the maximum error does not happen in the boundary regions, see Figure 2.8. This is an issue we will address in future.

Table 2.3: L^2- and $L^\infty-$errors for the DG approximation together with the SRV and new filters for a conservation law with an exponential flux, equation (2.9) using polynomials of degree $k = 3, 4$.

	DG				SRV Filter				New Filter			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
					$\mathbb{P}^3$							
20	1.27E-05	–	1.27E-04	–	9.99E-01	–	9.64E+00	–	2.14E-04	–	6.68E-04	–
40	7.35E-07	4.11	7.86E-06	4.01	6.46E-06	17.24	7.91E-05	16.90	3.15E-06	6.09	1.51E-05	5.47
80	4.58E-08	4.00	5.51E-07	3.83	2.38E-08	8.09	1.27E-07	9.28	2.38E-08	7.05	1.27E-07	6.89
					$\mathbb{P}^4$							
20	7.77E-07	–	5.12E-06	–	2.02E+01	–	1.44E+02	–	4.63E-03	–	2.43E-02	–
40	3.71E-08	4.39	3.04E-07	4.07	3.67E-03	12.42	5.08E-02	11.47	1.94E-06	11.22	9.30E-06	11.35
80	1.30E-09	4.84	1.35E-08	4.50	6.63E-09	19.08	3.65E-08	20.41	6.63E-09	8.19	3.65E-08	7.99

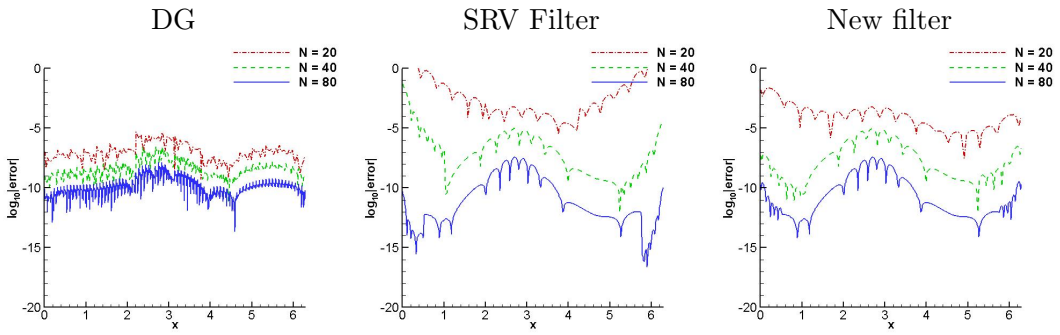


Figure 2.8: Comparison of the point-wise errors of the DG solution, the SRV filter and the new filter for a conservation law with an exponential flux over uniform meshes with polynomials $\mathbb{P}^4$.

SIAC Filtering for Shocks

Although the theorem for the SIAC filter has been established for smooth solutions, it is interesting to investigate the application of SIAC filter to solutions which contain shocks. To demonstrate the possibilites of the filtered results for such problems, we present the following three examples: First, a discontinuous coefficient equation; second, a one-dimensional Burgers equation after the shock has developed; and lastly, the double Mach reflection problem of the two-dimensional Euler equations.

- **Variable coefficient equation with stationary shocks**

  In this example, we consider a variable coefficient equation,

$$u_t + (au)_x = f, \quad (x,t) \in [0,1] \times (0,T] \tag{2.10}$$

with $T = 2\pi$. Here, to create discontinuities we consider a discontinuous variable coefficient

$$a(x) := \begin{cases} \frac{1}{2}, & x \in [-\frac{1}{2}, \frac{1}{2}], \\ 1, & \text{otherwise}, \end{cases}$$

and $f(x,t) = 0$. The following initial condition was also chosen:

$$u(x,0) := \begin{cases} -2\cos(4\pi x), & x \in [-\frac{1}{2}, \frac{1}{2}], \\ \cos(2\pi x), & \text{otherwise}, \end{cases}$$

with $T = 1$. The exact solution is given by

$$u(x,t) := \begin{cases} -2\cos(4\pi(x - \frac{1}{2}t)), & x \in [-\frac{1}{2}, \frac{1}{2}], \\ \cos(2\pi(x-t)), & \text{otherwise}, \end{cases}$$

which has two stationary shocks.

To use the SIAC filter, we have to consider both the discontinuities and the boundaries. Similar to the boundary problem, the one-sided filters can also be applied near the discontinuities using the same implementation strategy as used near a boundary.

Therefore, we divide the domain into three parts $[-1,-\frac{1}{2}],[-\frac{1}{2},\frac{1}{2}]$ and $[\frac{1}{2},1]$, then apply the filter. If the support of the filter is larger than the distance between two boundaries/discontinuities, we need to change the filter scaling, H, as in [65]. Such reduction of scaling can have negative consequences on the final results as noted in [43]. Since the two stationary shocks are located at the element interfaces of the DG solution, the DG solution maintains to keep its $k+1$ accuracy order and the errors are in Table 2.4. Figure 2.9 shows the point-wise error plots. Note that the results are not as good as in the previous tests due the support size of the filters (even the new filter) is still quite large for those three subintervals. However, after refining the mesh the accuracy does improve. This is similar to the observations in [43]. Comparing the results between the SRV and new filter, the new filter has a better performance when the distance between two boundaries/discontinuities is small.

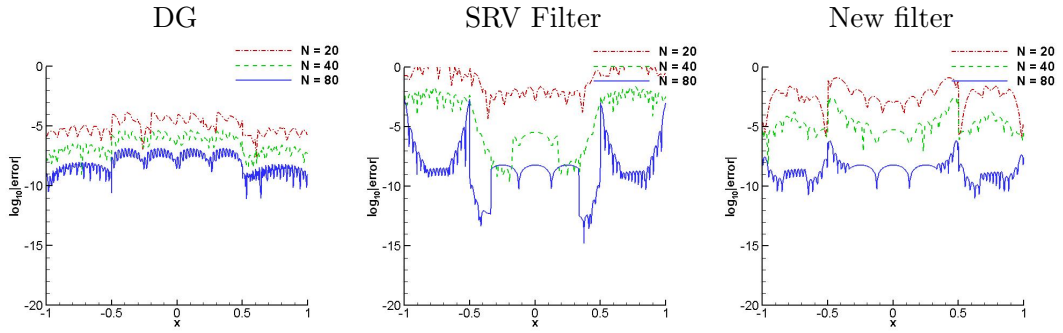


Figure 2.9: Comparison of the point-wise errors of the DG solution, the SRV filter and the new filter for discontinuous coefficient equation (2.10) over uniform meshes with polynomials $\mathbb{P}^4$.

Table 2.4: L^2- and $L^\infty-$errors for the DG approximation together with the SRV and new filters for the discontinuous coefficient equation (2.10) using polynomials of degree $k=3,4$.

	DG				SRV Filter				New Filter			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\mathcal{P}^3$												
20	7.39E-04	–	3.12E-03	–	1.97E-01	–	7.75E-01	–	7.34E-02	–	2.70E-01	–
40	4.66E-05	3.99	1.92E-04	4.02	1.95E-03	6.66	1.59E-02	5.61	6.50E-04	6.82	3.47E-03	6.28
80	2.92E-06	4.00	1.24E-05	3.95	5.09E-05	5.26	7.21E-04	4.46	6.44E-07	9.98	5.95E-06	9.19
$\mathcal{P}^4$												
20	4.59E-05	–	1.44E-04	–	3.18E+00	–	1.39E+01	–	3.74E-02	–	1.35E-01	–
40	1.45E-06	4.99	4.66E-06	4.95	5.83E-03	9.09	2.46E-02	9.14	6.16E-04	5.92	3.26E-03	5.38
80	4.54E-08	5.00	1.45E-07	5.01	1.70E-04	5.10	2.11E-03	3.54	9.35E-08	12.69	6.89E-07	12.21

Since the previous examples have already demonstrated superiority of the new filter, we use only the new filter for the following shock problems.

- **1D Burgers equation with a shock**

In this example, we consider the Burgers equation

$$\begin{aligned} u_t + uu_x &= 0, \quad (x,t) \in [0, 2\pi] \times (0, T), \\ u(x,0) &= \sin(x), \end{aligned} \tag{2.11}$$

with $T = 1$. Note that the solution contains a shock at $x = \pi$. We have implemented the symmetric filter in smooth regions and the boundary filter in the elements next to the boundaries and shocks. No filter is implemented in the element that contains the shock. The results for the point-wise errors are presented in Figure 2.10.

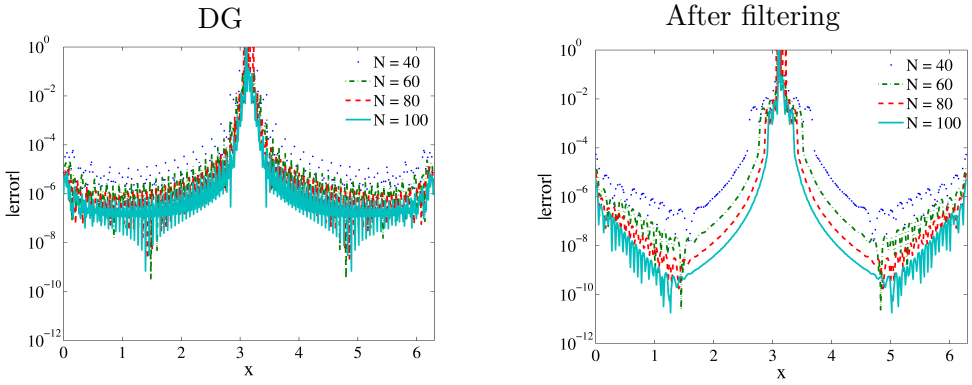


Figure 2.10: Comparison of the point-wise errors in log scale of the DG solutions and the filtered solutions (with the new filter) for Burgers equation (2.11), with approximation polynomial is $\mathbb{P}^2$.

- **Mach reflection problem of the 2D Euler equations**

In this example we apply the new filter to the two dimensional Euler equations for the double Mach reflection problem. We use the multiwavelet troubled cell indicator of Vuik [66] and plot the results for a zoomed in region of the solution in Figure 2.11. Note that from the results given for previous examples we expect that the difference when we examine the two solutions will be small. However, we do observe some reduced oscillations with the filtered solution.

2.4.2 Smoothly-Varying and Nonuniform Meshes

We emphasize that this is the first time that the position-dependent filters have been tested on nonuniform meshes. Due to Theorem 2.3.4, we using the SRV and new filter with scaling of $H = \Delta x_j$.

Remark 2.4.2. *According to [30] we also tested scalings of $H = \max \Delta x_j$, but the scaling $H = \Delta x_j$ provides better accuracy in boundary regions. Since the motivation of this chapter is to focus on boundaries, we only present results with scaling $H = \Delta x_j$. However, we point out that the errors produced using a scaling of $H = \max \Delta x_j$ are quite similar and often produce smoother errors in the interior of the domain for smoothly-varying meshes.*

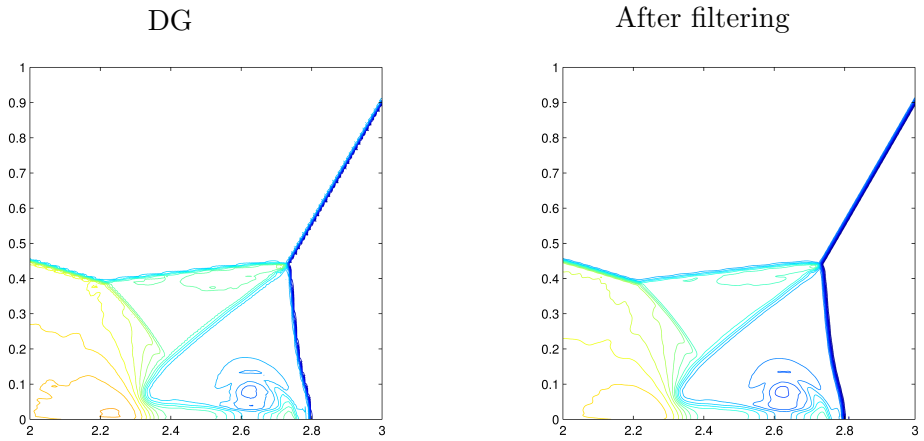


Figure 2.11: Results for the DG approximation and the filtered solution (with the new filter) for the double Mach reflection problem.

We begin by defining three nonuniform meshes that are used in the following examples:

Mesh 2.4.1. *Smoothly-Varying Mesh with Periodicity. The first mesh is a simple smoothly-varying mesh. It is defined by* $x = \xi + b\sin(\xi)$, *where* ξ *is a uniform mesh variable and* $b = 0.5$ *as in [30]. We note that the tests were also performed for different values of* b*; similar results were attained in all cases. This mesh has the nice feature that it is a periodic mesh and that the elements near the boundaries have a larger element size.*

Mesh 2.4.2. *Smooth Polynomial Mesh. The second mesh is also a smoothly-varying mesh but does not have a periodic structure. It is defined by* $x = \xi - 0.05(\xi - 2\pi)\xi$. *For this mesh, the size of elements gradually decreases from left to right.*

Mesh 2.4.3. *Randomly-Varying Mesh. The third mesh is a mesh with randomly distributed elements. The element size varies between* $[0.8h, 1.2h]$, *where* h *is the uniform mesh size.*

We will now present numerical results demonstrating the usefulness of the SRV filter and the new one-sided filter for the aforementioned meshes. First, we consider the same linear problems (2.7) and (2.8) used before. Now the DG solutions are calculated over the three different nonuniform meshes: Mesh 2.4.1, Mesh 2.4.2 and Mesh 2.4.3. The SRV and new filters are applied at the final time.

The point-wise error plots for the periodically smoothly varying mesh are given in Figure 2.12 with the corresponding errors presented in Table 2.5. In the boundary regions, the SRV filter behaves slightly better for coarse meshes than the new filter. However, we recall that this filter essentially doubles the support in the boundary

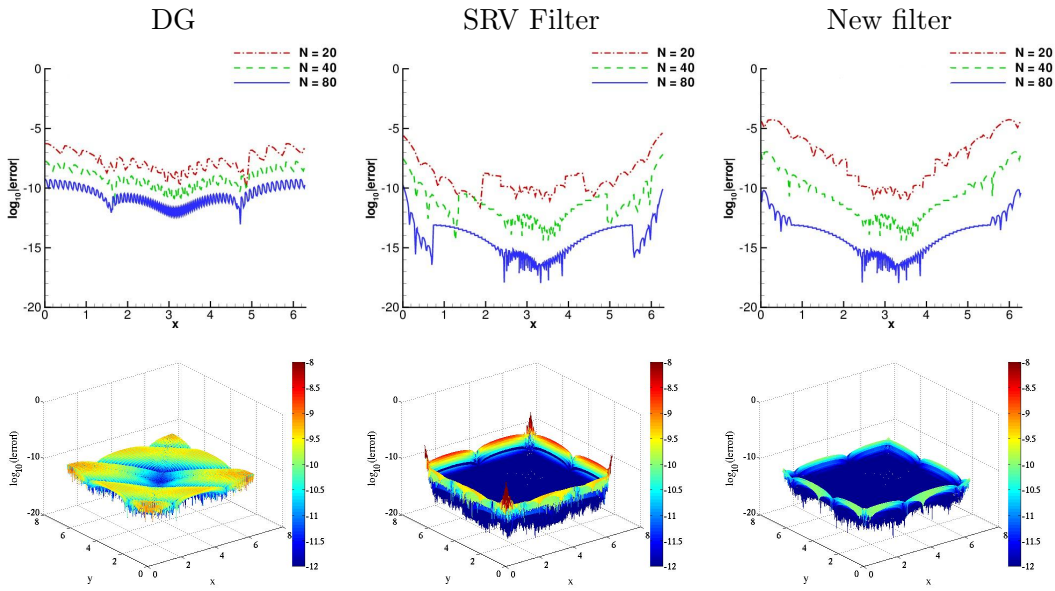


Figure 2.12: Comparison of the point-wise errors of the DG solution, the SRV filter and the new filter for linear hyperbolic equations (1D and 2D) over Mesh 2.4.1 with polynomials $\mathbb{P}^4$. The 2D mesh has 80×80 elements.

regions. Additionally, we see that the new filter has a higher convergence rate than $k+1$ which is better than the theoretically predicted convergence rate.

For the smooth polynomial mesh, Mesh 2.4.2 (without a periodic property), the results of using the scaling of $H = \Delta x_j$ are presented in Figure 2.13 and Table 2.5. Unlike the previous example, without the periodic property, the SRV filter leads to significantly worse performance. The SRV filter no longer enhances the accuracy order and has larger errors near the boundaries. On the other hand, the new filter still improves accuracy when the mesh is sufficiently refined ($N = 40$). Numerically the new filter obtains higher accuracy order than $k+1$. For higher order polynomials, $\mathcal{P}^3$ and $\mathcal{P}^4$, we see that it achieves accuracy order of $2k+1$, but this is not theoretically guaranteed.

Lastly, the filters were applied to DG solutions over a randomly distributed mesh, Mesh 2.4.3. For this randomly-varying mesh, the new filter again reduces the errors except for a very coarse mesh, see Table 2.5. The accuracy order is decreasing compared to smoothly-varying mesh examples, but it is still higher than $k+1$. Unlike smoothly-varying meshes, there are more oscillations in the errors if we compare Figure 2.14 with Figure 2.12. However, the oscillations are still reduced compared to the DG solutions. Note that the SRV filter does not improve the errors from the DG solution at all, and the errors even become worse than the original. This suggests that the SRV filter may be only suitable for uniform meshes.

For the 2D case, the nonuniform meshes we consider are rectangular grids, which the tessellations on $x-$ and $y-$ directions are generated by the same way as the Mesh

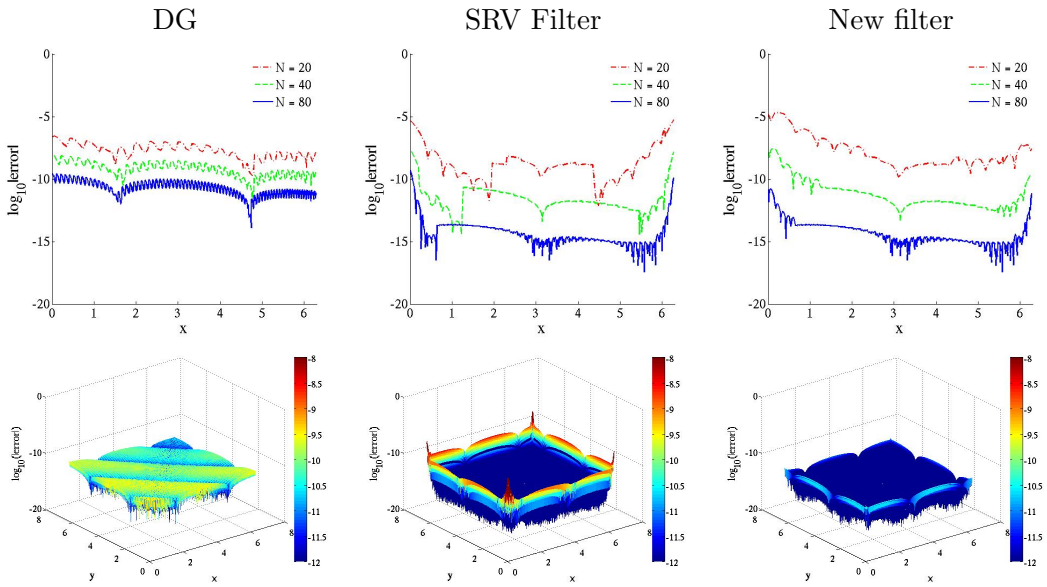


Figure 2.13: Comparison of the point-wise errors of the DG solution, the SRV filter and the new filter for linear hyperbolic equations (1D and 2D) over Mesh 2.4.2 with polynomials $\mathbb{P}^4$. The 2D mesh has 80×80 elements.

2.4.1, Mesh 2.4.2 and Mesh 2.4.3. Unlike the one-dimensional example, the results of the SRV filter are significantly affected by the round-off error, especially near the four corners of the grids. This round-off error completely destroys the accuracy and smoothness near the boundaries. Compared to the SRV filter, the new filter performs much better. In the following examples, we can clearly see the improvement of the accuracy and smoothness compared to the original DG approximations. From all the tests we performed, it is easy to see that the new filter is more suitable than the SRV filter over nonuniform meshes, and the practical performance of the new filter is better than the theoretical prediction.

For Mesh 2.4.1, because of the periodicity, the SRV filter seems slightly better in the L^2 norm than the new filter with polynomial $\mathbb{P}^3$, but worse with polynomial $\mathbb{P}^4$. However, if we look at the L^∞ norm, we can see the new filter still behaves better than the SRV filter, see Table 2.6. We notice that for the $\mathbb{P}^4$ case, even the ideal periodic property can not hide the fact that the SRV filter is not suitable for nonuniform meshes – the SRV filter is worse than the new filter and even the original dG solution. In Figure 2.12, the round-off error of the SRV filter is noticeably demonstrated. The new filter has better accuracy compared to the DG solution when the mesh is sufficiently refined.

Unlike mesh 2.4.1, Mesh 2.4.2 and Mesh 2.4.3 do not have the nice periodic property which is exactly where a one-sided filter is needed. The deficiencies of the SRV filter become significant. The results near the boundaries are worse than the original dG solution, see Figures 2.13 and 2.14. However, the new filter is still able to reduce errors. When the meshes are sufficiently fine, the filtered results with the new filter improve

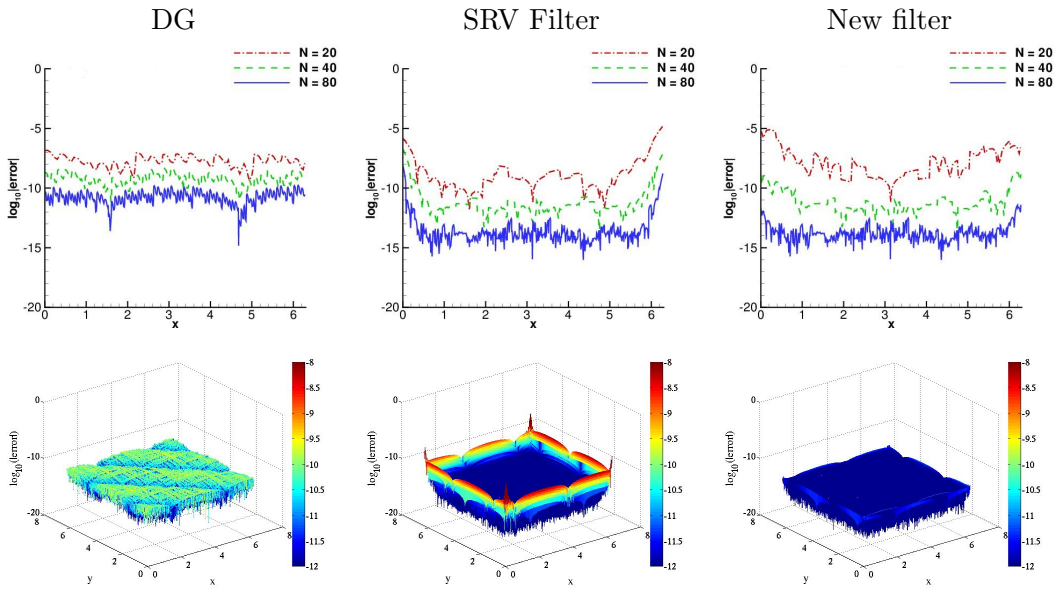


Figure 2.14: Comparison of the point-wise errors of the DG solution, the SRV filter and the new filter for linear hyperbolic equations (1D and 2D) over Mesh 2.4.3 with polynomials $\mathbb{P}^4$. The 2D mesh has 80×80 elements.

the magnitude and smoothness of the DG error. In the high degree case, such as $\mathbb{P}^4$, the results numerically show at least $2k+1$ accuracy order, see Table 2.6. According to this, we can see that the extra B-spline may damage the superconvergence property theoretically, but in a higher degree case the damage is negligible if we compare it to the benefits.

Variable coefficient equation

In this example, we consider a variable coefficient equation

$$\begin{aligned} &u_t + (au)_x = f, \quad x \in [0, 2\pi] \times (0, T] \\ &a(x,t) = 2 + \sin(x+t), \\ &u(x,0) = \sin(x), \end{aligned} \tag{2.12}$$

at $T = 2\pi$. Similar to the previous constant coefficient equation (2.7), we also test this variable coefficient equation (2.12) over three different nonuniform meshes (Mesh 2.4.1, Mesh 2.4.2 and Mesh 2.4.3). Since the results are similar to the previous linear equation examples, here we do not re-describe the detail of the results. We only note that the results of variable coefficient equation (2.12) have more wiggles than the constant coefficient equation. This may be an important issue in extending these ideas to nonlinear equations. Figure 2.15 shows the point-wise error plots for the DG and post-processed approximations over a smoothly-varying mesh. The corresponding

Table 2.5: L^2- and $L^\infty-$errors for the DG approximation together with the SRV and the new filters for linear equation (2.7) over three meshes 2.4.1,2.4.2,2.4.3. A scaling of $H = \Delta x_j$ along with quadruple precision was used in the computations.

	dG				Former Filter				New Filter			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
Mesh 2.4.1					$\mathbb{P}^3$							
20	5.45E-06	–	1.87E-05	–	6.43E-06	–	5.51E-05	–	6.36E-05	–	2.02E-04	–
40	3.39E-07	4.01	1.20E-06	3.96	3.11E-07	4.37	3.25E-06	4.09	1.72E-07	8.53	7.62E-07	8.05
80	2.12E-08	4.00	7.48E-08	4.01	1.45E-10	11.06	1.81E-09	10.81	2.81E-10	9.26	2.16E-09	8.46
					$\mathbb{P}^4$							
20	1.56E-07	–	5.20E-07	–	5.15E-07	–	4.11E-06	–	1.72E-05	–	5.41E-05	–
40	4.83E-09	5.01	1.66E-08	4.97	6.43E-09	6.32	6.56E-08	5.97	2.56E-08	9.39	1.12E-07	8.91
80	1.51E-10	5.00	5.22E-10	4.99	1.25E-11	9.01	1.80E-10	8.51	1.15E-11	11.13	7.77E-11	10.50
Mesh 2.4.2					$\mathbb{P}^3$							
20	3.15E-06	–	1.25E-05	–	1.52E-05	–	1.75E-04	–	1.53E-05	–	7.35E-05	–
40	1.96E-07	4.01	8.05E-07	3.96	9.80E-08	7.27	1.50E-06	6.86	3.50E-08	8.77	2.34E-07	8.29
80	1.22E-08	4.00	4.97E-08	4.02	2.31E-09	5.40	3.77E-08	5.32	5.63E-11	9.28	8.26E-10	8.15
					$\mathbb{P}^4$							
20	6.25E-08	–	2.67E-07	–	6.79E-07	–	5.38E-06	–	4.45E-06	–	2.13E-05	–
40	1.96E-09	5.00	8.77E-09	4.93	2.13E-09	8.32	2.34E-08	7.85	4.12E-09	10.08	2.76E-08	9.59
80	6.14E-11	5.00	2.79E-10	4.97	3.03E-11	6.13	5.01E-10	5.54	1.74E-12	11.21	1.59E-11	10.76
Mesh 2.4.3					$\mathbb{P}^3$							
20	2.49E-06	–	1.06E-05	–	3.35E-05	–	3.61E-04	–	5.64E-06	–	2.90E-05	–
40	1.55E-07	4.00	7.46E-07	3.83	7.42E-07	5.50	9.11E-06	5.31	6.23E-09	9.82	3.80E-08	9.58
80	1.02E-08	3.93	4.67E-08	4.00	2.46E-08	4.91	4.57E-07	4.32	1.54E-10	5.34	8.71E-10	5.45
					$\mathbb{P}^4$							
20	4.03E-08	–	1.49E-07	–	1.40E-06	–	1.47E-05	–	1.52E-06	–	7.85E-06	–
40	1.37E-09	4.88	5.25E-09	4.83	1.42E-08	6.63	1.78E-07	6.36	3.20E-10	12.21	1.98E-09	11.95
80	4.40E-11	4.96	1.70E-10	4.95	4.03E-10	5.13	7.93E-09	4.49	3.68E-13	9.77	3.95E-12	8.97

errors are given in Table 2.7. The results are similar to the linear examples, the two filters perform similarly, with the new filter being more computationally efficient.

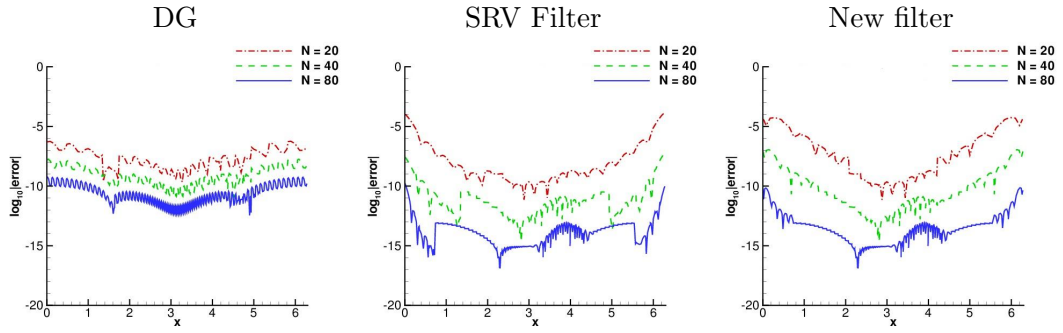


Figure 2.15: Comparison of the point-wise errors of the DG solution, the SRV filter and the new filter for variable coefficient equation (2.12) over Mesh 2.4.1 with polynomials $\mathbb{P}^4$.

For the smooth polynomial mesh 2.4.2, we show the point-wise error plots in Figure 2.16. The corresponding errors are given in Table 2.7. In this example we see that the new filter behaves better at the boundaries than the SRV filter. This may be due to

Table 2.6: L^2- and $L^\infty-$errors for the DG approximation together with the SRV and new filters for 2D linear equation (2.8) using polynomials of degree $k = 3, 4$ over three meshes: Mesh 2.4.1, Mesh 2.4.2 and Mesh 2.4.3.

	DG				SRV Filter				New Filter			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
Mesh 2.4.1					$\mathbb{P}^3$							
20×20	8.74E-06	–	5.39E-05	–	6.94E-06	–	1.10E-04	–	6.71E-05	–	4.04E-04	–
40×40	5.45E-07	4.00	3.39E-06	4.00	3.68E-07	4.24	6.49E-06	4.08	2.09E-07	8.33	1.66E-06	7.93
80×80	3.40E-08	4.00	2.06E-07	4.03	1.50E-10	11.76	9.01E-09	9.49	7.33E-10	8.16	7.76E-09	8.92
					$\mathbb{P}^4$							
20×20	1.93E-07	–	1.05E-06	–	9.25E-07	–	8.56E-06	–	3.26E-05	–	1.12E-04	–
40×40	6.00E-09	5.01	3.32E-08	4.98	3.38E-08	4.77	4.17E-06	1.04	2.67E-08	10.25	2.31E-07	8.92
80×80	1.88E-10	5.00	1.04E-09	5.00	2.07E-08	0.71	9.13E-06	-1.13	1.90E-11	10.46	1.61E-10	10.49
Mesh 2.4.2					$\mathbb{P}^3$							
20×20	4.56E-06	–	3.03E-05	–	1.55E-05	–	3.49E-04	–	1.59E-05	–	1.38E-04	–
40×40	2.85E-07	4.00	1.92E-06	3.98	1.23E-07	6.98	3.04E-06	6.84	4.67E-08	8.41	5.14E-07	8.67
80×80	1.78E-08	4.00	1.20E-07	4.00	3.35E-09	5.20	8.01E-08	5.25	2.43E-10	7.59	4.61E-09	6.80
					$\mathbb{P}^4$							
20×20	8.48E-08	–	3.27E-07	–	1.38E-06	–	1.48E-05	–	5.92E-06	–	3.27E-05	–
40×40	2.65E-09	5.00	1.74E-08	4.92	3.21E-08	5.43	4.99E-06	1.57	4.65E-09	10.30	5.79E-08	9.14
80×80	8.31E-11	5.00	5.58E-10	4.96	2.51E-08	0.35	6.88E-06	-0.46	3.29E-12	10.46	3.49E-11	10.27
Mesh 2.4.3					$\mathbb{P}^3$							
20×20	3.47E-06	–	2.16E-05	–	3.46E-05	–	6.43E-04	–	3.90E-06	–	3.83E-05	–
40×40	2.23E-07	3.96	1.52E-06	3.83	1.90E-06	4.19	3.59E-05	4.16	1.28E-08	8.25	1.25E-07	8.26
80×80	1.41E-08	3.98	9.65E-08	3.98	9.94E-08	4.26	2.71E-06	3.73	2.97E-10	5.43	3.88E-09	5.01
					$\mathbb{P}^4$							
20×20	5.83E-08	–	2.84E-07	–	3.06E-06	–	5.19E-05	–	1.06E-06	–	8.87E-05	–
40×40	1.90E-09	4.94	1.04E-08	4.77	2.64E-08	6.86	2.64E-06	4.30	7.80E-10	10.41	1.01E-08	9.78
80×80	6.06E-11	4.97	3.48E-10	4.90	1.46E-08	0.85	7.09E-06	-1.43	6.60E-13	10.20	7.85E-12	10.33

the more compact filter support size.

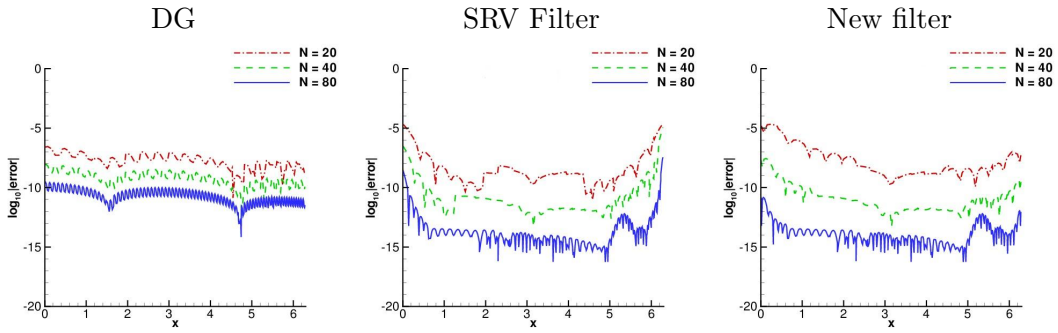


Figure 2.16: Comparison of the point-wise errors of the DG solution, the SRV filter and the new filter for variable coefficient equation (2.12) over Mesh 2.4.2 with polynomials $\mathbb{P}^4$.

Finally, we test the variable coefficient equation (2.12) over a randomly-varying mesh 2.4.3. Similar to the linear examples, the point-wise errors plots, Figure 2.17, show more oscillations than smoothly-varying mesh examples. We again see the new filter has better errors at the boundaries than the SRV filter.

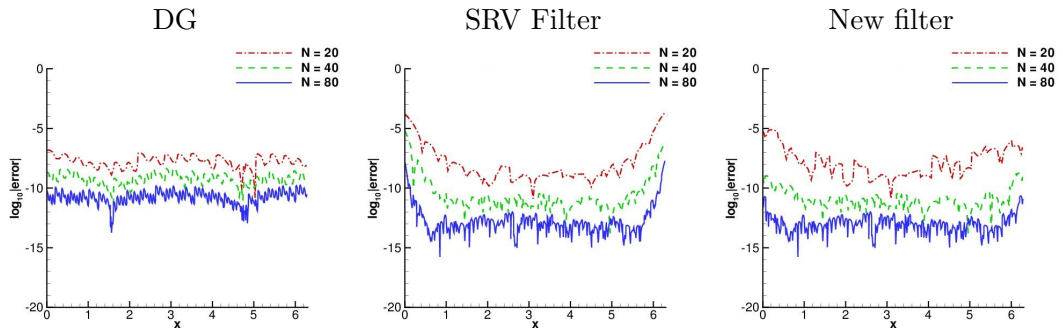


Figure 2.17: Comparison of the point-wise errors of the DG solution, the SRV filter and the new filter for variable coefficient equation (2.12) over Mesh 2.4.3 with polynomials $\mathbb{P}^4$.

2.5 Conclusion

In this chapter, we have proposed a new position-dependent SIAC filter, which can be applied to discontinuous Galerkin approximations for uniform and nonuniform meshes. The new filter was devised as a consequence of analyzing the constant in the previous error estimates and the practical requirement, such as streamline integration in Chapter 5. This filter was created by introducing an extra general B-spline to a filter consisting of $2k+1$ central B-splines. This strategy allows us to overcome shortcomings of the former one-sided filters: we can now reliably use double-precision to both produce and use our filter, and our new filter has a smaller geometric footprint and hence costs less (in terms of operations) to evaluate.

We have, for the first time, proved the accuracy order conserving nature of the SIAC filter globally and shown that this boundary filter does not affect the interior superconvergence properties. Additionally, we can extend our proofs to the accuracy properties of our new SIAC filter used over smoothly-varying meshes. We demonstrated the applicability of the new position-dependent filter for nonuniform meshes by choosing a proper scaling, H, which is obtained by analyzing smoothly-varying meshes.

In conclusion the new contributions of this chapter are:

- A new position-dependent SIAC filter that allows filtering up to boundaries and that ameliorates the principle deficiencies identified in the previous RS and SRV filter;

- Examination and documentation of the reasoning concerning the constant term in the error analysis that led to the proposed work;

- Introduction of the generalized B-splines into the SIAC filter structure. This provides more flexibility and possibilities for SIAC filter;

Table 2.7: L^2- and $L^\infty-$errors for the DG approximation together with the SRV and the new filters for variable coefficient equation (2.12) over the three meshes.

	DG				SRV Filter				New Filter			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
Mesh 2.4.1					$\mathbb{P}^3$							
20	5.54E-06	–	1.93E-05	–	4.40E-06	–	3.66E-05	–	6.36E-05	–	2.02E-04	–
40	3.41E-07	4.02	1.21E-06	4.00	3.14E-07	3.81	3.25E-06	3.49	1.72E-07	8.53	7.61E-07	8.05
80	2.12E-08	4.01	7.50E-08	4.01	1.45E-10	11.08	1.81E-09	10.81	2.78E-10	9.27	2.05E-09	8.53
					$\mathbb{P}^4$							
20	1.62E-07	–	5.69E-07	–	1.89E-05	–	1.44E-04	–	1.72E-05	–	5.41E-05	–
40	4.95E-09	5.03	1.77E-08	5.00	5.74E-09	11.68	5.82E-08	11.28	2.56E-08	9.39	1.12E-07	8.91
80	1.53E-10	5.01	5.48E-10	5.02	1.26E-11	8.83	1.76E-10	8.37	1.16E-11	11.11	7.26E-11	10.59
Mesh 2.4.2					$\mathbb{P}^3$							
20	3.15E-06	–	1.27E-05	–	2.70E-05	–	3.05E-04	–	1.53E-05	–	7.36E-05	–
40	1.96E-07	4.01	8.06E-07	3.98	1.31E-07	7.69	1.54E-06	7.62	3.55E-08	8.75	2.38E-07	8.28
80	1.22E-08	4.00	4.98E-08	4.02	7.51E-09	4.13	1.27E-07	3.60	6.25E-11	9.15	7.84E-10	8.24
					$\mathbb{P}^4$							
20	6.40E-08	–	2.82E-07	–	2.95E-06	–	2.42E-05	–	4.45E-06	–	2.13E-05	–
40	1.98E-09	5.01	8.94E-09	4.98	6.84E-07	2.11	1.12E-05	1.10	4.12E-09	10.08	2.76E-08	9.60
80	6.18E-11	5.00	2.80E-10	5.00	1.51E-09	8.83	3.50E-08	8.33	1.59E-12	11.34	1.55E-11	10.80
Mesh 2.4.3					$\mathbb{P}^3$							
20	2.49E-06	–	9.61E-06	–	1.11E-04	–	8.98E-04	–	5.63E-06	–	2.90E-05	–
40	1.56E-07	4.00	7.18E-07	3.74	2.12E-06	5.71	2.55E-05	5.14	7.96E-09	9.47	4.31E-08	9.39
80	1.02E-08	3.93	4.72E-08	3.93	5.91E-08	5.17	1.06E-06	4.59	3.15E-10	4.66	1.91E-09	4.50
					$\mathbb{P}^4$							
20	4.07E-08	–	1.56E-07	–	2.45E-05	–	1.96E-04	–	1.52E-06	–	7.85E-06	–
40	1.37E-09	4.89	5.31E-09	4.87	4.48E-07	5.77	6.18E-06	4.99	2.98E-10	12.31	1.79E-09	12.09
80	4.41E-11	4.96	1.73E-10	4.94	1.26E-09	8.48	1.91E-08	8.33	2.64E-12	6.82	2.19E-11	6.35

- Demonstration that the filtered approximation is always superconvergent in the uniform mesh linear polynomial case;
- Theoretical proof of the equivalence of smoothly-varying meshes and uniform meshes, in the view of accuracy order;
- Application of the scaled new filter to both smoothly-varying and nonuniform (random) meshes, for which cases we still see significant improvement in the smoothness and an error reduction of the original DG solution, although full superconvergence is not always achieved.

3 Derivative SIAC Filters

3.1 Introduction

In many cases, one can argue persuasively that the changes in values of a function are often more import than the values themselves, such as exhibited by streamline integration of fields. Therefore, an accurate derivative approximation is often required in many areas such as biomechanics, optimization, chemistry and visualization applications. However, computing derivatives of discontinuous Galerkin approximations is challenging because the DG solution only has weak continuity at element boundaries. This means that the strong form of derivatives for a DG solution technically do not hold at element boundaries, and computing the derivative directly does not always produce accurate results. For example, naive and careless use of the derivatives of the discontinuous Galerkin solution directly to streamline integration can produce inconsistent results with the exact solution [61]. Once derivatives are needed near the boundaries, the difficulty increases since the solution often has less regularity in those regions.

In order to obtain accurate approximations for the derivatives of DG solutions, one can use the symmetric derivative SIAC filter we introduced in Chapter 1. As its name implies, the SIAC filter can increase the smoothness of DG solutions, and this smoothness-increasing property helps to enhance the accuracy of derivatives of DG solutions. With the symmetric derivative filter, the accuracy order of the filtered DG solution can be improved from $k+1$ to $2k+1$ regardless of the derivative order. However, previous investigations of derivative filtering have two major limitations: the requirement of uniform meshes and periodic boundary conditions.

In this chapter, we focus on overcoming these two limitations. We propose position-dependent derivative filters to approximate the derivatives of the DG solution over nonuniform meshes and near boundaries. Our main contributions are:

Nonuniform Meshes. Filtering over nonuniform meshes has always been a significant challenge for SIAC filtering because the $2k+1$ accuracy order is no longer guaranteed in general. Most of previous work for nonuniform meshes (such as [30, 55]) only considered a particular family of nonuniform meshes, smoothly-varying meshes. Among these works, only [55] mentioned derivatives over nonuniform meshes. It dis-

cussed the challenges of derivative filtering over nonuniform meshes and presented preliminary results concerning smooth-varying meshes. In this chapter, we propose a method for arbitrary nonuniform meshes: using the scaling $H = h^{\mu}$ for filtering over nonuniform meshes. This does not guarantee that the derivative filtering improves the derivatives of DG solutions to accuracy order of $2k + 1$, but we prove that a higher convergence rate (compared to DG solution) is still obtained. Further, the numerical examples suggest that the accuracy is improved once the mesh is sufficiently refined.

Boundaries. First, we point out that previously derivative filters could not be used near boundaries except for periodic meshes. Without considering derivatives, as mentioned in the previous chapters there are three existing position-dependent filters that can be used to handle boundary regions. The first one is the RS filter (1.12) introduced by Ryan and Shu [57]; the second one is the SRV filter (1.14) given by van Slingerland, Ryan and Vuik [65]; lastly, the newly defined postion-dependent filter discussed in Chapter 2, which we simply refer to the new filter in this chapter. The RS and SRV filters, [57, 65], are constructed by only using central B-splines. They show good performance over uniform meshes. The new filter introduced in Chapter 2 is aimed at nonuniform meshes. It uses $2k+1$ central B-splines and an extra general B-spline. The results in Chapter 2 suggest that adding the extra general B-spline improves the performance of the position-dependent filter over nonuniform meshes compared to using only central B-splines. In this chapter, we extend the SRV position-dependent filter [65] and the new position-dependent filter in Chapter 2 to position-dependent derivative filters. Then, we discuss the advantages and disadvantages of these position-dependent derivative filters over uniform and nonuniform meshes. For nonuniform meshes, we prove that by using the position-dependent derivative filtering, the convergence rate of derivatives of the DG solution can be improved. Numerical comparisons over uniform and nonuniform meshes also demonstrate that the derivative filtered solutions are more accurate than the derivatives of DG approximations.

3.2 Symmetric and One-Sided Derivative Filters

Smoothness-Increasing Accuracy-Conserving filtering is named after its improvement of the smoothness of the filtered approximation. Using the filter in equation (1.6), the filtered solution is a $\mathcal{C}^{k-1}$ function. One can see that the smoothness is significantly improved from the original weakly continuous DG solution. By taking advantage of the improved smoothness, we can obtain better derivative approximations.

3.2.1 Derivative Filters over Nonuniform Meshes

The symmetric derivative filter over uniform meshes was introduced in [56, 62]. In these papers, the authors identified two ways to calculate the derivatives. The first method is a direct calculation of the derivatives of the filtered solution (1.5). The convergence rate of the filtered solution is higher than the derivatives of the DG approximation itself, but the accuracy order decreases and oscillations in the error increase with each successive derivative. The second method is employed to maintain a fixed accuracy order regardless of the derivative order. To calculate the αth derivative of the DG

approximation without losing any accuracy order, we have to use higher-order central B-splines to construct the filter. We remind the reader that

$$\begin{aligned}\partial_x^\alpha u_h^\star &= \partial_x^\alpha \left(K_h^{(2k+1,k+1+\alpha)} \star u_h \right) = \left(\frac{d^\alpha}{dx^\alpha} K_h^{(2k+1,k+1+\alpha)} \right) \star u_h \\ &= \left(\partial_h^\alpha \tilde{K}_h^{(2k+1,k+1,\alpha)} \right) \star u_h = \tilde{K}_h^{(2k+1,k+1,\alpha)} \star \partial_h^\alpha u_h,\end{aligned} \tag{3.1}$$

where the symmetric derivative filter is

$$K^{(2k+1,k+1+\alpha)}(x) = \sum_{\gamma=0}^{2k} c_\gamma^{(2k+1,k+1+\alpha)} \psi^{(k+1+\alpha)}(x+k-\gamma). \tag{3.2}$$

For uniform meshes, [56] showed the filtered solution (3.1) has superconvergence rate of $2k+1$ regardless of the derivative order α.

$$\|\partial_x^\alpha u - \partial_x^\alpha u_h^\star\|_0 \sim \mathcal{O}(h^{2k+1}).$$

Unfortunately, these methods are limited to uniform meshes. For nonuniform meshes, applying the SIAC filter becomes complicated, and the derivative SIAC filter is even more difficult. If we naively apply the same derivative filter (3.2) over nonuniform meshes, we will lose accuracy from $\mathcal{O}(h^{2k+1})$ to $\mathcal{O}(h^{k+1-\alpha})$ since over nonuniform meshes the divided differences of the DG solution no longer have the superconvergence property, see the next chapter for details.

A brief introductory description of symmetric derivative filtering over nonuniform meshes can be found in [55]. It discusses the challenges of symmetric derivative filtering over nonuniform meshes and gives preliminary results for smoothly-varying meshes (an affine mapping of a uniform mesh [30]). Also, in Chapter 2, we have already provided the error estimates for the filtered solutions over smoothly-varying meshes. From those previous results, we can see that the key to SIAC filtering over nonuniform meshes is: the filter scaling H and the divided differences of the DG solution. To begin the study, we first present an error estimate for the divided differences of the DG solution over general nonuniform meshes with a general scaling H.

Lemma 3.2.1. *Under the same conditions in Theorem 1.2.1, let u_h be the DG approximation over a nonuniform mesh. Denote $\Omega_0 \subset\subset \Omega_1 \subset\subset \Omega$, $\ell \geq k+1$. The negative order norm estimate of $u - u_h$ satisfies,*

$$\|(u-u_h)(T)\|_{-\ell,\Omega_1} \leq C h^{2k+1},$$

and

$$\|\partial_H^\alpha (u-u_h)(T)\|_{-\ell,\Omega_0} \leq C_\alpha h^{2k+1} H^{-|\alpha|},$$

where $\alpha = (\alpha_1, \ldots, \alpha_d)$ is an arbitrary multi-index and H is the scaling of the divided difference operator ∂_H^α.

Proof. The proof of the negative order norm estimate was given in [25] and the divided difference estimate was presented as a hypotheses. The proof is trivial and therefore we only give a proof for $d = 1$ case.

Set Ω_0 such that $\Omega_0 + \left[-\frac{|\alpha|H}{2}, \frac{|\alpha|H}{2}\right] \subset \Omega_1$. Consider the first divided difference, by the definition of the negative order norm, we have

$$\begin{aligned}
&\|\partial_H(u-u_h)\|_{-\ell,\Omega_0}\\
=&\sup_{\Phi\in\mathcal{C}_0^\infty(\Omega_0)}\left(\frac{\big((u-u_h)(x+\frac{H}{2}),\Phi\big)-\big((u-u_h)(x-\frac{H}{2}),\Phi\big)}{H\|\Phi\|_{\ell,\Omega_0}}\right),\\
\le&\sup_{\Phi\in\mathcal{C}_0^\infty(\Omega_0)}\frac{\big((u-u_h)(x+\frac{H}{2}),\Phi\big)}{H\|\Phi\|_{\ell,\Omega_0}}+\sup_{\Phi\in\mathcal{C}_0^\infty(\Omega_0)}\frac{\big((u-u_h)(x-\frac{H}{2}),\Phi\big)}{H\|\Phi\|_{\ell,\Omega_0}},\\
\le&\quad\frac{2}{H}\|u-u_h\|_{-\ell,\Omega_1}.
\end{aligned}$$

By induction, we have

$$\|\partial_H^\alpha(u-u_h)(T)\|_{-\ell,\Omega_0}\le C_\alpha h^{2k+1}H^{-|\alpha|},$$

where $C_\alpha = 2^{|\alpha|}C$. The proof is similar for $d>1$ case. □

Lemma 3.2.1 demonstrates the optimal accuracy order estimation of the divided differences of the DG approximation in the sense that the nonuniform mesh is arbitrary [25, 51]. Based on Lemma 3.2.1, we can give the following error estimations for nonuniform meshes.

Theorem 3.2.2. *Under the same conditions as in Lemma 3.2.1, let $K^{(r+1,k+1+\alpha)}$ be the symmetric derivative filter given in* (3.2)*. Denote*

$$\Omega_0 + 2supp(K_H^{(r+1,k+1+\alpha)}) \subset\subset \Omega_1 \subset\subset \Omega.$$

Then, for general nonuniform meshes, we have

$$\|\partial_x^\alpha u - \partial_x^\alpha\left(K_H^{(r+1,k+1+\alpha)}\star u_h\right)\|_{0,\Omega_0}\le Ch^{\frac{r+1}{r+k+2+\alpha}(2k+1)},$$

where $H=h^\mu$ and $\mu=\frac{2k+1}{r+k+2+\alpha}$.

Proof. Set $\Omega_{1/2}$ such that

$$\Omega_0 + supp(K_H^{(r+1,k+1+\alpha)}) \subset \Omega_{1/2}, \qquad \Omega_{1/2} + supp(K_H^{(r+1,k+1+\alpha)}) \subset \Omega_1.$$

By applying Lemma 1.2.2 and Lemma 3.2.1, we have

$$\begin{aligned}
&\left\|\partial_x^\alpha u - \partial_x^\alpha\left(K_H^{(r+1,k+1+\alpha)}\star u_h\right)\right\|_{0,\Omega_0}\\
\le&\left\|\partial_x^\alpha u - K_H^{(r+1,k+1+\alpha)}\star\partial_x^\alpha u\right\|_{0,\Omega_0}+\left\|\partial_x^\alpha\left(K_H^{(r+1,k+1+\alpha)}\star(u-u_h)\right)\right\|_{0,\Omega_0}\\
\le&C_0H^{r+1}+C_1\sum_{|\beta|\le k+1}\left\|\partial_x^{\alpha+\beta}\left(K_H^{(r+1,k+1+\alpha)}\star(u-u_h)\right)\right\|_{-(k+1),\Omega_{1/2}}\\
=&C_0H^{r+1}+C_1\sum_{|\beta|\le k+1}\left\|\left(\tilde{K}_H^{(r+1,k+1-\beta,\alpha+\beta)}\star\partial_H^{\alpha+\beta}(u-u_h)\right)\right\|_{-(k+1),\Omega_{1/2}}\\
=&C_0H^{r+1}+C_1\sum_{|\beta|\le k+1}\left\|\tilde{K}_H^{(r+1,k+1-\beta,\alpha+\beta)}\right\|_{L^1}\left\|\partial_H^{\alpha+\beta}(u-u_h)\right\|_{-(k+1),\Omega_1}\\
\le&C_0H^{r+1}+C_2h^{2k+1}H^{-(k+1+\alpha)},
\end{aligned}$$

Let the scaling $H = h^\mu$ such that

$$H^{r+1} = h^{2k+1} H^{-(k+1+\alpha)}.$$

We then have that $\mu = \frac{2k+1}{r+k+2+\alpha}$ and

$$\left\| \partial_x^\alpha u - \partial_x^\alpha \left(K_H^{(r+1,k+1+\alpha)} \star u_h \right) \right\|_{0,\Omega_0} \leq C h^{\frac{r+1}{r+k+2+\alpha}(2k+1)}.$$

□

Remark 3.2.1 (Discussion of the Number of B-splines)**.** *The filter given in* (3.2) *uses* $(r+1)$ *B-splines. Theorem 3.2.2 implies that by increasing the value of* r*, one can increase the value of* $\frac{r+1}{r+k+2+\alpha}$*, and then approximate the superconvergence rate* $2k+1$ *as close as we want and regardless of the derivative order* α*. However, increasing the value of* r *presents a serious inconvenience for computational implementation. For example, while* $r \gg 2k$*, a multi-precision package is required during the computation process, see Chapter 2. Another disadvantage is that the support size of the filter,* $(r+k+1+\alpha)h^\mu$*, increases with* r *[25]. The increased support size means the convolution involves more DG elements and that the computational cost is increased as well. For nonderivative filtering, we usually keep* $r = 2k$*. There is another consideration for derivative filtering. We notice that the accuracy order decreases with the derivative order* α *if we keep* $r = 2k$*. One solution is to eliminate the negative effect of the derivative order* α *is to use* $r = 2(k+\alpha)$ *instead of* $r = 2k$*. However, our experience shows that the benefit of using* $r = 2(k+\alpha)$ *is limited. It slightly improves the accuracy and smoothness, but increases the computational cost. In this chapter, we will focus on using* $r = 2k$ *for nonuniform meshes.*

3.2.2 Position-Dependent Derivative Filters

For convenience, we use symbol ℓ instead of $k+1+\alpha$ in following section.

Derivative RS Filter

To begin, we first introduce how to extend the first one-sided filter, the RS filter, to the derivative RS filter.

Since the RS filter is an integer shifted symmetric filter, we can easily extend it to the derivative RS filter by increasing the order of B-splines and then modifying the shift function.

$$K^{(2k+1,\ell)}(x) = \sum_{\gamma=0}^{2k} c_\gamma^{(2k+1,\ell)} \psi^{(\ell)} \left(x - x_\gamma(\bar{x}) \right), \tag{3.3}$$

where x_γ depends on the location of the evaluation point $\bar{x}$ and is given by

$$x_\gamma(\bar{x}) = -k + \gamma + [\lambda](\bar{x}),$$

with discrete shift

$$[\lambda](\bar{x}) = \begin{cases} \min\{0, -\frac{2k+\ell}{2} + \lfloor \frac{\bar{x}-x_L}{h} \rfloor\}, & \bar{x} \in [x_L, \frac{x_L+x_R}{2}), \\ \max\{0, \frac{2k+\ell}{2} + \lceil \frac{\bar{x}-x_R}{h} \rceil\}, & \bar{x} \in [\frac{x_L+x_R}{2}, x_R]. \end{cases}$$

Here x_L and x_R are the left and right boundaries, respectively. An example of the derivative RS filter (for the left boundary) with $k = 2$ is given in Figure 3.1.

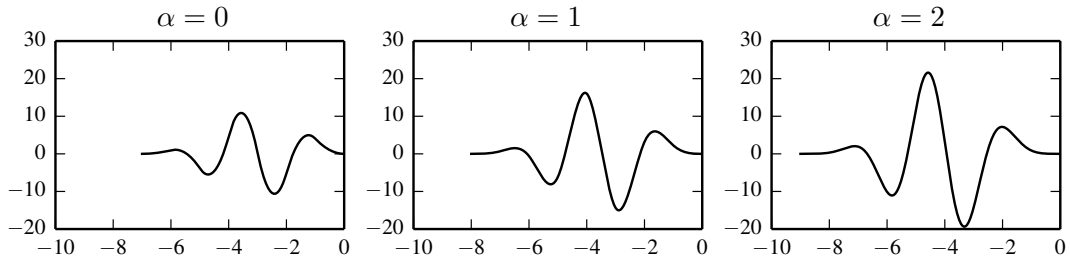


Figure 3.1: Derivative RS filter (3.3) with $k = 2$.

Derivative SRV Filter

The SRV filter has a similar structure as the RS filter, without the discrete shift. By using a similar method, we can extend it to the derivative SRV filter. This is given as

$$K^{(4k+1,\ell)}(x) = \sum_{\gamma=0}^{4k} c_\gamma^{(4k+1,\ell)} \psi^{(\ell)} \left(x - x_\gamma(\bar{x})\right), \tag{3.4}$$

where x_γ depends on the location of the evaluation point $\bar{x}$ and is given by

$$x_\gamma(\bar{x}) = -2k + \gamma + \lambda(\bar{x}).$$

We adjust the shift function $\lambda(\bar{x})$ (1.15) by

$$\lambda(\bar{x}) = \begin{cases} \min\{0, -\frac{4k+\ell}{2} + \frac{\bar{x}-x_L}{h}\}, & \bar{x} \in [x_L, \frac{x_L+x_R}{2}), \\ \max\{0, \frac{4k+\ell}{2} + \frac{\bar{x}-x_{right}}{h}\}, & \bar{x} \in [\frac{x_L+x_R}{2}, x_R]. \end{cases}$$

Here x_L and x_R are the left and right boundaries, respectively. An example of the derivative SRV filter (for the left boundary) with $k = 2$ is given in Figure 3.2.

Remark 3.2.2. *As mentioned in Chapter 1, the main difference between the RS filter and the SRV filter is the number of B-splines.*

Remark 3.2.3. *The theoretical analysis of the derivative RS and SRV filters remains the same as in Theorem 3.2.2. Unlike the derivative RS filter and the symmetric derivative filter, the derivative SRV filter has a scaling $H = h^{\frac{2k+1}{5k+2+\alpha}}$, which is much larger than the scaling of the symmetric derivative filter (or the derivative RS filter), $H = h^{\frac{2k+1}{3k+2+\alpha}}$.*

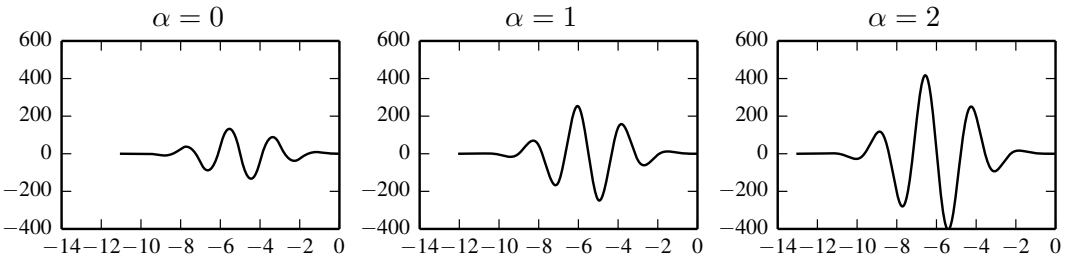


Figure 3.2: Derivative SRV filter (3.4) with $k = 2$.

New Position-Dependent Derivative Filter

For the new position-dependent filter introduced in Chapter 2, we need to shift the $2k+1$ central B-splines and then change the extra general B-spline according to the derivative order α. To complete these changes, we have to change the knot sequence of the original new position-dependent filter, which is used only for the DG approximation u_h without derivatives. For the new position-dependent derivative filter near the left boundary (similar for the right boundary), we need to redistribute the knots in the knot matrix $\mathbf{T}$ to meet the derivative requirement by

$$T(\gamma, j) = \begin{cases} -2k - \ell + j + \gamma + \frac{\bar{x} - x_L}{h}, & 0 \le \gamma \le 2k, 0 \le j \le \ell; \\ \frac{\bar{x} - x_L}{h} + \min\{j - \alpha, 0\}, & \gamma = 2k+1, 0 \le j \le \ell. \end{cases} \tag{3.5}$$

The position-dependent derivative filter is then given by

$$K_{\mathbf{T}}^{(2k+1,\ell)}(x) = \underbrace{\sum_{\gamma=0}^{2k} c_{\gamma}^{(2k+1,\ell)} \psi_{\mathbf{T}(\gamma)}^{(\ell)}(x)}_{2k+1 \text{ central B-splines}} + \underbrace{c_{2k+1}^{(2k+1,\ell)} \psi_{\mathbf{T}(2k+1)}^{(\ell)}}_{\text{General B-spline}}. \tag{3.6}$$

We note that in formula (3.5), if we keep the order of B-splines as $k+1$, then when $\alpha > k$ the general B-spline added at the boundary will reduce to the central B-spline and then the purpose of adding a special B-spline at the boundary fails. It is necessary to use B-splines of order $\ell = k + 1 + \alpha$ instead of $k+1$ when $\alpha > k$. We note that the new position-dependent derivative filter allows us to approximate arbitrary order derivatives near boundaries theoretically. For example, Figure 3.3 shows the new position-dependent derivative filters with $k = 2$ for the first and second derivatives at the left boundary. Compared to the derivative RS filter (3.3), the derivative SRV filter (3.4), the new position-dependent derivative filter clearly has reduced support and magnitude (range from -400 to 600 versus -4 to 6).

Theorem 3.2.3. *Under the same conditions as in Lemma 3.2.1, let $K_{\mathbf{T}}^{(2k+1,\ell)}$ be the new position-dependent derivative filter* (3.6)*. We have*

$$\left\| \partial_x^{\alpha} u - \partial_x^{\alpha} \left(K_{H\mathbf{T}}^{(2k+1,\ell)} \star u_h \right) \right\|_{0,\Omega_0} \le C h^{\mu(2k+2)},$$

where $H = h^{\mu}$, $\mu = \frac{2k+1}{3k+3+\alpha}$.

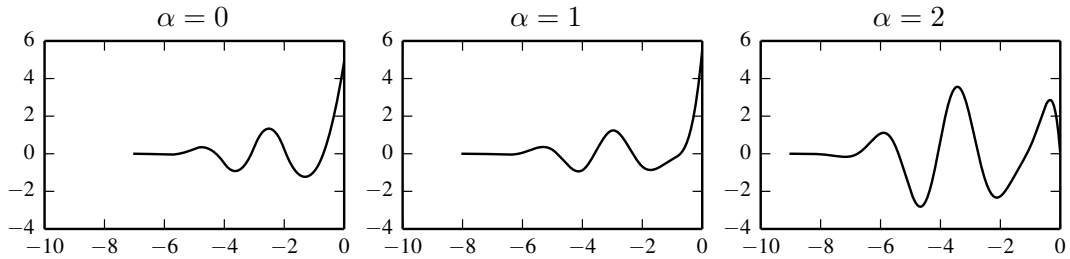


Figure 3.3: New position-dependent derivative filter (3.6) with $k = 2$.

Proof.

$$\begin{aligned}
&\left\|\partial_x^\alpha u - \partial_x^\alpha \left(K_{H\mathbf{T}}^{(2k+1,\ell)} \star u_h\right)\right\|_{0,\Omega_0} \\
\leq & C_0 H^{2k+2} + \left\|\partial_x^\alpha \left(\sum_{\gamma=0}^{2k} c_\gamma \psi_{H\mathbf{T}(\gamma)}^{(\ell)} \star (u - u_h)\right)\right\|_{0,\Omega_0} \\
&+ \left\|\partial_x^\alpha \left(c_{2k+1} \psi_{H\mathbf{T}(2k+1)}^{(\ell)} \star (u - u_h)\right)\right\|_{0,\Omega_0}
\end{aligned}$$

For the second term on the left side of the above inequality, which only involves central B-splines, similar to Theorem 3.2.2, we have

$$\left\|\partial_x^\alpha \left(\sum_{\gamma=0}^{2k} c_\gamma \psi_{H\mathbf{T}(\gamma)}^{(\ell)} \star (u - u_h)\right)\right\|_{0,\Omega_0} \leq C_1 h^{2k+1} H^{-(k+1+\alpha)}.$$

For the third term with a general B-spline, we have

$$\begin{aligned}
&\left\|\partial_x^\alpha \left(c_{2k+1} \psi_{H\mathbf{T}(2k+1)}^{(\ell)} \star (u - u_h)\right)\right\|_{0,\Omega_0} \\
\leq & C_2 \sum_{\beta \leq k+1} \left\|c_{2k+1} \left(\frac{d^{\alpha+\beta}}{dx^{\alpha+\beta}} \psi_{H\mathbf{T}(2k+1)}^{(\ell)}\right) \star (u - u_h)\right\|_{-(k+1),\Omega_{1/2}} \\
\leq & C_2 \sum_{\beta \leq k+1} \left\|c_{2k+1} \left(\frac{d^{\alpha+\beta}}{dx^{\alpha+\beta}} \psi_{H\mathbf{T}(2k+1)}^{(\ell)}\right)\right\|_{L^1} \|u - u_h\|_{-(k+1),\Omega_1} \\
\leq & C_3 \sum_{\beta \leq k+1} H^{-(\alpha+\beta)} \left\|\left(\frac{d^{\alpha+\beta}}{dx^{\alpha+\beta}} \psi_{\mathbf{T}(2k+1)}^{(\ell)}\right)\right\|_{L^1} \|u - u_h\|_{-(k+1),\Omega_1} \\
\leq & C_4 h^{2k+1} H^{-(k+1+\alpha)},
\end{aligned}$$

where

$$\Omega_0 + supp(K_{H\mathbf{T}}^{(2k+1,\ell)}) \subset \Omega_{1/2}, \quad \Omega_{1/2} + supp(K_{H\mathbf{T}}^{(2k+1,\ell)}) \subset \Omega_1.$$

Then, we have

$$\left\|\partial_x^\alpha u - \partial_x^\alpha \left(K_{H\mathbf{T}}^{(2k+1,\ell)} \star u_h\right)\right\|_{0,\Omega_0} \leq C_0 H^{2k+2} + C_5 h^{2k+1} H^{-(k+1+\alpha)}.$$

Similar to the symmetric filter case in Theorem 3.2.2, we require that the scaling H satisfies $H^{2k+2} = h^{2k+1}H^{-(k+1+\alpha)}$ and finally, we have

$$\left\|\partial_x^\alpha u - \partial_x^\alpha \left(K_{H\mathbf{T}}^{(2k+1,\ell)} \star u_h\right)\right\|_{0,\Omega_0} \leq C h^{\mu(2k+2)},$$

where $H = h^\mu$ and $\mu = \frac{2k+1}{3k+3+\alpha}$. □

3.2.3 Computational Considerations

Theorem 3.2.2 and Theorem 3.2.3 give convergence rates of the symmetric and position-dependent derivative filters, respectively. One can easily verify that the convergence rates are better than calculating the derivatives of DG approximation directly, $k+1-\alpha$. Now, let us consider the computational efficiency of the one-sided derivative filters.

Support Size of the Filters

As mentioned before, the support size of the filters is one import component which affects the computational cost of using the filters. For nonuniform meshes, Theorem 3.2.2 and Theorem 3.2.3 require the scaling has the form $H = h^\mu$. For convenience we let the degree $k \to \infty$. Then, the symmetric derivative filter, the derivative RS filter, and the new derivative filters have the same scaling $H = h^{2/3}$ and support size $(3k + 1 + \alpha)h^{2/3}$; the derivative SRV filter has a much larger scaling $H = h^{2/5}$ and support size $(5k+1+\alpha)h^{2/5}$. It is obvious that the SRV filter has a significantly larger support size compared to other filters, especially when h is very small (a very fine mesh). It follows that the computational cost of using the new filter is much cheaper than using other filters.

However, we notice that the scaling $H = h^\mu$ is still quite large compared to h. The large support usually has negative effects on the accuracy over coarse meshes. Let the domain be $\Omega = [0, 1]$ and $h = 1/N$, where N is the number of elements. In order to guarantee the conclusions in Theorem 3.2.2 and Theorem 3.2.3, we must choose N large enough so that the support size of the filters is less than the domain size, which requires

$$(r + k + \alpha + 1)h^\mu \leq 1 \Longrightarrow N \geq (r + k + \alpha + 1)^{1/\mu},$$

here $r = 2k$ for the symmetric and new position-dependent derivative filters and $r = 4k$ for the derivative SRV filter. Table 3.1 gives the minimum number of elements for different filters. We note that for the SRV filter, the required number of elements is always too large, this is one important reason that the SRV filter performs poorly over nonuniform meshes. Once N is smaller than the minimum requirement given in Table 3.1, we have to rescale the filter by using scaling $H = 1/(r+k+\alpha+1)$. However, this rescaling technique normally has negative effects on the accuracy order.

Table 3.1: The minimum number of elements according to the derivative order α. Here, N_1 is used for the symmetric and new position-dependent derivative filters and N_2 is used for the derivative SRV filter.

N_1 / N_2	$\alpha = 0$	$\alpha = 1$	$\alpha = 2$	$\alpha = 3$	$\alpha = 4$	$\alpha = 5$
$k = 1$	8 / 89	12 / 130	15 / 182	19 / 243	23 / 317	27 / 402
$k = 2$	19 / 402	23 / 499	27 / 610	32 / 734	37 / 872	42 / 1024
$k = 3$	32 / 1024	37 / 1192	42 / 1375	47 / 1574	53 / 1789	59 / 2021

Order of the B-splines

Although we know that the new derivative filter is already much more efficient than the derivative SRV filter, it still less efficient compared to the filters for uniform meshes (scaling $H = \mathcal{O}(h)$). It is understandable that one needs more effort to deal with nonuniform meshes compared to uniform meshes, but we still want to reduce the computational cost if possible. Based on the current support size, $(3k + 1 + \alpha)h^{\mu}$ with $\mu = 2/3$, there are two directions to reduce the support size. The first one is to increase the value of μ. This idea will lead to a new analysis about the relation of μ and the structure of nonuniform meshes. This will be discussed in the next chapter. The second thought is to reduce the order of the B-splines. One can use only the order $k + 1$ for derivatives of order $\leq k$. Then the support size is reduced to $(3k + 1)h^{2/3}$. Although the improvement is not too much and has a risk to reduce the accuracy when $\alpha \gtrapprox k$.

Remark 3.2.4. *Reducing the support size of the filters has another benefit besides reducing the computational cost. Once the support size of the filters is reduced, one can apply the symmetric filter in a larger region, which enhances the performance.*

3.3 Numerical Results

In the previous section, we proposed three position-dependent derivative filters and investigated how to choose the proper scaling of the filters over nonuniform meshes. We also proved that the filtered solutions have better accuracy order and smoothness compared to the original DG approximations regardless of the derivative order α. We now turn to the numerical examples for the position-dependent derivative filters. The aims of this section are:

1. Testing the position-dependent derivative filters (the SRV and new filter) for uniform meshes, which has never been done before;

2. Applying the symmetric and position-dependent derivative filters over different nonuniform meshes;

3. Comparing the derivative filters with different order B-splines. For convenience, we introduce the following notation:

- the derivative of the filtered solution, $\partial_x^\alpha u_h^\star$. This filtered solution uses the standard filter and then takes the derivative.
- the filtered derivative, $\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$, which uses the higher order derivative filter $K_H^{(r+1,k+1+\alpha)}$ for filtering the DG solution.

Remark 3.3.1. *Here, we do not present the results of using the derivative RS filter as it is very similar to the derivative SRV filter. Their difference is the number of B-splines, and we will discuss the effects of the number of B-splines in the next chapter. We note that for the following uniform mesh examples, the SRV filter has better performance compared to the RS filter.*

We note that the DG approximation makes sense only when $\alpha \leq k$. In addition, the derivative of the filtered solution $\partial_x^\alpha u_h^\star$ loses the wanted accuracy order when $\alpha > k$ ($u_h^\star = K_H^{r+1,k+1} \star u_h$ is a $\mathcal{C}^{k-1}$ function only). Therefore, we mainly present comparison examples with $\alpha \leq k$ situation in this section. When $\alpha > k$, we only present the results of the filtered derivative $\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$, and we point out that the filtered solution $\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$ has a theoretical meaning for an arbitrary α, but the accuracy deteriorates with each successive derivative. However, we also note that once $\alpha > k$, the nonuniform meshes have to be sufficiently refined in order to see the accuracy improvement. Because of these reasons, we only present $\alpha = k+1$ case for nonuniform meshes. Also, since the symmetric derivative filter is applied in the interior region of each example, we do not present it separately.

3.3.1 Uniform Mesh

Before approaching nonuniform meshes, we first apply the position-dependent derivative filters over uniform meshes. Here we present results of using both the SRV filter and the new filter since each of them has an advantage over uniform meshes that we address in the following examples. Consider a linear convection equation

$$\begin{aligned} u_t + u_x &= 0, \qquad x \in [0,1], \\ u(x,0) &= \sin(2\pi x), \end{aligned} \tag{3.7}$$

at time $T = 1$ with periodic boundary conditions. For uniform meshes, we can also use scaling $H = h^\mu$ and obtain results as Theorems 3.2.2 and 3.2.3 described. However, according to the analysis in [25, 65], in order to maximize the benefits of using central B-splines over uniform meshes, we choose the uniform mesh size, h, as the filter scaling. Here, we compare the derivatives of the DG approximation, the filtered solutions (the SRV and new filter) with using B-splines of order $k+1+\alpha$ (Table 3.2 and Figure 3.4) and using B-splines of order $k+1$ (Table 3.3 and Figure 3.5). From the tables, we can see that the filtered solutions $\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$ and $\partial_x^\alpha (K_H \star u_h)$ have better accuracy compared to the original DG solutions. Generally speaking, $\partial_x^\alpha (K_H \star u_h)$ is better for high order derivatives and $\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$ performs well for the first derivative. Further comparison of these two methods will list in the end of the section.

With the scaling $H = h$, the SRV filter clearly has an advantage for uniform meshes. Because the SRV is constructed using only central B-splines, and was proven to have $2k+1$ accuracy order regardless of the derivative order α for linear equations over uniform meshes in [57]. In Tables 3.2 and 3.3, the SRV filter shows smaller errors compared to the new filter near the boundaries, especially when α is large. For the new position-dependent derivative filter, we notice that the filtered solutions only have accuracy of order $k+1-\alpha$ in Tables 3.2 and 3.3. This is because we use a scaling $H = h$ instead of a scaling $H = h^\mu$ as in Theorem 3.2.3. We note that if using a multi-precision package is acceptable, then the SRV filter is more advantageous for the accuracy order. However, if only double precision is available during computation (for example, GPU computing), then in order to avoid round-off errors, the new position-dependent filter is a better choice, see Chapter 2. However, when $\alpha > k$, the optimal choice is still the SRV filter with B-splines of order $k+1+\alpha$ because only this filter does not lose the accuracy with each successive derivative.

We note that the derivative of filtered solution $\partial_x^\alpha (K_H \star u_h)$ also performs well near boundaries for uniform meshes, especially for the new position-dependent filter. However, for the derivative order $\alpha > k$, we still need to use higher-order B-splines to construct the derivative filters. Figures 3.4 and 3.5 present the point-wise error plots in log scale using the DG approximation of degree $k = 2$. After filtering, the filtered approximations are much smoother than the DG solution. In order to reduce oscillations in the interior regions, we still have to use B-splines of order $k+1+\alpha$.

Remark 3.3.2. *For uniform meshes, we choose to use the scaling $H = h$ instead of the scaling $H = h^\mu$ in Theorem 1.3.4. This is because for uniform meshes, the scaling $H = h$ can provide a better accuracy order of $2k+1$ compared to the conclusion in Theorem 1.3.4, especially in the interior region. Also, the SRV filter benefits of the scaling $H = h$ in the boundary region once quadruple precision is used. If the scaling $H = h^\mu$ is used for uniform meshes, the accuracy order will decrease and the error magnitude will increase in the interior region. However, the RLKV filter will have better accuracy order in the boundary region, and the error magnitude will improve once the mesh is sufficiently refined.*

3.3.2 Nonuniform Mesh

Now we show the main results of this chapter: the position-dependent derivative filtering over nonuniform meshes. Before proceeding further, we first give the numerical setting of nonuniform meshes. In order to generate a more general format for nonuniform meshes, we use a random number generator to design the following two meshes.

Mesh 3.3.1. *The first nonuniform mesh that we consider is defined by*

$$x_{\frac{1}{2}} = 0, \quad x_{N+\frac{1}{2}} = 1, \quad x_{j+\frac{1}{2}} = \left(j + b \cdot r_{j+\frac{1}{2}}\right) h, \qquad j = 1, \ldots, N-1$$

where $\left\{r_{j+\frac{1}{2}}\right\}_{j=1}^{N-1}$ are random numbers between $(-1, 1)$, and $b \in (0, 0.5]$ is a constant number. The size of element $\Delta x_j = x_{j+\frac{1}{2}} - x_{j-\frac{1}{2}}$ is between $((1-2b)h, (1+2b)h)$. In

Table 3.2: L^2- and $L^\infty-$errors for the αth derivative of the DG approximation $\partial_x^\alpha u_h$ together with the two filtered solutions (the SRV and new filters) for linear convection equation (3.7), over uniform meshes. The B-spline order is $k+1+\alpha$ and the filter scaling is taken as $H=h$.

	$\partial_x^\alpha u_h$				$\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$ (SRV)				$\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$ (New)			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\boldsymbol{\alpha=1}$					$\mathbb{P}^1$							
20	4.62E-01	–	1.22E+00	–	1.43E-02	–	4.41E-02	–	1.22E-02	–	2.07E-02	–
40	2.32E-01	0.99	6.22E-01	0.98	1.55E-03	3.20	2.61E-03	4.08	1.55E-03	2.97	4.60E-03	2.17
80	1.16E-01	1.00	3.12E-01	0.99	1.91E-04	3.02	2.74E-04	3.25	2.04E-04	2.92	1.20E-03	1.94
160	5.82E-02	1.00	1.56E-01	1.00	2.37E-05	3.01	3.36E-05	3.03	2.84E-05	2.84	3.01E-04	1.99
320	2.91E-02	1.00	7.81E-02	1.00	2.96E-06	3.01	4.18E-06	3.01	4.22E-06	2.75	7.50E-05	2.00
					$\mathbb{P}^2$							
20	2.19E-02	–	7.97E-02	–	1.40E-04	–	9.00E-04	–	4.78E-04	–	3.09E-03	–
40	5.48E-03	2.00	2.01E-02	1.98	6.69E-07	7.71	1.91E-06	8.88	8.14E-05	2.55	6.83E-04	2.18
80	1.37E-03	2.00	5.05E-03	2.00	1.69E-08	5.31	2.52E-08	6.24	1.44E-05	2.50	1.68E-04	2.02
160	3.43E-04	2.00	1.26E-03	2.00	5.13E-10	5.04	7.37E-10	5.09	2.54E-06	2.50	4.20E-05	2.00
320	8.56E-05	2.00	3.16E-04	2.00	2.14E-11	4.58	3.04E-11	4.60	4.50E-07	2.50	1.05E-05	2.00
					$\mathbb{P}^3$							
20	6.55E-04	–	2.80E-03	–	2.41E-06	–	1.59E-05	–	6.24E-06	–	2.50E-05	–
40	8.20E-05	3.00	3.53E-04	2.99	2.10E-09	10.16	3.89E-09	11.99	1.04E-07	5.91	7.61E-07	5.04
80	1.02E-05	3.00	4.42E-05	3.00	9.95E-12	7.72	1.63E-11	7.90	2.18E-09	5.58	2.99E-08	4.67
160	1.28E-06	3.00	5.53E-06	3.00	1.10E-13	6.50	1.62E-13	6.65	9.58E-11	4.51	1.78E-09	4.07
320	1.60E-07	3.00	6.92E-07	3.00	8.98E-15	3.62	1.27E-14	3.67	4.24E-12	4.50	1.11E-10	4.00
$\boldsymbol{\alpha=2}$					$\mathbb{P}^2$							
20	2.67E+00	–	6.96E+00	–	7.20E-04	–	4.12E-03	–	6.42E-02	–	3.71E-01	–
40	1.34E+00	1.00	3.50E+00	0.99	5.90E-06	6.93	1.73E-05	7.89	2.27E-02	1.50	1.74E-01	1.09
80	6.70E-01	1.00	1.75E+00	1.00	1.29E-07	5.51	1.83E-07	6.57	8.03E-03	1.50	8.67E-02	1.00
160	3.35E-01	1.00	8.78E-01	1.00	3.55E-09	5.19	5.02E-09	5.19	2.84E-03	1.50	4.33E-02	1.00
320	1.67E-01	1.00	4.39E-01	1.00	1.39E-10	4.67	1.97E-10	4.67	1.00E-03	1.50	2.17E-02	1.00
					$\mathbb{P}^3$							
20	1.34E-01	–	4.78E-01	–	6.13E-05	–	3.87E-04	–	6.48E-04	–	4.84E-03	–
40	3.36E-02	2.00	1.21E-01	1.99	2.35E-08	11.35	3.59E-08	13.39	1.58E-05	5.36	1.49E-04	5.03
80	8.40E-03	2.00	3.02E-02	2.00	1.03E-10	7.83	1.48E-10	7.93	1.39E-06	3.51	1.73E-05	3.10
160	2.10E-03	2.00	7.56E-03	2.00	8.46E-13	6.93	1.20E-12	6.95	1.23E-07	3.50	2.16E-06	3.00
320	5.25E-04	2.00	1.89E-03	2.00	5.70E-14	3.89	8.06E-14	3.89	1.09E-08	3.50	2.70E-07	3.00
$\boldsymbol{\alpha=3}$					$\mathbb{P}^3$							
20	1.64E+01	–	4.16E+01	–	3.68E-04	–	2.26E-03	–	1.74E-02	–	1.09E-01	–
40	8.19E+00	1.00	2.09E+01	0.99	1.93E-07	10.90	8.61E-07	11.36	3.07E-03	2.50	2.54E-02	2.10
80	4.10E+00	1.00	1.05E+01	1.00	7.68E-10	7.97	1.30E-09	9.38	5.43E-04	2.50	6.42E-03	1.99
160	2.05E+00	1.00	5.24E+00	1.00	5.93E-12	7.02	8.96E-12	7.18	9.60E-05	2.50	1.61E-03	2.00
320	1.02E+00	1.00	2.62E+00	1.00	8.50E-13	2.80	2.53E-11	-1.50	1.70E-05	2.50	4.02E-04	2.00

order to save space, in this chapter we present an example with $b=0.4$ only, other values of b such as $0.1, 0.2$ and 0.45 have also been calculated.

Mesh 3.3.2. *The second nonuniform mesh is more irregular than the first one. We distribute the element interface by $x_{j+\frac{1}{2}}$, $j=1,\ldots,N-1$ randomly for the entire domain and require only*

$$\Delta x_j = x_{j+\frac{1}{2}} - x_{j-\frac{1}{2}} \geq b \cdot h, \qquad j=0,\ldots,N.$$

In this paper, we only present $b=0.5$ case for this mesh, other values of b such as $0.6, 0.8$, etc. have also been calculated.

We remark that we have tested various differential equations over both kinds of nonuniform meshes: Mesh 3.3.1 and Mesh 3.3.2. However, the filtered approximations

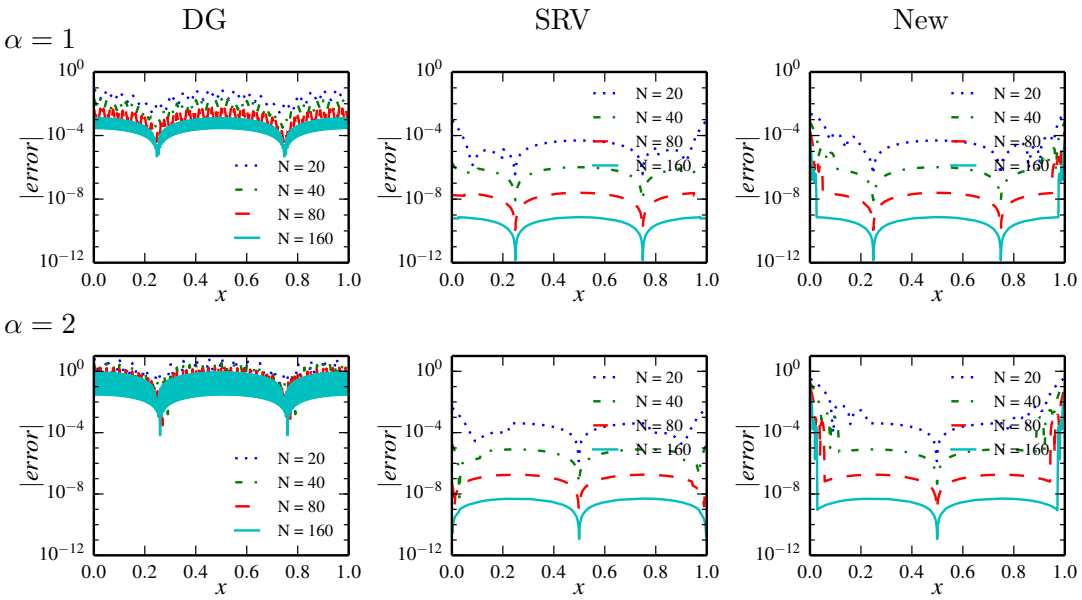


Figure 3.4: The point-wise errors in log scale of the first and second derivatives of the DG approximation $\partial_x^\alpha u_h$ together with the two filtered solutions (the SRV and new filters) for linear equation (3.7), over uniform meshes. The B-spline order is $k+1+\alpha$, the filter scaling is taken as $H = h$, and $k = 2$.

have similar performance since the performance mainly depends on the mesh. In order to save space, we present one equation for each nonuniform mesh.

Comparison of the SRV filter and new filter over Nonuniform Mesh 3.3.1

In Chapter 2, we showed that the SRV filter has worse performance compared to the new position-dependent filter over nonuniform meshes for filtering the solution itself. We also mentioned that the enormous support size of the SRV filter causes problems: we have to rescale the SRV filter to fit the domain size then we can not guarantee either the accuracy order nor error reduction. Table 3.1 shows the minimum requirement of the number of elements for the SRV filter, and we can see that it is difficult to satisfy. Based on these deficiencies, we conclude that the SRV filter is not suitable for approximating derivatives over nonuniform meshes. However, in order to provide a complete view of the position-dependent derivative filters, we still give one example of using the SRV filter for the first derivative over Mesh 3.3.1. Table 3.4 shows that with the SRV filter, the filtered solutions (no matter what order of B-splines are used) are even worse than the derivative of the DG approximation. In the rest of this section, we focus on testing the new filter over nonuniform meshes.

Table 3.3: L^2- and $L^\infty-$errors for the αth derivative of the DG approximation $\partial_x^\alpha u_h$ together with the two filtered solutions (the SRV and new filters) for linear convection equation (3.7), over uniform meshes. The B-spline order is $k+1$ and the filter scaling is taken as $H = h$.

	$\partial_x^\alpha u_h$				$\partial_x^\alpha (K_H \star u_h)$ (SRV)				$\partial_x^\alpha (K_H \star u_h)$ (New)			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\boldsymbol{\alpha = 1}$					$\mathbb{P}^1$							
20	4.62E-01	–	1.22E+00	–	1.25E-02	–	2.52E-02	–	1.45E-02	–	6.89E-02	–
40	2.32E-01	0.99	6.22E-01	0.98	1.53E-03	3.03	2.25E-03	3.48	1.91E-03	2.92	1.28E-02	2.43
80	1.16E-01	1.00	3.12E-01	0.99	1.91E-04	3.01	2.72E-04	3.05	2.63E-04	2.86	2.64E-03	2.28
160	5.82E-02	1.00	1.56E-01	1.00	2.38E-05	3.00	3.38E-05	3.01	3.84E-05	2.78	5.90E-04	2.16
320	2.91E-02	1.00	7.81E-02	1.00	2.96E-06	3.00	4.22E-06	3.00	5.95E-06	2.69	1.39E-04	2.09
					$\mathbb{P}^2$							
20	2.19E-02	–	7.97E-02	–	5.03E-05	–	3.23E-04	–	5.66E-05	–	2.84E-04	–
40	5.48E-03	2.00	2.01E-02	1.98	5.38E-07	6.55	9.68E-07	8.38	1.05E-06	5.75	8.34E-06	5.09
80	1.37E-03	2.00	5.05E-03	2.00	1.51E-08	5.16	2.22E-08	5.44	2.25E-08	5.55	2.87E-07	4.86
160	3.43E-04	2.00	1.26E-03	2.00	4.83E-10	4.96	6.93E-10	5.00	7.49E-10	4.91	1.39E-08	4.37
320	8.56E-05	2.00	3.16E-04	2.00	2.10E-11	4.53	2.98E-11	4.54	3.22E-11	4.54	8.03E-10	4.12
					$\mathbb{P}^3$							
20	6.55E-04	–	2.80E-03	–	9.62E-07	–	6.45E-06	–	4.03E-06	–	1.64E-05	–
40	8.20E-05	3.00	3.53E-04	2.99	1.34E-09	9.48	2.51E-09	11.33	3.56E-08	6.83	2.59E-07	5.98
80	1.02E-05	3.00	4.42E-05	3.00	6.65E-12	7.66	1.09E-11	7.85	4.55E-10	6.29	6.34E-09	5.35
160	1.28E-06	3.00	5.53E-06	3.00	9.66E-14	6.11	1.41E-13	6.27	1.95E-11	4.55	3.50E-10	4.18
320	1.60E-07	3.00	6.92E-07	3.00	8.92E-15	3.44	1.26E-14	3.48	8.61E-13	4.50	2.17E-11	4.01
$\boldsymbol{\alpha = 2}$					$\mathbb{P}^2$							
20	2.67E+00	–	6.96E+00	–	1.94E-04	–	5.41E-04	–	1.14E-03	–	1.11E-02	–
40	1.34E+00	1.00	3.50E+00	0.99	1.19E-05	4.03	2.50E-05	4.44	7.49E-05	3.92	7.18E-04	3.95
80	6.70E-01	1.00	1.75E+00	1.00	7.42E-07	4.00	1.49E-06	4.07	6.31E-06	3.57	8.11E-05	3.15
160	3.35E-01	1.00	8.78E-01	1.00	4.64E-08	4.00	9.73E-08	3.93	5.45E-07	3.53	9.89E-06	3.04
320	1.67E-01	1.00	4.39E-01	1.00	2.90E-09	4.00	6.16E-09	3.98	4.76E-08	3.52	1.22E-06	3.02
					$\mathbb{P}^3$							
20	1.34E-01	–	4.78E-01	–	5.93E-06	–	4.03E-05	–	1.24E-04	–	1.02E-03	–
40	3.36E-02	2.00	1.21E-01	1.99	1.04E-08	9.15	1.91E-08	11.05	2.22E-07	9.12	2.66E-06	8.58
80	8.40E-03	2.00	3.02E-02	2.00	5.43E-11	7.58	1.20E-10	7.31	3.92E-10	9.15	7.91E-09	8.39
160	2.10E-03	2.00	7.56E-03	2.00	7.45E-13	6.19	1.70E-12	6.15	5.52E-12	6.15	1.03E-10	6.27
320	5.25E-04	2.00	1.89E-03	2.00	5.65E-14	3.72	9.20E-14	4.21	1.67E-13	5.05	5.58E-12	4.20
$\boldsymbol{\alpha = 3}$					$\mathbb{P}^3$							
20	1.64E+01	–	4.16E+01	–	3.87E-05	–	2.55E-04	–	4.23E-04	–	3.61E-03	–
40	8.19E+00	1.00	2.09E+01	0.99	4.36E-07	6.47	1.12E-06	7.83	4.26E-06	6.63	5.85E-05	5.95
80	4.10E+00	1.00	1.05E+01	1.00	1.48E-08	4.88	3.47E-08	5.01	8.67E-08	5.62	1.26E-06	5.54
160	2.05E+00	1.00	5.24E+00	1.00	4.68E-10	4.98	1.09E-09	5.00	3.73E-09	4.54	6.67E-08	4.24
320	1.02E+00	1.00	2.62E+00	1.00	1.48E-11	4.98	7.85E-11	3.79	1.65E-10	4.50	4.15E-09	4.01

Linear Equation over Mesh 3.3.1

For Mesh 3.3.1, we present results for the linear equation (3.7) with the first, second and third derivatives. The L^2 and L^∞ norm errors are given in Table 3.5 and Figure 3.6 shows the point-wise error in log scale. When $\alpha \leq k$, both the derivative of filtered solutions $\partial_x^\alpha (K_H \star u_h)$ and the filtered derivative $\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$ have better accuracy and convergence rates than the original DG approximation. The filtered approximation $\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$ shows better smoothness and theoretically has a better accuracy order than $\partial_x^\alpha (K_H \star u_h)$ when $\alpha \leq k$, but $\partial_x^\alpha (K_H \star u_h)$ has better accuracy near the boundaries. For smoothness, the results are similar to the uniform mesh case; $\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$ has fewer oscillations compared to the DG solution and $\partial_x^\alpha (K_H \star u_h)$. Furthermore, we point out that by using higher-order B-splines we can disregard the requirement that $\alpha \leq k$.

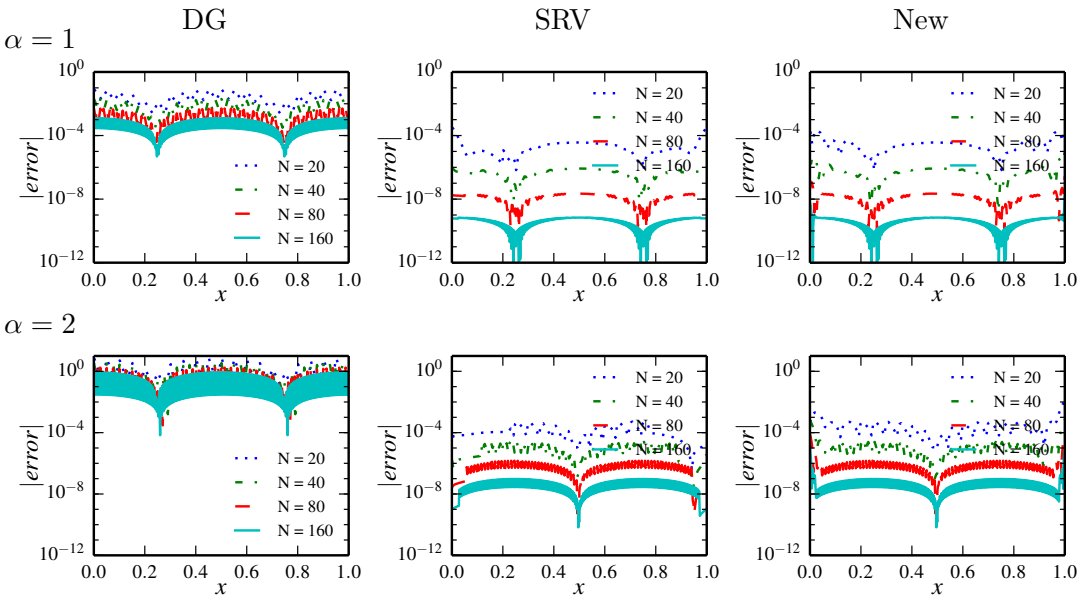


Figure 3.5: The point-wise errors in log scale of the first and second derivatives of the DG approximation $\partial_x^\alpha u_h$ together with the two filtered solutions (the SRV and new filters) for linear equation (3.7), over uniform meshes. The B-spline order is $k+1$, the filter scaling is taken as $H = h$, and $k = 2$.

The point-wise error plots are given in Figure 3.6, the middle column is the filtered approximation $\partial_x^\alpha (K_H \star u_h)$, which shows more oscillations than $\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$, especially in the interior regions. We note, however that the support size of the filter that uses a higher-order B-spline increases with the derivative order α and it slightly increases the computational cost. Near the boundaries, the filtered solutions have a larger error magnitude than those in the interior region. This is because near the boundaries we cannot use symmetric information around the filtered points, and the general B-spline has less regularity compared to the central B-spline. We note that the coarse meshes (such as $N = 20$ or even $N = 40$) are not sufficient to use the position-dependent derivative filter, the filtered solution may have larger errors compared to the original DG approximation.

Variable Coefficient Equation over Mesh 3.3.2

After testing the constant coefficient equation (3.7), we move to a variable coefficient equation,

$$\begin{aligned} u_t + (au)_x &= f, \quad (x,t) \in [0,1] \times (0,T] \\ u(x,0) &= \sin(2\pi x), \end{aligned} \tag{3.8}$$

Table 3.4: L^2- and $L^\infty-$errors for the first derivative of the DG approximation $\partial_x^\alpha u_h$ together with the two filtered solution $\partial_x^\alpha (K_H \star u_h)$ and $\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$ (with the SRV filter) for linear equation (3.7), over Mesh 3.3.1. The filter scaling is taken as $H = h^{2/5}$ near boundaries and $H = h^{2/3}$ for the interior region (where the symmetric filter is applied) .

	$\partial_x^\alpha u_h$				$\partial_x^\alpha (K_H \star u_h)$				$\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\mathbb{P}^1$												
20	5.48E-01	–	1.76E+00	–	2.85E-01	–	1.49E+00	–	5.91E-01	–	2.54E+00	–
40	2.82E-01	0.96	1.05E+00	0.74	2.63E-01	0.11	1.57E+00	-0.08	4.90E-01	0.27	2.37E+00	0.10
80	1.37E-01	1.05	4.98E-01	1.08	2.11E-01	0.32	1.52E+00	0.05	4.17E-01	0.24	2.38E+00	-0.00
160	6.72E-02	1.02	2.57E-01	0.96	1.29E-01	0.71	1.18E+00	0.36	3.08E-01	0.43	2.30E+00	0.05
320	3.38E-02	0.99	1.30E-01	0.98	3.12E-02	2.05	3.75E-01	1.65	9.55E-02	1.69	9.77E-01	1.24
$\mathbb{P}^2$												
20	3.56E-02	–	2.01E-01	–	1.15E-02	–	8.74E-02	–	3.30E-02	–	1.35E-01	–
40	8.96E-03	1.99	5.80E-02	1.79	2.32E-03	2.31	1.90E-02	2.20	4.21E-03	2.97	2.41E-02	2.49
80	1.96E-03	2.20	1.16E-02	2.32	2.08E-03	0.16	1.49E-02	0.35	3.70E-03	0.19	2.34E-02	0.05
160	4.86E-04	2.01	3.88E-03	1.58	1.68E-03	0.30	1.49E-02	-0.00	3.16E-03	0.23	2.35E-02	-0.01
320	1.32E-04	1.88	8.95E-04	2.11	1.36E-03	0.30	1.50E-02	-0.01	2.63E-03	0.27	2.36E-02	-0.01
$\mathbb{P}^3$												
20	1.53E-03	–	1.10E-02	–	1.33E-02	–	7.20E-02	–	4.03E-02	–	2.58E-01	–
40	2.10E-04	2.86	1.72E-03	2.68	7.05E-05	7.56	3.40E-04	7.73	4.44E-05	9.83	3.07E-04	9.72
80	2.27E-05	3.21	1.80E-04	3.26	4.84E-06	3.86	3.58E-05	3.25	6.48E-06	2.78	4.29E-05	2.84
160	2.72E-06	3.06	2.52E-05	2.84	3.30E-06	0.55	2.89E-05	0.31	5.84E-06	0.15	4.53E-05	-0.08
320	3.42E-07	2.99	3.22E-06	2.97	2.71E-06	0.28	2.91E-05	-0.01	4.97E-06	0.23	4.55E-05	-0.01

where the variable coefficient is $a(x,t) = 2+\sin(2\pi(x+t))$ and the forcing term, $f(x,t)$, is chosen to make the exact solution

$$u(x,t) = \sin(2\pi(x-t)).$$

As with the linear example, we present the L^2 and L^∞ errors in Table 3.6 with the first three derivatives over Mesh 3.3.2. The respective point-wise error plots ($k = 2$ case) are shown in Figure 3.7. The results are similar to the results for the constant coefficient case. In order to save space we no longer repeat the descriptions, which are similar. However, we still want to point out one phenomenon. In this variable coefficient case, the filtered solution $\partial_x^\alpha (K_H \star u_h)$ shows somewhat better accuracy than the filtered solution $\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$ near the boundaries when $\alpha \leq k$. This performance suggests that when $\alpha \leq k$ we can consider not increasing the order of the B-splines, although it causes more oscillations in the error.

Remark 3.3.3. *Here we conclude by discussing the consequences of using B-splines of order $k+1$ compared to using the usual order $k+1+\alpha$; they are the following:*

- *it can give better accuracy near the boundaries;*
- *it can give better accuracy in the interior regions (when $\alpha \ll k$), but it damages the smoothness of filtered solution (more oscillations);*
- *it has a smaller support size;*
- *it allows the use of the symmetric filter over a larger area; and*

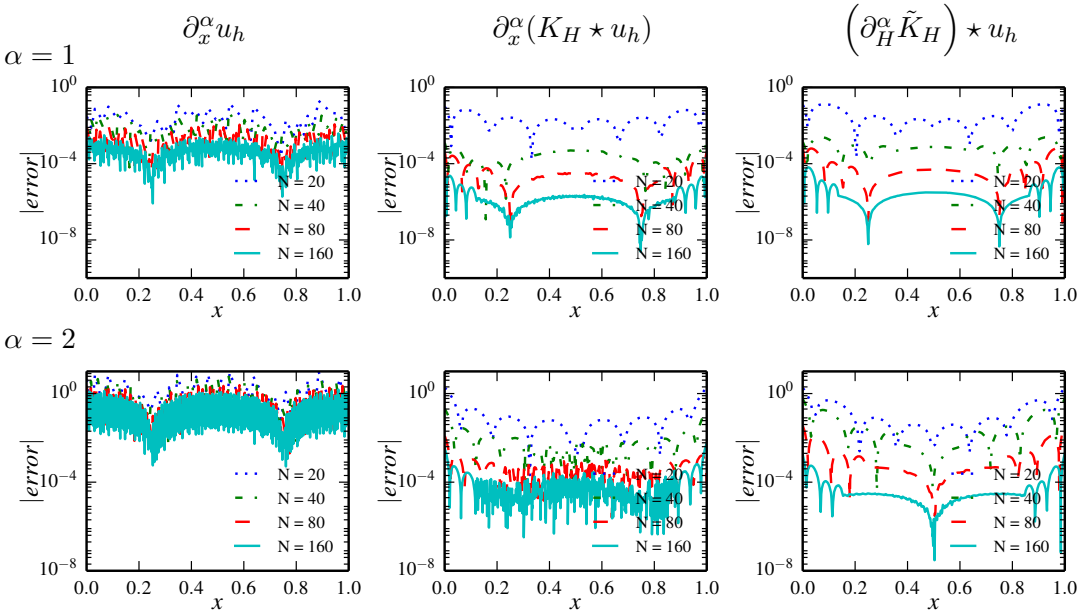


Figure 3.6: The point-wise errors in log scale for the first and second derivatives of DG approximation $\partial_x^\alpha u_h$ together with the two filtered solutions $\partial_x^\alpha (K_H \star u_h)$ and $\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$ (with the new filter) for linear convection equation (3.7) over Mesh 3.3.1. The filter scaling is taken as $H = h^{2/3}$, and $k = 2$.

- *it requires* $\alpha \leq k$.

Remark 3.3.4. *We notice that in Tables 3.5 and 3.6, the accuracy order is smaller than the conclusion in Theorem* **??**. *The reason is twofold:*

- *The first one is the effect of the boundary region. The error magnitude of the filtered solution in the interior region is much better than the error magnitude in the boundary region. Therefore, the accuracy order in the* L^2 *norm appears to be unstable and the numbers are smaller than the theoretical expectation. If we look at Figures 3.6 and 3.7 (right column), the convergence rates are stable respective to boundary and interior regions separately.*

- *The second reason is that the Meshes 3.3.1 and 3.3.2 are randomly generated. There is no strict refined relation among the meshes. Therefore, the accuracy order is affected by the randomness of the meshes, and a very stable accuracy order is not observed. One can see that once the effect caused by randomness becomes smaller in the two-dimensional example, the accuracy order becomes stable, see Tables 3.7 and 3.8.*

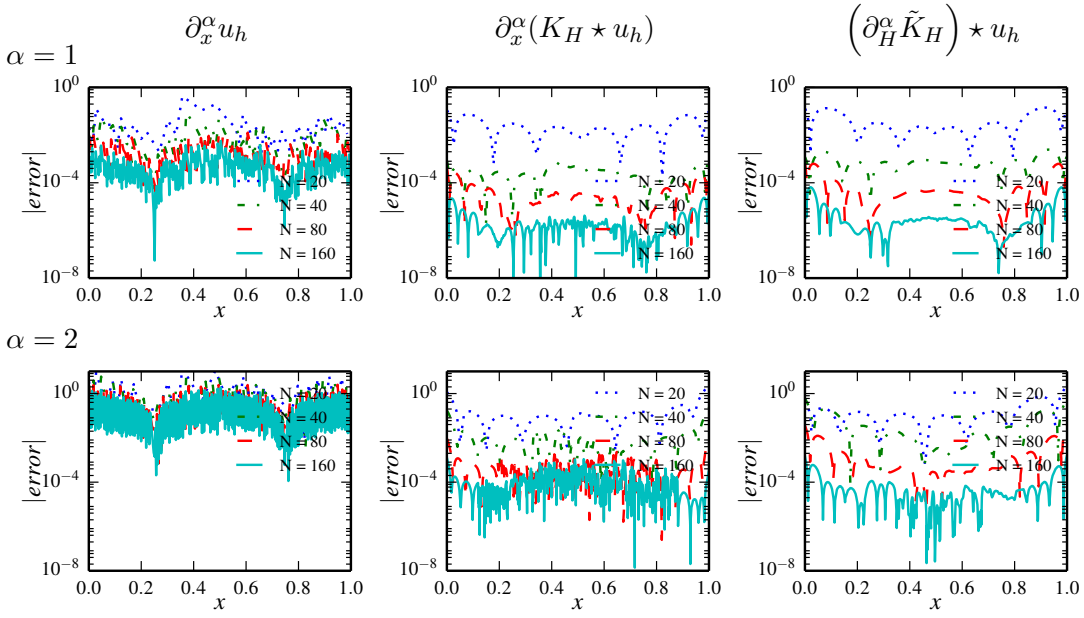


Figure 3.7: The point-wise errors in log scale of the first and second derivative of the DG approximation $\partial_x^\alpha u_h$ together with the two filtered solutions $\partial_x^\alpha (K_H \star u_h)$ and $\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$ (with the new filter) for variable coefficient equation (3.8) over Mesh 3.3.2. The filter scaling is taken as $H = h^{2/3}$, and $k = 2$.

3.4 Two-Dimensional Example

For the two-dimensional example, we consider a 2D version of the linear hyperbolic equation

$$\begin{aligned} u_t + u_x + u_y &= 0, \qquad (x, y) \in [0, 1] \times [0, 1], \\ u(x, y, 0) &= \sin(2\pi x + 2\pi y), \end{aligned} \tag{3.9}$$

at time $T = 1$. The nonuniform meshes we used are the 2D quadrilateral extension of Meshes 3.3.1 and 3.3.2. Here, we show the cross-derivative ∂_{xy}^2, the first derivatives ∂_x and ∂_y are omitted as they are similar to the 1D results. We give the L^2 and L^∞ error in Tables 3.7 - 3.8 and the point-wise error plots in Figures 3.8 - 3.9. We note that the filtered accuracy error seems slightly worse than the DG approximation over coarse meshes, because near the boundary regions we need sufficiently refined meshes to show the advantage of the position-dependent filter. Once the mesh is sufficient refined, we see better results. We also note that although we require a relatively refined mesh for boundary regions, the results in the interior regions are always much better (see the point-wise error plots Figures 3.8 and 3.9).

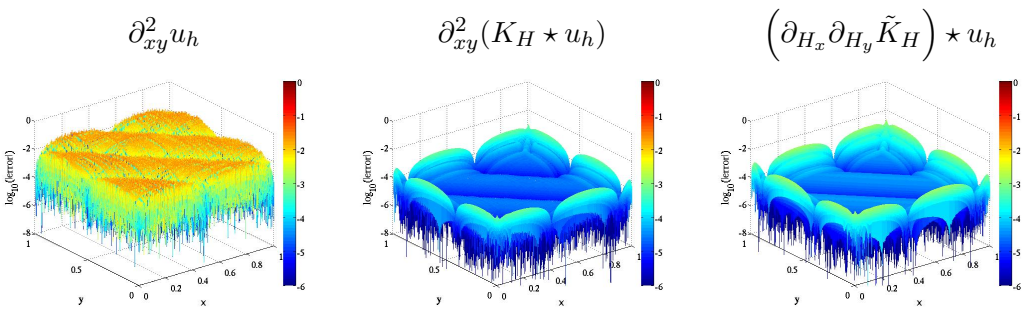


Figure 3.8: The point-wise errors in log scale of the cross-derivative DG approximation $\partial^2_{xy} u_h$ together with the filtered solution $\partial^2_{xy} u^\star_h$ for the two-dimensional linear equation (3.9) over Mesh 3.3.1 (2D, $N = 160 \times 160$).

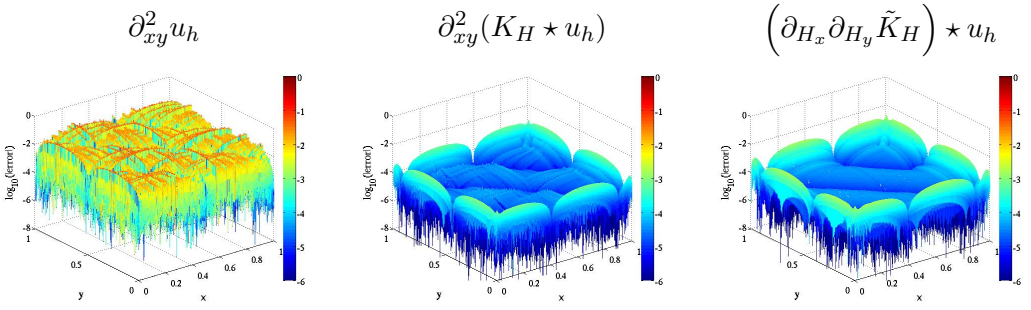


Figure 3.9: The point-wise errors in log scale of the cross-derivative DG approximation $\partial^2_{xy} u_h$ together with the filtered solution $\partial^2_{xy} u^\star_h$ for the two-dimensional linear equation (3.9) over Mesh 3.3.1 (2D, $N = 160 \times 160$).

3.5 Conclusion

In this chapter, we have proposed three position-dependent derivative filter, to approximate the derivatives of a discontinuous Galerkin solution over uniform and nonuniform meshes. These position-dependent derivative filters allow us to obtain more accurate derivatives of the DG solutions compared to calculating the derivatives of DG solutions directly. The derivative SRV filter uses $4k + 1$ central B-splines, and obtains a convergence rate of $2k+1$ over uniform meshes regardless of derivative order. The new position-dependent derivative filter uses $2k + 1$ central B-splines and an extra general B-spline, where the B-splines rely on the derivative order α. We have proved that the new position-dependent derivative filter has accuracy order of $\mu(2k + 2)$ when using filter scaling $H = h^\mu$ ($\mu \approx 2/3$). Additionally, we are able, for the first time, to extend the symmetric derivative filter to nonuniform meshes. Through numerical examples, we compared the derivative SRV and new filter over uniform and nonuniform meshes.

We demonstrated that once the required conditions are satisfied the derivative SRV filter has a better performance over uniform meshes compared to the new derivative filter. However, for nonuniform meshes, only the new derivative filter can maintain its performance and improve the accuracy of the DG approximations. Also, we compared derivative filters with different orders of B-splines: order $k+1$ and order $k+1+\alpha$. Numerical results indicate that using B-splines of order $k+1$ may improve the accuracy of the filtered solution near the boundaries. For interior regions where the symmetric derivative filtering is applied, using B-splines of order $k+1+\alpha$ shows better accuracy and smoothness. Lastly, we point out that for given nonuniform meshes there may exist a better scaling that allows us to obtain better results.

Our new contributions are:

- Testing the position-dependent derivative filters for uniform meshes, which was not previously accomplished before;
- Applying the symmetric and position-dependent derivative filters over different nonuniform meshes.

Table 3.5: L^2- and $L^\infty-$errors for the αth derivative of the DG approximation $\partial_x^\alpha u_h$ together with the two filtered solutions $\partial_x^\alpha (K_H \star u_h)$ and $\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$ (with the new filter) for linear convection equation (3.7), over Mesh 3.3.1. The filter scaling is taken as $H = h^{2/3}$.

	$\partial_x^\alpha u_h$				$\partial_x^\alpha (K_H \star u_h)$				$\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\boldsymbol{\alpha = 1}$					$\mathbb{P}^1$							
20	5.48E-01	–	1.76E+00	–	4.19E-02	–	1.52E-01	–	5.36E-02	–	9.92E-02	–
40	2.82E-01	0.96	1.05E+00	0.74	8.18E-03	2.36	3.36E-02	2.18	1.14E-02	2.23	4.02E-02	1.30
80	1.37E-01	1.05	4.98E-01	1.08	1.89E-03	2.11	6.12E-03	2.46	2.19E-03	2.38	8.22E-03	2.29
160	6.72E-02	1.02	2.57E-01	0.96	4.93E-04	1.94	2.09E-03	1.55	3.51E-04	2.64	1.44E-03	2.51
320	3.38E-02	0.99	1.30E-01	0.98	1.46E-04	1.76	6.15E-04	1.76	5.04E-05	2.80	2.35E-04	2.61
					$\mathbb{P}^2$							
20	3.56E-02	–	2.01E-01	–	3.13E-02	–	9.60E-02	–	5.84E-02	–	1.32E-01	–
40	8.96E-03	1.99	5.80E-02	1.79	3.22E-04	6.60	1.26E-03	6.25	1.04E-03	5.82	2.45E-03	5.75
80	1.96E-03	2.20	1.16E-02	2.32	7.59E-05	2.08	4.26E-04	1.57	1.78E-04	2.54	6.73E-04	1.86
160	4.86E-04	2.01	3.88E-03	1.58	5.28E-06	3.85	3.78E-05	3.49	1.46E-05	3.61	6.65E-05	3.34
320	1.32E-04	1.88	8.95E-04	2.11	3.20E-07	4.04	2.60E-06	3.86	8.71E-07	4.07	4.83E-06	3.78
					$\mathbb{P}^3$							
20	1.53E-03	–	1.10E-02	–	4.08E-03	–	1.23E-02	–	5.34E-03	–	1.42E-02	–
40	2.10E-04	2.86	1.72E-03	2.68	8.01E-04	2.35	2.63E-03	2.22	2.89E-03	0.88	7.96E-03	0.83
80	2.27E-05	3.21	1.80E-04	3.26	4.79E-06	7.38	2.39E-05	6.78	3.10E-06	9.87	1.44E-05	9.11
160	2.72E-06	3.06	2.52E-05	2.84	3.62E-07	3.73	2.03E-06	3.56	9.36E-07	1.73	4.15E-06	1.79
320	3.42E-07	2.99	3.22E-06	2.97	9.64E-09	5.23	6.87E-08	4.89	2.71E-08	5.11	1.51E-07	4.78
$\boldsymbol{\alpha = 2}$					$\mathbb{P}^1$							
20	–	–	–	–	–	–	–	–	1.94E+00	–	9.16E+00	–
40	–	–	–	–	–	–	–	–	2.63E-01	2.89	1.51E+00	2.60
80	–	–	–	–	–	–	–	–	3.42E-02	2.94	1.99E-01	2.93
160	–	–	–	–	–	–	–	–	6.39E-03	2.42	2.11E-02	3.23
320	–	–	–	–	–	–	–	–	2.19E-03	1.54	8.55E-03	1.30
					$\mathbb{P}^2$							
20	3.16E+00	–	9.99E+00	–	3.19E-01	–	1.83E+00	–	3.42E-01	–	1.80E+00	–
40	1.60E+00	0.98	5.79E+00	0.79	2.87E-02	3.47	2.05E-01	3.16	1.00E-01	1.77	5.59E-01	1.69
80	7.57E-01	1.08	2.60E+00	1.16	1.11E-03	4.69	1.21E-02	4.07	5.25E-03	4.25	3.72E-02	3.91
160	3.78E-01	1.00	1.52E+00	0.78	5.10E-04	1.12	5.79E-03	1.07	2.07E-04	4.66	1.96E-03	4.24
320	1.96E-01	0.94	7.38E-01	1.04	2.65E-05	4.26	2.02E-04	4.85	8.54E-06	4.60	4.75E-05	5.37
					$\mathbb{P}^3$							
20	2.15E-01	–	1.12E+00	–	2.11E-02	–	1.39E-01	–	2.21E-02	–	1.21E-01	–
40	5.70E-02	1.92	3.46E-01	1.70	9.02E-03	1.23	6.45E-02	1.11	2.23E-02	-0.01	1.25E-01	-0.06
80	1.31E-02	2.12	7.71E-02	2.17	2.40E-04	5.23	2.13E-03	4.92	1.15E-03	4.28	7.69E-03	4.03
160	3.17E-03	2.05	2.05E-02	1.91	3.76E-06	6.00	4.21E-05	5.66	2.06E-05	5.80	1.71E-04	5.49
320	7.98E-04	1.99	5.27E-03	1.96	5.25E-08	6.16	7.68E-07	5.78	2.93E-07	6.14	3.10E-06	5.79
$\boldsymbol{\alpha = 3}$					$\mathbb{P}^2$							
20	–	–	–	–	–	–	–	–	4.98E+00	–	2.58E+01	–
40	–	–	–	–	–	–	–	–	1.01E+00	2.31	5.55E+00	2.22
80	–	–	–	–	–	–	–	–	3.06E-02	5.04	2.73E-01	4.35
160	–	–	–	–	–	–	–	–	3.25E-03	3.24	2.95E-02	3.21
320	–	–	–	–	–	–	–	–	1.64E-03	0.99	1.66E-02	0.83
					$\mathbb{P}^3$							
20	1.95E+01	–	5.77E+01	–	2.74E-01	–	1.78E+00	–	3.40E-01	–	2.21E+00	–
40	9.94E+00	0.97	3.54E+01	0.70	7.63E-02	1.84	5.36E-01	1.73	3.45E-01	-0.02	2.12E+00	0.06
80	4.81E+00	1.05	1.67E+01	1.08	1.59E-03	5.59	1.51E-02	5.15	5.12E-03	6.07	3.62E-02	5.87
160	2.37E+00	1.02	8.62E+00	0.96	3.54E-04	2.17	4.15E-03	1.87	1.81E-04	4.83	1.73E-03	4.39
320	1.19E+00	0.99	4.37E+00	0.98	1.42E-05	4.64	2.12E-04	4.29	6.44E-06	4.81	7.71E-05	4.49

Table 3.6: L^2- and $L^\infty-$errors for the αth derivative of the DG approximation $\partial_x^\alpha u_h$ together with the two filtered solutions $\partial_x^\alpha (K_H \star u_h)$ and $\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$ (with the new filter) for variable coefficient equation (3.8), over Mesh 3.3.2. The filter scaling is taken as $H = h^{2/3}$.

	$\partial_x^\alpha u_h$				$\partial_x^\alpha (K_H \star u_h)$				$\left(\partial_H^\alpha \tilde{K}_H\right) \star u_h$			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\boldsymbol{\alpha = 1}$						$\mathbb{P}^1$						
20	5.73E-01	–	2.04E+00	–	4.28E-02	–	9.17E-02	–	4.04E-02	–	8.51E-02	–
40	2.76E-01	1.05	9.98E-01	1.03	1.29E-02	1.73	6.62E-02	0.47	1.47E-02	1.46	5.53E-02	0.62
80	1.53E-01	0.85	6.35E-01	0.65	3.44E-03	1.91	1.31E-02	2.34	2.72E-03	2.43	7.98E-03	2.79
160	7.16E-02	1.10	3.43E-01	0.89	1.01E-03	1.76	5.63E-03	1.22	6.26E-04	2.12	1.82E-03	2.13
320	4.07E-02	0.82	2.46E-01	0.48	9.81E-04	0.05	8.33E-03	-0.57	6.37E-04	-0.02	3.11E-03	-0.77
						$\mathbb{P}^2$						
20	6.60E-02	–	4.27E-01	–	3.38E-02	–	1.10E-01	–	6.16E-02	–	1.42E-01	–
40	1.27E-02	2.38	9.22E-02	2.21	3.17E-04	6.73	1.26E-03	6.45	9.68E-04	5.99	2.46E-03	5.85
80	2.12E-03	2.58	1.27E-02	2.87	7.71E-05	2.04	4.28E-04	1.55	1.78E-04	2.44	6.61E-04	1.89
160	6.40E-04	1.73	5.66E-03	1.16	5.33E-06	3.85	3.71E-05	3.53	1.47E-05	3.60	6.54E-05	3.34
320	2.48E-04	1.37	2.99E-03	0.92	4.67E-07	3.51	2.70E-06	3.78	8.72E-07	4.08	4.93E-06	3.73
						$\mathbb{P}^3$						
20	2.95E-03	–	2.29E-02	–	4.28E-03	–	1.29E-02	–	5.72E-03	–	1.52E-02	–
40	2.88E-04	3.35	2.84E-03	3.01	8.01E-04	2.42	2.61E-03	2.31	2.89E-03	0.98	7.97E-03	0.93
80	3.95E-05	2.87	3.84E-04	2.89	4.82E-06	7.38	2.36E-05	6.79	3.10E-06	9.87	1.41E-05	9.15
160	4.74E-06	3.06	6.15E-05	2.64	3.62E-07	3.73	2.05E-06	3.53	9.36E-07	1.73	4.15E-06	1.76
320	1.28E-06	1.89	2.14E-05	1.52	9.64E-09	5.23	6.95E-08	4.88	2.71E-08	5.11	1.51E-07	4.78
$\boldsymbol{\alpha = 2}$						$\mathbb{P}^1$						
20	–	–	–	–	–	–	–	–	2.45E+00	–	1.14E+01	–
40	–	–	–	–	–	–	–	–	4.39E-01	2.48	2.61E+00	2.12
80	–	–	–	–	–	–	–	–	6.57E-02	2.74	2.23E-01	3.55
160	–	–	–	–	–	–	–	–	3.48E-02	0.92	1.13E-01	0.98
320	–	–	–	–	–	–	–	–	5.99E-02	-0.78	2.80E-01	-1.31
						$\mathbb{P}^2$						
20	3.89E+00	–	1.40E+01	–	3.34E-01	–	2.36E+00	–	3.39E-01	–	2.26E+00	–
40	1.78E+00	1.13	6.49E+00	1.10	2.93E-02	3.51	2.23E-01	3.40	1.03E-01	1.72	5.84E-01	1.95
80	7.66E-01	1.22	2.54E+00	1.35	1.11E-03	4.72	9.19E-03	4.60	5.31E-03	4.28	3.73E-02	3.97
160	4.10E-01	0.90	1.86E+00	0.45	2.28E-04	2.28	1.26E-03	2.87	1.86E-04	4.84	1.62E-03	4.52
320	2.28E-01	0.85	1.36E+00	0.45	2.00E-04	0.19	1.38E-03	-0.14	1.39E-05	3.74	8.73E-05	4.22
						$\mathbb{P}^3$						
20	2.89E-01	–	1.67E+00	–	2.16E-02	–	1.63E-01	–	1.84E-02	–	1.13E-01	–
40	6.24E-02	2.21	4.20E-01	1.99	9.03E-03	1.26	6.49E-02	1.33	2.22E-02	-0.28	1.25E-01	-0.15
80	1.80E-02	1.79	1.27E-01	1.72	2.35E-04	5.27	2.09E-03	4.96	1.15E-03	4.28	7.60E-03	4.04
160	4.12E-03	2.13	3.67E-02	1.79	3.66E-06	6.00	4.28E-05	5.61	2.06E-05	5.80	1.73E-04	5.46
320	1.51E-03	1.45	1.79E-02	1.03	1.06E-07	5.10	7.61E-07	5.81	2.93E-07	6.14	3.11E-06	5.80
$\boldsymbol{\alpha = 3}$						$\mathbb{P}^2$						
20	–	–	–	–	–	–	–	–	6.75E+00	–	4.36E+01	–
40	–	–	–	–	–	–	–	–	1.18E+00	2.51	7.23E+00	2.59
80	–	–	–	–	–	–	–	–	2.36E-02	5.65	2.19E-01	5.04
160	–	–	–	–	–	–	–	–	7.55E-03	1.64	7.38E-02	1.57
320	–	–	–	–	–	–	–	–	1.28E-03	2.57	7.79E-03	3.24
						$\mathbb{P}^3$						
20	2.04E+01	–	6.86E+01	–	1.68E+00	–	1.10E+01	–	2.37E+00	–	1.56E+01	–
40	9.76E+00	1.06	3.31E+01	1.05	9.36E-02	4.17	6.06E-01	4.18	3.52E-01	2.75	2.16E+00	2.85
80	5.39E+00	0.86	2.14E+01	0.63	4.87E-04	7.59	3.77E-03	7.33	5.15E-03	6.10	3.73E-02	5.86
160	2.52E+00	1.10	1.15E+01	0.89	1.55E-04	1.65	1.84E-03	1.04	1.76E-04	4.87	1.73E-03	4.43
320	1.39E+00	0.86	7.30E+00	0.66	6.97E-05	1.16	5.24E-04	1.81	6.60E-06	4.74	7.91E-05	4.45

Table 3.7: L^2- and $L^\infty-$errors for the cross-derivative DG approximation $\partial^2_{xy}u_h$ together with the filtered solution $\partial^2_{xy}u^\star_h$ for the two-dimensional linear equation (3.9) over Mesh 3.3.1 (2D).

	$\partial^2_{xy}u_h$				$\partial^2_{xy}(K_H \star u_h)$				$\partial_{H_x}\partial_{H_y}\bar{K}_H \star u_h$			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\mathbb{P}^1$												
20×20	5.47E+00	–	2.32E+01	–	–	–	–	–	1.32E+00	–	1.25E+01	–
40×40	2.71E+00	1.01	1.33E+01	0.81	–	–	–	–	1.97E-01	2.75	1.80E+00	2.79
80×80	1.33E+00	1.03	6.39E+00	1.06	–	–	–	–	2.81E-02	2.81	2.56E-01	2.81
160×160	6.62E-01	1.00	3.38E+00	0.92	–	–	–	–	4.08E-03	2.78	3.39E-02	2.92
$\mathbb{P}^2$												
20×20	3.48E-01	–	2.49E+00	–	4.68E-01	–	3.44E+00	–	5.66E-01	–	3.59E+00	–
40×40	8.16E-02	2.09	7.13E-01	1.80	2.65E-02	4.14	3.63E-01	3.24	5.38E-02	3.40	6.81E-01	2.40
80×80	1.93E-02	2.08	1.81E-01	1.98	1.38E-03	4.26	1.96E-02	4.22	2.83E-03	4.25	4.03E-02	4.08
160×160	4.79E-03	2.01	4.53E-02	2.00	6.86E-05	4.33	8.74E-04	4.48	1.44E-04	4.30	1.84E-03	4.45
$\mathbb{P}^3$												
20×20	1.54E-02	–	1.47E-01	–	4.11E-02	–	2.79E-01	–	4.06E-02	–	2.63E-01	–
40×40	1.75E-03	3.13	2.29E-02	2.68	1.14E-02	1.85	1.27E-01	1.13	2.45E-02	0.73	2.04E-01	0.37
80×80	2.00E-04	3.13	2.51E-03	3.19	2.42E-04	5.56	4.26E-03	4.90	5.30E-04	5.53	8.51E-03	4.58
160×160	2.47E-05	3.02	3.58E-04	2.81	4.91E-06	5.62	8.59E-05	5.63	1.08E-05	5.62	1.81E-04	5.56

Table 3.8: L^2- and $L^\infty-$errors for the cross-derivative DG approximation $\partial^2_{xy}u_h$ together with the filtered solution $\partial^2_{xy}u^\star_h$ for the two-dimensional linear equation (3.9) over Mesh 3.3.2 (2D).

	$\partial^2_{xy}u_h$				$\partial^2_{xy}(K_H \star u_h)$				$\partial_{H_x}\partial_{H_y}\bar{K}_H \star u_h$			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\mathbb{P}^1$												
20×20	6.03E+00	–	2.94E+01	–	–	–	–	–	1.53E+00	–	1.27E+01	–
40×40	3.20E+00	0.91	1.95E+01	0.59	–	–	–	–	2.72E-01	2.50	2.07E+00	2.62
80×80	1.61E+00	0.99	1.09E+01	0.84	–	–	–	–	5.44E-02	2.32	4.01E-01	2.37
160×160	7.39E-01	1.12	5.18E+00	1.07	–	–	–	–	8.35E-03	2.70	1.37E-01	1.55
$\mathbb{P}^2$												
20×20	5.60E-01	–	7.00E+00	–	4.73E-01	–	3.48E+00	–	5.68E-01	–	3.59E+00	–
40×40	1.68E-01	1.73	2.65E+00	1.40	2.67E-02	4.15	3.67E-01	3.24	5.34E-02	3.41	6.88E-01	2.38
80×80	3.85E-02	2.13	5.30E-01	2.32	1.36E-03	4.29	1.94E-02	4.24	2.98E-03	4.16	4.12E-02	4.06
160×160	7.27E-03	2.41	1.03E-01	2.37	7.90E-05	4.11	1.05E-03	4.21	1.54E-04	4.27	1.89E-03	4.44
$\mathbb{P}^3$												
20×20	4.02E-02	–	3.66E-01	–	4.12E-02	–	2.79E-01	–	4.05E-02	–	2.64E-01	–
40×40	7.60E-03	2.40	1.03E-01	1.83	1.14E-02	1.86	1.29E-01	1.12	2.45E-02	0.72	2.06E-01	0.36
80×80	7.71E-04	3.30	1.68E-02	2.61	2.42E-04	5.55	4.21E-03	4.93	5.30E-04	5.53	8.40E-03	4.62
160×160	5.76E-05	3.74	1.31E-03	3.69	4.91E-06	5.62	8.65E-05	5.60	1.08E-05	5.62	1.82E-04	5.51

4

SIAC Filters over Nonuniform Meshes

In practical applications, there are strong motivators for the adoption of unstructured meshes for handling complex geometries and using adaptive mesh refinement techniques. Based on this practical necessity, it is widely believed that discontinuous Galerkin methods, which provide high-order accuracy on unstructured meshes, will become one of the standard numerical methods for future generations. However, SIAC filters are still limited primarily to structured meshes. For general nonuniform meshes, the quality of the filtered solution is usually unsatisfactory. The ability to deal with nonuniform meshes is an obstacle to the further development of SIAC filters.

In this chapter, we focus on applying the SIAC filter for DG solutions over nonuniform meshes. Specifically, this study focuses on the barrier to applying the SIAC filter to nonuniform meshes – the scaling. We establish a relation between the filtered solutions and the unstructuredness of nonuniform meshes. Further, we demonstrate that there exist an optimal accuracy of the filtered solution for a given nonuniform mesh, and it is possible to approximate the optimal accuracy by the method we propose. By applying the newly designed SIAC filter over nonuniform meshes, the filtered solution has demonstrated improvement in accuracy order as well as improve the quality of the numerical solution. The key concept in the extension to unstructured meshes is understanding the divided differences, as discussed in the following section.

4.1 Divided Differences: Uniform Meshes

In order to use the SIAC filter to improve the quality of DG solutions over nonuniform meshes, we need to study the theoretical challenges of using the SIAC filter, namely the divided differences of DG solutions. To get full appreciation of the situation, we begin discussing from the uniform meshes in this section.

As presented in Chapter 1, Theorem 1.2.1 is the theoretical foundation of using SIAC filters for DG approximations. Theorem 1.2.1 shows that the divided differences of the DG solution have the same accuracy order as the DG solution itself in the L^2 norm (1.3) and the negative order norm (1.4). When the divided difference of order $\alpha = 0$, the conclusion is well known and studied. Nevertheless, the estimates for $\alpha > 0$ are equally important for the theoretical foundations of the SIAC filter. In [25], after

presenting the proof for the $\alpha = 0$ case, the authors simply claimed the conclusion for $\alpha > 0$ case also holds for translation invariant meshes without presenting details. In order to study the divided differences of the DG approximations over nonuniform meshes, we have to understand what happens over uniform meshes.

For writing convenience, instead of considering equation (1.1), we only write the analysis for the simplest one-dimensional linear hyperbolic equation,

$$\begin{aligned} &u_t + u_x = 0, \quad (x,t) \in [0,1] \times (0,T] \\ &u(x,0) = u_0(x), \end{aligned} \tag{4.1}$$

where u_0 is sufficiently smooth and the domain $\Omega = [0,1]$ is covered by a uniform mesh $\{I_j\}_{j=1}^N$, where $I_j = [x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}]$ with size $h = \frac{1}{N}$. Then, the DG scheme of equation (4.1) is given by

$$\begin{aligned} \int_{I_j} (u_h)_t v_h dx - \int_{I_j} u_h (v_h)_x dx \\ + u_h(x^-_{j+\frac{1}{2}}) v_h(x^-_{j+\frac{1}{2}}) - u_h(x^-_{j-\frac{1}{2}}) v_h(x^+_{j-\frac{1}{2}}) = 0, \end{aligned} \tag{4.2}$$

where $u_h, v_h \in V_h^k = \left\{\varphi \in L^2(\Omega) : \varphi|_{I_j} \in \mathbb{P}^k,\, j = 1, \ldots, N\right\}$, and the upwind flux is used.

4.1.1 Scaling h: $\partial_h u_h$

Consider the DG scheme (4.2) over $I_{j-\frac{1}{2}} = [x_{j-1}, x_j]$ and $I_{j+\frac{1}{2}} = [x_j, x_{j+\frac{1}{2}}]$. This can be written as

$$\begin{aligned} \int_{I_{j-\frac{1}{2}}} \left(u_h(x+\frac{h}{2})\right)_t v_h(x+\frac{h}{2}) dx - \int_{I_{j-\frac{1}{2}}} u_h(x+\frac{h}{2}) \left(v_h(x+\frac{h}{2})\right)_x dx \\ + u_h(x^-_{j+\frac{1}{2}}) v_h(x^-_{j+\frac{1}{2}}) - u_h(x^-_{j-\frac{1}{2}}) v_h(x^+_{j-\frac{1}{2}}) = 0, \end{aligned}$$

$$\begin{aligned} \int_{I_{j+\frac{1}{2}}} \left(u_h(x-\frac{h}{2})\right)_t v_h(x-\frac{h}{2}) dx - \int_{I_{j+\frac{1}{2}}} u_h(x-\frac{h}{2}) \left(v_h(x-\frac{h}{2})\right)_x dx \\ + u_h(x^-_{j+\frac{1}{2}}) v_h(x^-_{j+\frac{1}{2}}) - u_h(x^-_{j-\frac{1}{2}}) v_h(x^+_{j-\frac{1}{2}}) = 0. \end{aligned}$$

Since the mesh is uniform, and h is the mesh element size, with the periodic boundary condition we know the space $V_h^k(x-\frac{h}{2})$ and $V_h^k(x+\frac{h}{2})$ are the same piecewise polynomial space, see Figure 4.1. Denote this space as

$$\tilde{V}_h^k = \left\{\varphi \in L^2(\Omega) : \varphi|_{I_{j-\frac{1}{2}}} \in \mathbb{P}^k,\, j = 1, \ldots, N\right\}.$$

Then we can rewrite the previous formula as

$$\begin{aligned} \int_{I_{j-\frac{1}{2}}} \left(u_h(x+\frac{h}{2})\right)_t \tilde{v}_h(x) dx - \int_{I_{j-\frac{1}{2}}} u_h(x+\frac{h}{2}) \left(\tilde{v}_h(x)\right)_x dx \\ + u_h(x^-_{j+\frac{1}{2}}) \tilde{v}_h(x^-_j) - u_h(x^-_{j-\frac{1}{2}}) \tilde{v}_h(x^+_{j-1}) = 0, \end{aligned}$$

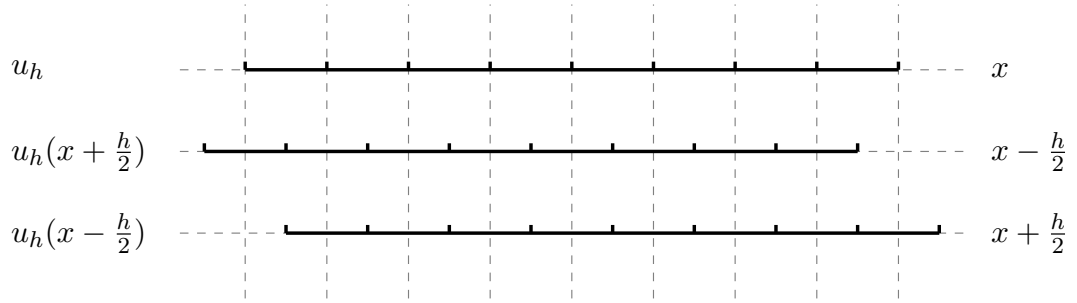


Figure 4.1: The top mesh is for the original DG solution u_h over a uniform mesh x, the middle mesh is the value $u_h(x+\frac{h}{2})$ which shifts the mesh x by $-\frac{h}{2}$, the bottom mesh is the value $u_h(x-\frac{h}{2})$ which shifts the mesh x by $\frac{h}{2}$.

$$\int_{I_{j-\frac{1}{2}}} \left(u_h(x-\frac{h}{2})\right)_t \tilde{v}_h(x)dx - \int_{I_{j-\frac{1}{2}}} u_h(x-\frac{h}{2})\left(\tilde{v}_h(x)\right)_x dx + u_h(x^-_{j-\frac{1}{2}})\tilde{v}_h(x^-_j) - u_h(x^-_{j-\frac{3}{2}})\tilde{v}_h(x^+_{j-1}) = 0,$$

where $\tilde{v}_h \in \tilde{V}_h^k$. Subtracting the above two formulas and dividing by h, we obtain

$$\begin{aligned} &B_{j-\frac{1}{2}}(\partial_h u_h; \tilde{v}_h) \\ &= \int_{I_{j-\frac{1}{2}}} (\partial_h u_h(x))_t\, \tilde{v}_h(x)dx - \int_{I_{j-\frac{1}{2}}} \partial_h u_h(x)\left(\tilde{v}_h(x)\right)_x dx \\ &\quad + \partial_h u_h(x^-_j)\tilde{v}_h(x^-_j) - \partial_h u_h(x^-_{j-1})\tilde{v}_h(x^+_{j-1}) = 0, \end{aligned} \tag{4.3}$$

where $\partial_h u_h(x^-_j) = \left(u_h(x^-_{j+\frac{1}{2}}) - u_h(x^-_{j-\frac{1}{2}})\right)/h$.

Proposition 4.1.1 (Cell Entropy Inequality)**.** *The solution $\partial_h u_h$ to scheme* (4.3) *satisfies the following cell entropy inequality*

$$\frac{d}{dt}\int_{I_{j-\frac{1}{2}}} U(\partial_h u_h)dx + \hat{F}_j - \hat{F}_{j-1} \le 0,$$

where the entropy $U(\partial_h u) = \frac{1}{2}\left(\partial u_h\right)^2$, for entropy flux $\hat{F}_j = \frac{1}{2}(\partial_h u_h)^2(x^-_j)$.

Proof. Taking $\tilde{v}_h = \partial_h u_h$ in (4.3),

$$\begin{aligned} B_{j-\frac{1}{2}}(\partial_h u_h; \partial_h u_h) &= \int_{I_{j-\frac{1}{2}}} U(\partial_h u_h(x))_t dx - \frac{1}{2}(\partial_h u_h(x))^2|_{\partial I_{j-\frac{1}{2}}} dx \\ &\quad + (\partial_h u_h(x^-_j))^2 - \partial_h u_h(x^-_{j-1})\partial_h u_h(x^+_{j-1}) \\ &= \frac{d}{dt}\int_{I_{j-\frac{1}{2}}} U(\partial_h u_h(x))dx + \hat{F}_j - \hat{F}_{j-1} + \Theta_{j-1} = 0 \end{aligned}$$

where

$$\Theta_{j-1} = \frac{1}{2}\left(\partial_h u_h(x_{j-1}^+) - \partial_h u_h(x_{j-1}^-)\right)^2 \geq 0$$

This completes the proof of the cell entropy inequality. □

Similar to the primary value u_h, we also have

Proposition 4.1.2 (L^2 stability)**.** *For periodic boundary conditions, the divided difference, $\partial_h u_h$, to scheme* (4.3) *satisfies the following L^2 stability condition*

$$\frac{d}{dt}\int_0^1 (\partial_h u_h)^2\, dx \leq 0 \quad or \quad \|\partial_h u_h(\cdot,t)\| \leq \|\partial_h u_h(\cdot,0)\|.$$

Proposition 4.1.3 (Error Estimate)**.** *The divided difference $\partial_h u_h$ of DG scheme* (4.2) *for equation* (4.1) *with a smooth solution u satisfies the following error estimates. In the L^2 norm:*

$$\|\partial_h u - \partial_h u_h\|_{0,\Omega} \leq Ch^{k+1},$$

and in the negative order norm:

$$\|\partial_h u - \partial_h u_h\|_{-(k+1),\Omega} \leq Ch^{k+1},$$

where C depends on u and its derivatives but is independent of h.

Remark 4.1.1. *Using induction, the above propositions also hold for $\alpha > 1$.*

4.1.2 Constant Scaling H: $\partial_H u_h$

In the previous analysis, we give the error estimate of the divided difference of the DG solution $\partial_h u_h$ with the scaling h, where h is the uniform mesh size. In this section, we investigate the behavior of using a general constant scaling H. Based on the DG scheme (4.2), similar to using scaling h, we have

$$\begin{aligned}\int_{I_j-\frac{H}{2}} \left(u_h(x+\frac{H}{2})\right)_t v_h(x+\frac{H}{2})dx - \int_{I_j-\frac{H}{2}} u_h(x+\frac{H}{2})\left(v_h(x+\frac{H}{2})\right)_x dx \\ + u_h(x^-_{j+\frac{1}{2}})v_h(x^-_{j+\frac{1}{2}}) - u_h(x^-_{j-\frac{1}{2}})v_h(x^+_{j-\frac{1}{2}}) = 0,\end{aligned}$$

$$\begin{aligned}\int_{I_j+\frac{H}{2}} \left(u_h(x-\frac{H}{2})\right)_t v_h(x-\frac{H}{2})dx - \int_{I_j+\frac{H}{2}} u_h(x-\frac{H}{2})\left(v_h(x-\frac{H}{2})\right)_x dx \\ + u_h(x^-_{j+\frac{1}{2}})v_h(x^-_{j+\frac{1}{2}}) - u_h(x^-_{j-\frac{1}{2}})v_h(x^+_{j-\frac{1}{2}}) = 0,\end{aligned}$$

If $H = h$, then the situation will follow the previous analysis, we can subtract the above two formulas directly. However, if $H \neq h$, then things become complicated. We can not subtract them directly like in the $H = h$ case, since the space $V_h^k(x-\frac{H}{2})$ and $V_h^k(x+\frac{H}{2})$ are no longer the same space, see Figure 4.2. Once the spaces $V_h^k(x-\frac{H}{2})$ and $V_h^k(x+\frac{H}{2})$ are not the same space, the traditional analysis techniques can not be performed. Therefore, in order to obtain a scheme for $\partial_H u_h$, we have to require the

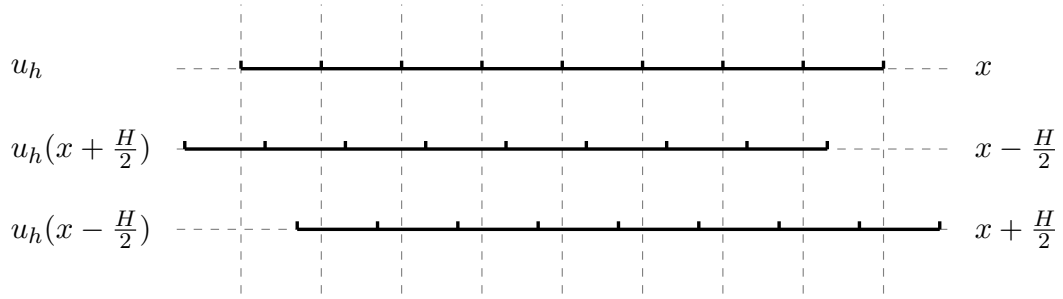


Figure 4.2: The top mesh is the original DG solution u_h over a uniform mesh x, the middle mesh is the value $u_h(x+\frac{H}{2})$ which means shifting the mesh x by $-\frac{H}{2}$, the bottom mesh is the value $u_h(x-\frac{H}{2})$ which shifts the mesh x by $\frac{H}{2}$.

scaling H to satisfy some other conditions. For uniform mesh with periodic conditions, we need a scaling H which makes the space $V_h^k(x-\frac{H}{2})$ and $V_h^k(x+\frac{H}{2})$ be the same polynomial space. This requires

$$I_j - \frac{H}{2} + mh = I_j + \frac{H}{2},$$

where m is a positive integer. The above relation is equivalent to

$$H = mh.$$

Figure 4.3 shows a example with the scaling $H = 2h$.

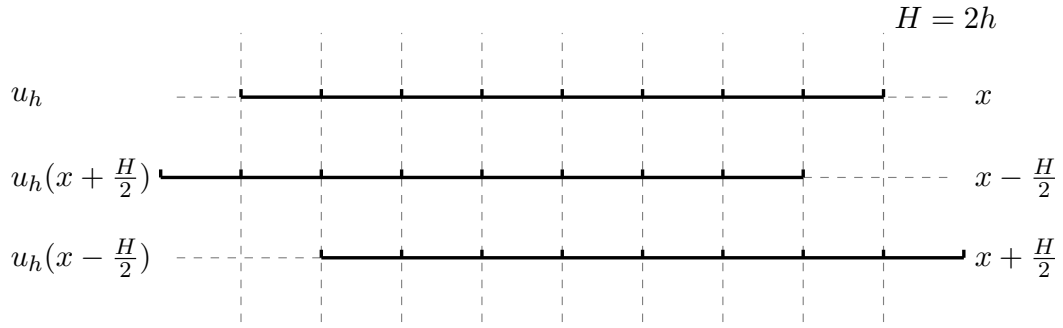


Figure 4.3: The top mesh is the original DG solution u_h over a uniform mesh x, the middle mesh is the value $u_h(x+\frac{H}{2})$ which means shifting the mesh x by $-\frac{H}{2}$, the bottom mesh is the value $u_h(x-\frac{H}{2})$ which shifts the mesh x by $\frac{H}{2}$. Here, $H = 2h$.

When $H = mh$, denote

$$\tilde{V}_h^k = V_h^k(x+\frac{H}{2}) = V_h^k(x-\frac{H}{2}),$$

we have

$$\int_{I_{j-\frac{m}{2}}} \left(u_h(x+\frac{mh}{2})\right)_t \tilde{v}_h(x)dx - \int_{I_{j-\frac{m}{2}}} u_h(x+\frac{mh}{2}) \left(\tilde{v}_h(x)\right)_x dx + u_h(x^-_{j+\frac{1}{2}})\tilde{v}_h(x^-_{j-\frac{m-1}{2}}) - u_h(x^-_{j-\frac{1}{2}})\tilde{v}_h(x^+_{j-1-\frac{m-1}{2}}) = 0,$$

$$\int_{I_{j-\frac{m}{2}}} \left(u_h(x-\frac{mh}{2})\right)_t \tilde{v}_h(x)dx - \int_{I_{j-\frac{m}{2}}} u_h(x-\frac{mh}{2}) \left(\tilde{v}_h(x)\right)_x dx + u_h(x^-_{j-m+\frac{1}{2}})\tilde{v}_h(x^-_{j-\frac{m-1}{2}}) - u_h(x^-_{j-m-\frac{1}{2}})\tilde{v}_h(x^+_{j-1-\frac{m-1}{2}}) = 0,$$

where $\tilde{v}_h \in \tilde{V}_h^k$. Subtracting the above two formulas and dividing by $H = mh$, we obtain

$$\int_{I_{j-\frac{m}{2}}} \left(\partial_H u_h(x)\right)_t \tilde{v}_h(x)dx - \int_{I_{j-\frac{m}{2}}} \partial_H u_h(x) \left(\tilde{v}_h(x)\right)_x dx + \partial_H u_h(x^-_{j-\frac{m-1}{2}})\tilde{v}_h(x^-_{j-\frac{m-1}{2}}) - \partial_H u_h(x^-_{j-1-\frac{m-1}{2}})\tilde{v}_h(x^+_{j-1-\frac{m-1}{2}}) = 0,$$

where $\partial_H u_h(x_{j-\frac{m-1}{2}}) = \left(u_h(x^-_{j+\frac{1}{2}}) - u_h(x^-_{j-m+\frac{1}{2}})\right)/(mh)$. The rest of the analysis is same as the $\partial_h u_h$ case. We then have the following corollaries:

Corollary 4.1.4. *Under the same conditions as in Theorem 1.2.1. The divided differences of the DG approximation, $\partial_H^\alpha u_h$ in the L^2 norm have the error*

$$\|\partial_H^\alpha (u - u_h)\|_{0,\Omega} \leq Ch^{k+1},$$

and in the negative order norm:

$$\|\partial_H^\alpha (u - u_h)\|_{-(k+1),\Omega} \leq Ch^{2k+1},$$

where $H = mh$ and $\alpha = (\alpha_1, \ldots, \alpha_d)$ is an arbitrary multi-index.

Corollary 4.1.5. *Under the same conditions as in Theorem 1.3.4, the scaling $H = mh$, then*

$$\|u - K_H^{(2k+1,k+1)} \star u_h\|_{0,\Omega_0} \leq Ch^{2k+1}.$$

Remark 4.1.2. *Aside from $H = mh$ case, it is very difficult to analyze the rest of the cases. Numerical examples suggest that the accuracy order is drops from $2k+1$ for uniform meshes.*

Although it is very difficult to analyze a general scaling H directly, when H is very close to a known results such as h or mh we can take advantage of the known results to bound $\partial_H^\alpha (u - u_h)$.

For example, considering a scaling of $H = (1 - c \cdot h)h$ or $H = (1 - \mathcal{O}(h))h$. Denote the error $e = u - u_h$. For the first divided difference of the DG error e, we have:

$$\begin{aligned}
\partial_H e &= \frac{1}{H}\left(e(x + \frac{1}{2}h - \frac{c}{2}h^2) - e(x - \frac{1}{2}h + \frac{c}{2}h^2)\right) \\
&= \frac{1}{H}\left\{\left(e(x + \frac{1}{h}) - \frac{ch^2}{2}e'(x + \frac{1}{h}) + \frac{1}{2}e^{(2)}(\xi)\left(\frac{ch^2}{2}\right)^2\right)\right. \\
&\quad \left. - \left(e(x - \frac{1}{h}) + \frac{ch^2}{2}e'(x - \frac{1}{h}) + \frac{1}{2}e^{(2)}(\zeta)\left(\frac{ch^2}{2}\right)^2\right)\right\} \\
&= \frac{1}{(1 - c \cdot h)h}\left\{h\partial_h e - \frac{ch^2}{2}\left(e'(x + \frac{1}{h}) + e'(x - \frac{1}{h})\right) + \mathcal{O}(h^4)\right\} \\
&= \frac{1}{(1 - c \cdot h)}\left\{\partial_h e - \frac{ch}{2}\left(e'(x + \frac{1}{h}) + e'(x - \frac{1}{h})\right) + \mathcal{O}(h^3)\right\}
\end{aligned}$$

In the L^2 norm, then

$$\|\partial_H e\|_0 \leq C_0\|\partial_h e\|_0 + C_1 h\|e'\| + C_2 h^3\|e^{(2)}\|.$$

Since $\|e^{(i)}\|_0 \leq C\|e\|_i \leq Ch^{-i}\|e\|_0 \leq Ch^{k+1-i}$, we have

$$\|\partial_H e\|_0 \leq C_0\|\partial_h e\|_0 + C_3 h^{k+1}.$$

Now, we have shown that using a scaling $H = (1 - c \cdot h)h$ has no negative effect on the accuracy order of the divided differences of the DG error. In fact, we can change the value of c to approach the optimal accuracy for the filtered solution in the L^2 norm (usually the optimal value of c is larger than zero and the respective scaling $H = (1 - c \cdot h)h < h$).

4.2 Divided Differences: Nonuniform Meshes

After investigation of the divided differences of DG solutions for uniform meshes, we move to nonuniform meshes. Since we do not know what is the suitable scaling for nonuniform meshes, we use the scaling H to represent a general scaling.

Similar to using a general scaling H for uniform meshes, the challenge is to for $u_h(x + \frac{H}{2})$ and $u_h(x - \frac{H}{2})$ to be in the a same space. Figure 4.4 indicates a general situation of the first divided difference. We can see that the situation is much worse in Figure 4.4 than in Figure 4.2. For a general nonuniform mesh, it is almost impossible to find a constant scaling H such that $u_h(x + \frac{H}{2})$ and $u_h(x - \frac{H}{2})$ belong to the same approximation space. In other words, for general nonuniform meshes, we can not provide a traditional analysis such as the error estimates for uniform meshes.

Therefore, we have to find a different way of finding a suitable scaling for nonuniform meshes. Generally speaking, there are two possible directions to consider:

- Using known results (results over uniform meshes or smoothly-varying meshes) to bound the errors over nonuniform meshes. This is the method we used to analyze the results over smoothness-varying meshes in Chapter 2.

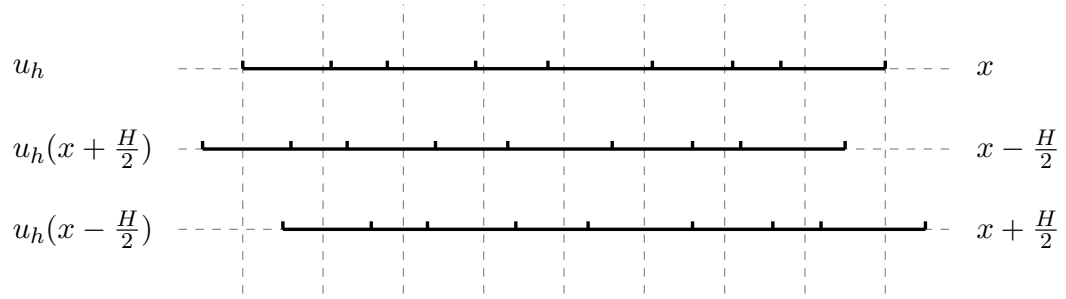


Figure 4.4: The top mesh is the original DG solution u_h over a nonuniform mesh x, the middle mesh is the value $u_h(x+\frac{H}{2})$ which means shifting the mesh x by $-\frac{H}{2}$, the bottom mesh is the value $u_h(x-\frac{H}{2})$ which shifts the mesh x by $\frac{H}{2}$.

- Considering a variable scaling $H(x)$ to force the divided difference components, $u_h(x+\frac{H}{2})$ and $u_h(x-\frac{H}{2})$, into the same space. Then, processing the same analysis as used in Section 4.1.

4.2.1 Variable Scaling $H(x)$

In this section, we first consider the idea of using a variable scaling $H(x), x \in \Omega$. For the first divided difference, we try to choose an $H(x)$ such that $u_h(x+\frac{H}{2})$ and $u_h(x-\frac{H}{2})$ are in the same space. For element $I_j = [x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}]$, we denote that

$$I_j \pm \frac{H}{2} = \left\{ x \pm \frac{H(x)}{2}, x \in I_j \right\},$$

of course, we require $x \pm \frac{H(x)}{2}$ are monotonically increasing with x. In order to get the same space we require that (the choice is the simplest one but not unique)

$$I_j + \frac{H}{2} = I_{j+1} - \frac{H}{2}, \quad j = 1, \ldots, N-1. \text{(the choice is not unique)}$$

Denote $H(x_{j-1/2}) = H_{j-1/2}$, this gives

$$x_{j-1/2} + \frac{H_{j-1/2}}{2} = x_{j+1/2} - \frac{H_{j+1/2}}{2} \quad j = 1, \ldots, N-1,$$

or,

$$\begin{aligned} \frac{1}{2}\left(H_{1/2} + H_{3/2}\right) &= \Delta x_1 \\ \frac{1}{2}\left(H_{3/2} + H_{5/2}\right) &= \Delta x_2 \\ &\cdots \\ \frac{1}{2}\left(H_{N-1/2} + H_{N+1/2}\right) &= \Delta x_N. \end{aligned}$$

Here we have N equations with $N+1$ unknown variables $H_{1/2}, \ldots, H_{N+1/2}$, hence we still need one more relation to solve this system. We can let $H_{1/2} = H_{N+1/2}$, but then the linear system may not have a solution. Another approach is to make an initial guess of $H_{1/2}$ ($H_{N+1/2}$) then solve the entire linear system. Once we obtain $H_{1/2}, \ldots, H_{N+1/2}$, the scaling $H(x)$ can be designed as

$$H(x) = \frac{1}{2}(H_{j-1/2} + H_{j+1/2}) - \frac{1}{2}(H_{j-1/2} - H_{j+1/2})\frac{x - x_j}{\Delta x_j}, \quad x \in I_j.$$

Finally, we obtain $x - \frac{H}{2}$ and $x - \frac{H}{2}$ in the same space, the remain proof will be the same as for uniform meshes.

To give a simple example of this idea, we consider a nonuniform mesh x with $N = 4$ elements, $\Delta x_1 = 2$, $\Delta x_2 = 1$, $\Delta x_3 = 1$ and $\Delta x_4 = 2$, with $x_{\frac{1}{2}} = 0$, $x_{\frac{3}{2}} = 2$, $x_{\frac{5}{2}} = 3$, $x_{\frac{7}{2}} = 4$ and $x_{\frac{9}{2}} = 6$. Here a solution $\{H_{j+\frac{1}{2}}\}_{j=0}^{4}$ satisfies the condition we mentioned, $I_j + \frac{H}{2} = I_{j+1} - \frac{H}{2}$, $H_{\frac{1}{2}} = 3.0$, $H_{\frac{3}{2}} = 1.0$, $H_{\frac{5}{2}} = 1.0$, $H_{\frac{7}{2}} = 1.0$ and $H_{\frac{9}{2}} = 3.0$. By designing a proper scaling $H(x)$, we can obtain a result such as in Figure 4.5 which has a similar structure to the uniform mesh.

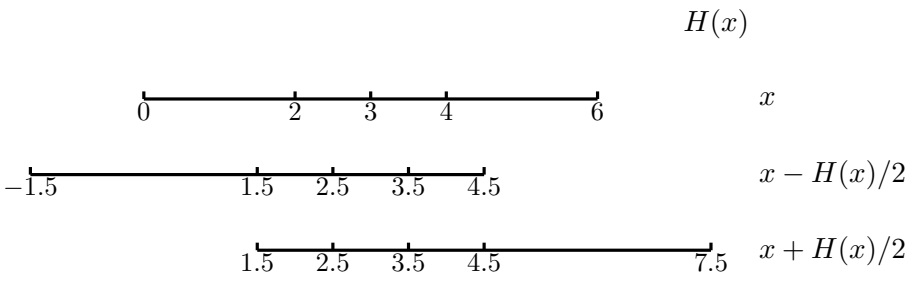


Figure 4.5: A four element nonuniform mesh with a proper choice variable scaling $H(x)$.

Here, we present an example of using the variable scaling for a smoothly-varying mesh used in Chapter 2.

Example 4.2.1. *As a simple example of the DG method and SIAC filter, consider the linear hyperbolic equation*

$$\begin{aligned} u_t + u_x &= 0, \quad (x,t) \in [0,1] \times (0,T] \\ u(x,0) &= \sin(2\pi x) \end{aligned}$$

with final time $T = 1$ over a smoothly-varying mesh: Mesh 2.4.1. The L^2 and L^∞ norm errors and respective accuracy order are given in Table 4.1, and Figure 4.6 shows the point-wise errors in log scale.

The preliminary results of using the variable scaling seems to work well for smoothly-varying meshes. However, we point out the following problems for general nonuniform meshes:

Table 4.1: L^2- and $L^\infty-$errors for the DG approximation u_h and the filtered solution $u_h^\star$.

	DG error				After filtering			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
				$\mathbb{P}^1$				
20	8.41E-03	–	2.25E-02	–	4.42E-03	–	6.77E-03	–
40	1.95E-03	2.11	6.38E-03	1.82	5.48E-04	3.01	8.17E-04	3.05
80	4.75E-04	2.03	1.68E-03	1.92	6.79E-05	3.01	1.01E-04	3.02
160	1.18E-04	2.01	4.30E-04	1.97	8.45E-06	3.01	1.25E-05	3.01
				$\mathbb{P}^2$				
20	3.07E-04	–	1.54E-03	–	2.51E-05	–	4.47E-05	–
40	3.85E-05	2.99	1.98E-04	2.96	5.36E-07	5.55	8.97E-07	5.64
80	4.82E-06	3.00	2.49E-05	2.99	1.28E-08	5.38	2.07E-08	5.43
160	6.03E-07	3.00	3.12E-06	3.00	3.49E-10	5.20	5.56E-10	5.22
				$\mathbb{P}^3$				
20	7.45E-06	–	2.63E-05	–	1.21E-06	–	2.35E-06	–
40	4.70E-07	3.99	1.65E-06	4.00	5.03E-09	7.91	9.28E-09	7.99
80	2.94E-08	4.00	1.04E-07	3.99	2.02E-11	7.96	3.82E-11	7.92
160	1.84E-09	4.00	6.49E-09	4.00	9.06E-14	7.80	1.74E-13	7.78

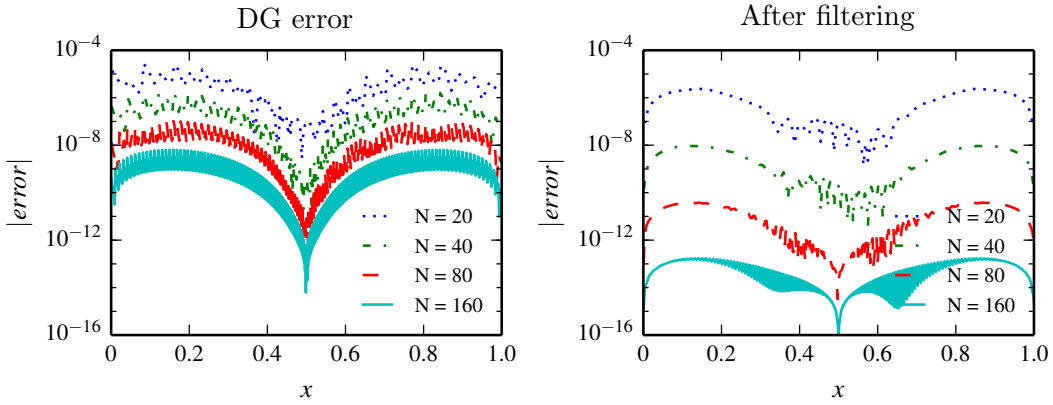


Figure 4.6: Comparison of the point-wise errors in log scale of the DG approximation together the filtered solution with polynomial $\mathbb{P}^3$.

- We can not guarantee that $I_1 - H/2$ and $I_N + H/2$ are in the same space, or there may not exist a proper scaling H.
- $H(x)$ may be negative for some region of x, then the filter is undefined.
- It is only for the first divided difference, the higher order divided differences still are not be guaranteed.

Further, there is a fundamental problem of using a variable scaling $H(x)$. Indeed, a variable scaling $H(x)$ is flexible to address the divided difference issue. However, we have to remind the reader that the divided difference operator is from the derivative of the filter (more precisely, the central B-splines). The divided difference scaling is also the filter scaling. In fact, consider the first order derivative of a scaled central B-spline,

$$\frac{d}{dx}\psi_H = \frac{d}{dx}\left(\frac{1}{H}\psi\left(\frac{x}{H}\right)\right) = -\frac{H'}{H^2}\psi\left(\frac{x}{H}\right) + \frac{1}{H}\psi'\left(\frac{x}{H}\right)\left(\frac{1-xH'}{H^2}\right).$$

From the above formula, we can see that if the scaling H is a variable function with respect to x, then we no longer are able to convert the derivatives into divided differences. That means we have to develop a completely new theoretical foundation for using a variable scaling H. Hence, we leave this direction for future work.

Now, we draw the attention back to using a constant scaling H. We note that a constant scaling is very important to keep the smoothness property of the filtered solutions.

4.3 Optimal Accuracy of Filtered Solutions

4.3.1 Preliminary Results over Nonuniform Meshes

In Chapter 1, Theorem 1.2.1 shows the error estimates of the DG approximation and its divided differences over uniform meshes. Unfortunately, for nonuniform meshes, estimates (1.3) and (1.4) are only vaild for the DG approximation itself, that is

Lemma 4.3.1 (Cockburn et al. [25])**.** *Under the same conditions as in Theorem 1.2.1. The DG approximation over a nonuniform mesh satisfies*

$$\|u - u_h\|_{0,\Omega} \leq Ch^{k+1},$$

and in the negative order norm:

$$\|u - u_h\|_{-(k+1),\Omega} \leq Ch^{2k+1}.$$

As for the divided differences, $\partial_h^\alpha u_h$, over nonuniform meshes, instead of (1.4), we only have

Lemma 4.3.2. *Under the same conditions as in Lemma 4.3.1, let H be a general constant, for nonuniform meshes the divided differences of the DG approximation in the L^2 norm satisfies*

$$\|\partial_H^\alpha(u - u_h)\|_{0,\Omega} \leq C_\alpha h^{2k+1} H^{-|\alpha|},$$

and in the negative order norm:

$$\|\partial_H^\alpha(u - u_h)\|_{-(k+1),\Omega} \leq C_\alpha h^{2k+1} H^{-|\alpha|},$$

where $\alpha = (\alpha_1, \ldots, \alpha_d)$ is an arbitrary multi-index.

Proof. c.f. Lemma 3.2.1. □

Here, we claim that without further assumptions on the nonuniform meshes, the estimates in Lemma 4.3.2 are already optimal. To explain this statement, in Table 4.2, we provide the L^2 and L^∞ errors for the divided differences of the L^2 projection of a sine function for a nonuniform mesh. The results show that the divided differences $\partial_h^\alpha u_h$ have accuracy order of only $k+1-\alpha$ in the L^2 norm for a nonuniform mesh. Since the premise that the divided differences also have $k+1$ order accuracy in the L^2 norm has already failed, the accuracy order of $2k+1$ cannot be theoretical guaranteed in the negative order norm.

Table 4.2: The L^2 error for the DG approximation and its divided differences of equation (4.17) at the initial time with $u(x,0)=\sin(x)$, $u_h=\mathcal{P}\sin(x)$, over a nonuniform mesh.

	u_h		$\partial_h u_h$		$\partial_h^2 u_h$	
Mesh	L^2 error	order	L^2 error	order	L^2 error	order
$\mathbb{P}^2$						
20	8.43E-05	–	1.29E-03	–	3.63E-02	–
40	1.02E-05	3.05	3.61E-04	1.84	1.79E-02	1.02
60	2.92E-06	3.09	1.44E-04	2.27	1.08E-02	1.26
80	1.19E-06	3.13	8.46E-05	1.84	8.33E-03	0.89
$\mathbb{P}^3$						
20	1.78E-06	–	2.99E-05	–	7.01E-04	–
40	1.17E-07	3.93	4.39E-06	2.77	1.75E-04	2.01
60	2.03E-08	4.32	1.10E-06	3.42	6.86E-05	2.30
80	6.50E-09	3.96	4.57E-07	3.05	3.83E-05	2.03

Based on Lemma 4.3.1 and Lemma 4.3.2, for general nonuniform meshes, the only theoretical estimate is

Theorem 4.3.3. *Under the same conditions as in Lemma 4.3.1, denote*

$$\Omega_0 + 2supp(K_H^{(2k+1,k+1)}) \subset\subset \Omega_1 \subset\subset \Omega.$$

Then, for general nonuniform meshes, we have

$$\|u - K_H^{(2k+1,k+1)} \star u_h\|_{0,\Omega_0} \leq C h^{\mu(2k+1)},$$

where the scaling H is chosen as

$$H = h^\mu, \quad \mu = \frac{2k+1}{3k+2}.$$

Proof. c.f. Theorem 3.2.2 or [25]. □

In some respects, Theorem 1.3.4 gives a useful conclusion that allows us to enhance the accuracy order of the DG solution, especially the derivatives of the DG solution in Chapter 3. However, if we consider the error reduction of the DG solution, there is still room for improvement. We will discuss the details in the following sections. For convenience, in this paper we refer to μ as the scaling order and $\mu_0 = \frac{2k+1}{3k+2}$.

Remark 4.3.1. *A more precise conclusion of Theorem 4.3.3 should use scaling order* $\mu_0 = \frac{2k+1}{3k+3}$.

The conclusion of Theorem 4.3.3 can easily be extended to hyperbolic conservation laws by using the same method. However, the error estimates of the divided differences of the DG solutions for nonlinear equations become quite complicated. It follows that there is no complete theoretical estimate of the filtered solutions for nonlinear hyperbolic conservation laws even for uniform meshes.

Now, we again focus on Theorem 4.3.3 itself. As mentioned earlier, Theorem 4.3.3 is the only theoretical result for general nonuniform meshes. In Chapter 3, the results demonstrated that this theorem is quite useful for improving the derivatives of DG approximations. However, from the perspective of improving the DG approximation itself, Theorem 4.3.3 is impractical in practice. With respect to improving the accuracy order, only if $k \geq 2$ is the accuracy order higher than the original DG approximation:

$$\mu_0(2k+1) > k+1 \quad \Rightarrow \quad k \geq 2.$$

If at least one order higher accuracy order is desired, then $k \geq 5$:

$$\mu_0(2k+1) \geq k+2 \quad \Rightarrow \quad k \geq 5.$$

With respect to the computational efficiency, as given in Chapter 3, when h is small (fine mesh), the filter scaling $H = h^{\mu_0} \geq h^{2/3}$ dramatically increases the support size of the filter. It follows that the computational cost dramatically increases too. Because the improvement in the accuracy order is quite small compared to the dramatically increased computational cost, Theorem 4.3.3 has rarely been used in practical applications since it was introduced in [25].

More importantly, instead of increasing the accuracy order, practical applications are more concerned about reducing the error. Although Theorem 4.3.3 improves the accuracy order, many practical examples suggest that using a scaling order of μ_0 usually increases the errors. For example, for the numerical experiments given in this chapter (Section 4.5), the filtered solutions that use scaling order of μ_0 have a worse error in the L^2 norm compared to the original DG solutions.

4.3.2 The Optimal Accuracy

Due to the impracticality and dissatisfactory accuracy of Theorem 4.3.3, we have to reconsider the filter scaling for nonuniform meshes. To complete this task, we first explore the relation between the filter scaling and the error of the filtered solution. We remind the readers that in this chapter, H represents the filter scaling and h represents the mesh size. By using Lemma 1.2.2, we can write the error estimate of the filtered solution as

$$\begin{aligned} \|u - u_h^\star\|_{0,\Omega_0} &\leq \|u - K_H^{(2k+1,k+1)} \star u\|_{0,\Omega_0} + \|K_H^{(2k+1,k+1)} \star (u - u_h)\|_{0,\Omega_0} \\ &\leq \Theta_1 + \Theta_2, \end{aligned} \tag{4.4}$$

where

$$\Theta_1 = \|u - K_H^{(2k+1,k+1)} \star u\|_{0,\Omega_0} \leq C_1 H^{2k+2} |u|_{H^{2k+2}}, \quad \text{(Property 1.3.2)} \tag{4.5}$$

and

$$\Theta_2 = C_0 \sum_{|\alpha|\leq k+1} \|D^\alpha K_H^{(2k+1,k+1)} \star (u-u_h)\|_{-(k+1),\Omega_{1/2}}, \quad \text{(Lemma 1.2.2)}$$
$$\leq C_0 C_1 \sum_{|\alpha|\leq k+1} \|\partial_H^\alpha (u-u_h)\|_{-(k+1),\Omega_1}, \tag{4.6}$$

where $\Omega_0 + supp(K_H^{(2k+1,k+1)}) \subset \Omega_{1/2}$ and $\Omega_{1/2} + supp(K_H^{(2k+1,k+1)}) \subset \Omega_1$. According to the above estimates, the error is bounded by Θ_1 and Θ_2, where Θ_1 describes the error generated by reproducing polynomials and Θ_2 represents the error in the negative order norm.

The estimate for Θ_1 is clear. The error is given by the polynomial reproduction property (1.9) and the exact solution u. It is obvious from (4.5) that Θ_1, $C_1 H^{2k+2}|u|_{H^{2k+2}}$, is increasing with the scaling H and is mainly determined by the scaling H and the exact solution u.

The Θ_2 term is a challenge. Lemma 4.3.2 gives an estimate of $\|\partial_H^\alpha (u-u_h)\|_{-(k+1),\Omega_1}$ for nonuniform meshes,

$$\|\partial_H^\alpha (u-u_h)\|_{-(k+1),\Omega_1} \leq C h^{2k+1} H^{-|\alpha|}.$$

The above estimate can be used for any nonuniform mesh, but it is not accurate for many nonuniform meshes, such as for the smoothly-varying meshes. It is easy to see that the Θ_2 term is strongly dependent on the unstructuredness of the mesh. However, based on [25], there is a trend that Θ_2 decreases with the scaling H. One can refer to Figure 4.7 for numerical support.

In this chapter, the purpose is to obtain the optimal accuracy of the filtered solution (minimize the error of the filtered solution). To do this, we need to find a proper scaling order μ (the scaling $H = h^\mu$) such that $\Theta_1 = \Theta_2$. As mentioned in [25], in the worst case the scaling order $\mu = \mu_0 = \frac{2k+1}{3k+2} \geq 0.6$, and in the best case $\mu \approx 1$. We examine the L^2 and L^∞ errors with scaling order μ in the range of $[0.6, 1]$ over different nonuniform meshes: Mesh 3.3.1 and Mesh 3.3.2. Figure 4.7 shows the variations. We can see that the optimal accuracy in the L^2 and L^∞ norms correspond to different scaling orders μ, see also Table 4.3. Since the theoretical estimates are based on the L^2 norm, in the following we focus only on the optimal accuracy in the L^2 norm. For convenience, we denote the value of μ that minimizes the error in the L^2 norm of the filtered solutions to be $\mu^\star$ and refer to it as the optimal scaling order.

The Convergence Rate

As shown in Figure 4.7, we notice that once $\mu < \mu^\star$, the errors of filtered solutions are dominated by the Θ_1 term, which has the convergence rate of $\mu(2k+2)$ (straight line with μ in the plots). Tables 4.4 and 4.5 show the results of using μ such that $\mu_0 < \mu < \mu^\star$. The filtered solutions have a higher accuracy order, and the errors are reduced compared to the original DG solutions. We also compare the results to the filtered solutions that use a scaling order μ_0 to demonstrate the improvement of using scaling order $\mu > \mu_0$. However, limited by the desire to obtain the same

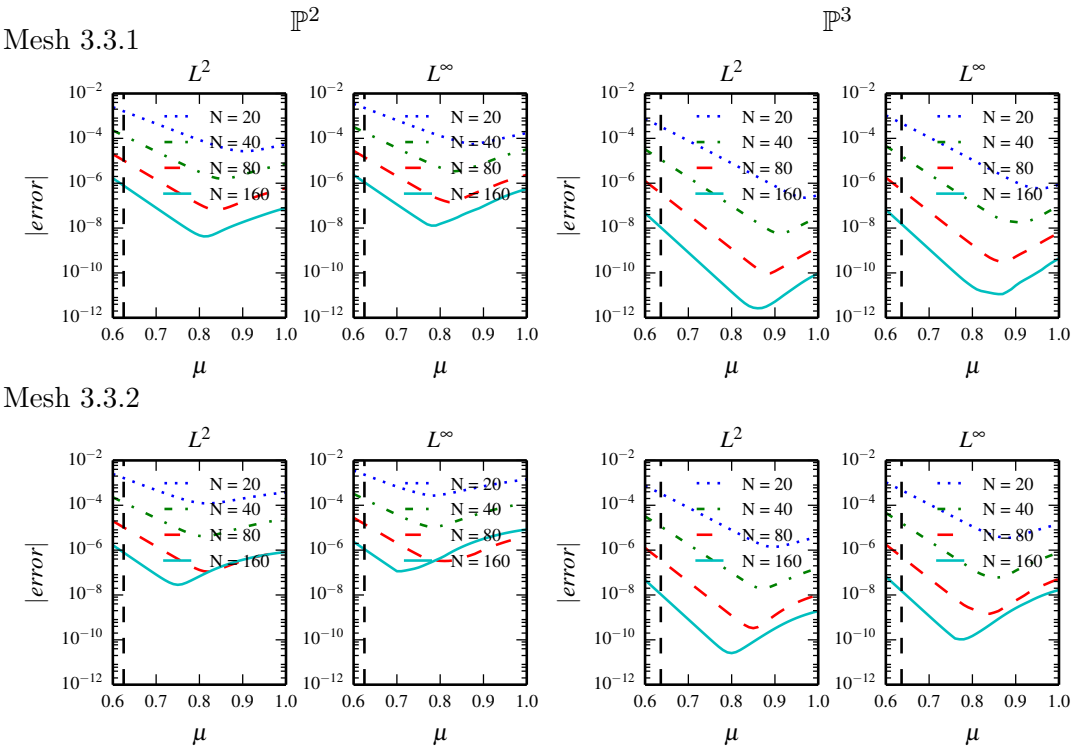


Figure 4.7: The L^2 and L^∞ errors in log scale of the filtered solutions with various scaling $H = h^\mu$, $\mu \in [0.6, 1.0]$. The black dashed line marks the location of $\mu_0 = \frac{2k+1}{3k+2}$. The DG approximation is for the linear equation (4.17) with polynomials of degree $k = 2, 3$ over Mesh 3.3.1 and Mesh 3.3.2.

accuracy order for all the filtered solutions, the results are still far from optimal. In fact, for nonuniform meshes, it does not make much sense to compare the results of the two meshes with different numbers of elements. For nonuniform meshes, we mainly concentrate on the given nonuniform mesh only, in other words, we want to find the optimal accuracy of the filtered solutions over the given nonuniform mesh.

Effects of the Number of B-splines

Aside from the scaling, the number of B-splines also has a significant effect on the filtered solution. Usually, the filter is constructed using $2k+1$ B-splines. $2k+1$ is the minimum requirement to obtain $2k+1$ accuracy order for uniform meshes. In Chapter 3, we reported that the accuracy order in Theorem 4.3.3 can be further improved by increasing the number of B-splines. Instead of considering the accuracy order, here we investigate the effect of the number of B-splines on the optimal accuracy. In order for Property 1.3.2 to remain valid, we consider using only $2(k+\beta)+1$ B-splines. The relation between β and the optimal L^2 accuracy, and the respective optimal scaling order are given in Table 4.6. The filtered solutions have a smaller error in the L^2 norm compared to the DG solutions. We see that by increasing the number of B-splines, both the optimal accuracy and computational cost (support size) are increased, which

Table 4.3: The optimal scaling order $\mu^\star$ with respect to Mesh 3.3.1 and Mesh 3.3.2 with $N = 20, 40, 80, 160$.

Mesh	Mesh 3.3.1					Mesh 3.3.2				
	u_h		$u_h^\star$			u_h		$u_h^\star$		
N	L^2 error	order	$\mu^\star$	L^2 error	order	L^2 error	order	$\mu^\star$	L^2 error	order
$\mathbb{P}^2$										
20	2.62E-04	–	0.90	2.69E-05	–	8.01E-04	–	0.82	1.21E-04	–
40	3.26E-05	3.00	0.85	1.58E-06	4.08	6.30E-05	3.67	0.81	4.16E-06	4.87
80	3.23E-06	3.34	0.84	6.50E-08	4.61	3.86E-06	4.03	0.82	1.10E-07	5.24
160	4.03E-07	3.00	0.81	4.25E-09	3.94	1.43E-06	1.44	0.75	2.84E-08	1.96
$\mathbb{P}^3$										
20	7.31E-06	–	0.97	2.25E-07	–	2.07E-05	–	0.90	1.39E-06	–
40	5.23E-07	3.80	0.91	5.69E-09	5.31	9.49E-07	4.45	0.87	1.95E-08	6.16
80	2.64E-08	4.31	0.88	9.46E-11	5.91	7.12E-08	3.74	0.85	3.31E-10	5.88
160	1.58E-09	4.07	0.86	2.65E-12	5.16	5.77E-09	3.63	0.80	2.56E-11	3.69

Table 4.4: L^2- and $L^\infty-$errors for the DG approximation u_h together with two filtered solutions (using a scaling of order $\mu = \mu_0$ and $\mu = 0.75$) for the linear equation (4.17) over Mesh 3.3.1.

	u_h				$\mu = \mu_0$				$\mu = 0.75$			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\mathbb{P}^1$												
20	7.59E-03	–	3.00E-02	–	2.91E-02	–	4.12E-02	–	4.39E-03	–	7.68E-03	–
40	1.87E-03	2.02	9.51E-03	1.66	7.47E-03	1.96	1.06E-02	1.96	6.03E-04	2.86	1.39E-03	2.47
80	4.17E-04	2.16	2.23E-03	2.10	1.88E-03	1.99	2.66E-03	1.99	6.97E-05	3.11	1.94E-04	2.84
160	1.00E-04	2.06	5.95E-04	1.90	4.74E-04	1.99	6.71E-04	1.99	9.35E-06	2.90	3.23E-05	2.59
$\mathbb{P}^2$												
20	2.62E-04	–	1.64E-03	–	5.13E-03	–	7.25E-03	–	6.12E-05	–	9.86E-05	–
40	3.26E-05	3.00	2.36E-04	2.80	5.86E-04	3.13	8.29E-04	3.13	2.75E-06	4.48	4.40E-06	4.49
80	3.23E-06	3.34	2.11E-05	3.49	6.21E-05	3.24	8.79E-05	3.24	1.19E-07	4.53	1.85E-07	4.57
160	4.03E-07	3.00	4.01E-06	2.39	6.36E-06	3.29	8.99E-06	3.29	5.48E-09	4.44	1.33E-08	3.80
$\mathbb{P}^3$												
20	7.31E-06	–	4.16E-05	–	1.08E-03	–	1.52E-03	–	3.82E-06	–	5.45E-06	–
40	5.23E-07	3.80	3.23E-06	3.68	5.17E-05	4.38	7.31E-05	4.38	6.26E-08	5.93	9.09E-08	5.91
80	2.64E-08	4.31	1.60E-07	4.33	2.22E-06	4.54	3.14E-06	4.54	9.94E-10	5.98	1.49E-09	5.93
160	1.58E-09	4.07	1.16E-08	3.79	9.10E-08	4.61	1.29E-07	4.61	1.57E-11	5.99	2.53E-11	5.88

implies that we can either increase the computational cost to obtain better accuracy or sacrifice accuracy to reduce the computational cost. In this chapter, for consistency we use only the filter $K^{(2k+1,k+1)}$.

Remark 4.3.2 (Effects of the order of B-splines)**.** *Unlike using a different number of B-splines, using a different order of B-splines has only a small effect on the optimal accuracy. Our study shows that there is a negative impact on the optimal accuracy by using B-splines of an order less than $k+1$, and using B-splines of an order higher than $k+1$ does not provide any added benefits. The details are neglected in this chapter.*

4.4 The Unstructuredness of Nonuniform Meshes

In the previous section, we demonstrated that there exists an optimal scaling order $\mu^\star$ such that using a scaling of $H = h^{\mu^\star}$ minimizes the error of the filtered solutions in

Table 4.5: L^2- and $L^\infty-$errors for the DG approximation u_h together with two filtered solutions (using a scaling order of $\mu = \mu_0$ and $\mu = 0.7$) for the linear equation (4.17) over Mesh 3.3.2.

	u_h				$\mu = \mu_0$				$\mu = 0.7$			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\mathbb{P}^1$												
20	1.00E-02	–	3.12E-02	–	3.16E-02	–	4.46E-02	–	7.81E-03	–	1.17E-02	–
40	1.99E-03	2.34	1.03E-02	1.60	7.60E-03	2.06	1.07E-02	2.05	8.42E-04	3.21	1.50E-03	2.96
80	6.38E-04	1.64	3.99E-03	1.37	1.90E-03	2.00	2.70E-03	1.99	1.10E-04	2.94	2.88E-04	2.38
160	1.43E-04	2.15	1.06E-03	1.92	4.79E-04	1.99	6.80E-04	1.99	1.97E-05	2.48	5.86E-05	2.30
$\mathbb{P}^2$												
20	8.01E-04	–	5.52E-03	–	5.15E-03	–	7.28E-03	–	1.64E-04	–	2.63E-04	–
40	6.30E-05	3.67	5.42E-04	3.35	5.87E-04	3.13	8.30E-04	3.13	7.96E-06	4.37	1.28E-05	4.37
80	3.86E-06	4.03	2.67E-05	4.35	6.22E-05	3.24	8.79E-05	3.24	4.21E-07	4.24	6.20E-07	4.36
160	1.43E-06	1.44	2.23E-05	0.26	6.36E-06	3.29	8.99E-06	3.29	3.05E-08	3.79	1.53E-07	2.02
$\mathbb{P}^3$												
20	2.07E-05	–	1.17E-04	–	1.08E-03	–	1.52E-03	–	1.24E-05	–	1.79E-05	–
40	9.49E-07	4.45	7.44E-06	3.97	5.17E-05	4.38	7.31E-05	4.38	2.71E-07	5.52	3.84E-07	5.54
80	7.12E-08	3.74	5.57E-07	3.74	2.22E-06	4.54	3.14E-06	4.54	5.71E-09	5.57	8.47E-09	5.50
160	5.77E-09	3.63	6.75E-08	3.04	9.10E-08	4.61	1.29E-07	4.61	1.19E-10	5.58	1.78E-10	5.57

the L^2 norm. Then, the remaining question is how to find $\mu^\star$ for a given nonuniform mesh. Table 4.3 provides $\mu^\star$ by testing different values of the scaling, which is certainly impractical in practice. Theoretically, even for uniform meshes whose optimal scaling order is $\mu^\star \approx 1$, it is impossible to find the exact value of $\mu^\star$. However, in this section, we propose an approximation μ_h that is sufficiently close to $\mu^\star$ and leads to filtered solutions with improved quality.

Figure 4.7 suggests that for different structures of nonuniform meshes, the optimal scaling order is different. The rule of thumb is that the more unstructured the mesh, the smaller the value of $\mu^\star$. In order to approximate the value of $\mu^\star$, it is important to define a measure of the unstructuredness of nonuniform meshes.

4.4.1 The Measure of Unstructuredness

Before discussing the unstructuredness, we first provide a definition of structured meshes.

Definition 4.4.1 (Structured Mesh)**.** *A mesh with N elements is considered* **structured** *if there exists a function $f \in \mathcal{C}^\infty$ and $f' > 0$, such that*

$$x_{j+\frac{1}{2}} = f(\xi_{j+\frac{1}{2}}), \quad \forall j = 0, \dots, N, \tag{4.7}$$

where $\left\{\xi_{j+\frac{1}{2}}\right\}_{j=0}^{N}$ corresponds to a uniform mesh with N elements over the same domain.

According to Chapter 2, filtered solutions for structured meshes have the same accuracy order ($2k+1$ for linear hyperbolic equations) as for uniform meshes.

Now we introduce a new parameter σ, the unstructuredness of the nonuniform mesh, to measure the difference between the given nonuniform mesh and a structured mesh with the same number of elements.

Table 4.6: The optimal L^2 accuracy and the respective $\mu^\star$ (scaling $H = h^{\mu^\star}$) with filters constructed by $2(k+\beta)+1$ B-splines for the linear equation (4.17) over Mesh 3.3.1 and Mesh 3.3.2.

	$\beta=-2$		$\beta=-1$		$\beta=0$		$\beta=1$		$\beta=2$	
Mesh	L^2 error	$\mu^\star$	L^2 error	$\mu^\star$	L^2 error	$\mu^\star$	L^2 error	$\mu^\star$	L^2 error	$\mu^\star$
Mesh 3.3.1					$\mathbb{P}^2$					
20	–	–	9.78e-05	1.08	2.69e-05	0.90	1.25e-05	0.82	9.01e-06	0.77
40	–	–	9.85e-06	1.02	1.58e-06	0.85	5.24e-07	0.77	3.49e-07	0.71
80	–	–	7.16e-07	1.01	6.50e-08	0.84	1.80e-08	0.75	9.82e-09	0.66
160	–	–	7.16e-08	0.98	4.25e-09	0.81	8.73e-10	0.72	3.83e-10	0.63
					$\mathbb{P}^3$					
20	4.62e-04	1.39	1.16e-06	1.09	2.25e-07	0.97	6.39e-08	0.89	2.73e-08	0.84
40	3.33e-05	1.31	5.31e-08	1.02	5.69e-09	0.91	1.30e-09	0.84	7.86e-10	0.77
80	1.67e-07	1.27	1.52e-09	1.00	9.46e-11	0.88	2.36e-11	0.82	8.60e-12	0.72
160	1.01e-08	1.24	5.32e-11	0.97	2.65e-12	0.86	6.07e-13	0.75	1.48e-13	0.69
Mesh 3.3.2					$\mathbb{P}^2$					
20	–	–	3.50e-04	0.97	1.21e-04	0.82	5.99e-05	0.74	4.15e-05	0.70
40	–	–	2.16e-05	0.97	4.16e-06	0.81	1.34e-06	0.73	8.60e-07	0.68
80	–	–	9.76e-07	1.00	1.10e-07	0.82	2.65e-08	0.73	1.97e-08	0.65
160	–	–	4.01e-07	0.90	2.84e-08	0.75	8.33e-09	0.65	3.45e-09	0.60
					$\mathbb{P}^3$					
20	1.46e-05	1.30	4.93e-06	1.01	1.39e-06	0.90	4.99e-07	0.83	2.40e-07	0.78
40	6.61e-07	1.27	1.37e-07	0.98	1.95e-08	0.87	4.75e-09	0.80	2.45e-09	0.74
80	4.53e-08	1.22	5.17e-09	0.95	3.31e-10	0.85	8.20e-11	0.77	4.66e-11	0.69
160	4.20e-09	1.17	4.07e-10	0.90	2.56e-11	0.80	6.34e-12	0.71	1.73e-12	0.65

Definition 4.4.2 (Unstructuredness)**.** *For a nonuniform mesh* $\left\{x_{j+\frac{1}{2}}\right\}_{j=0}^{N}$*, its unstructuredness* σ *is given by*

$$\sigma = \inf_{f\in\mathcal{C}^\infty, f'>0} \left(\sum_{j=0}^{N} \left(f(\xi_{j+\frac{1}{2}}) - x_{j+\frac{1}{2}} \right)^2 / (N+1) \right)^{\frac{1}{2}}, \tag{4.8}$$

where $\left\{\xi_{j+\frac{1}{2}}\right\}_{j=0}^{N}$ *corresponds to the uniform mesh with* N *elements over the same domain. The smaller the* σ*, the more structured the mesh.*

Without loss of generality, we denote the domain $\Omega = [0,1]$. Then in the worst case we have

$$\left(\sum_{j=0}^{N} \left(f(\xi_{j+\frac{1}{2}}) - x_{j+\frac{1}{2}} \right)^2 / (N+1) \right)^{\frac{1}{2}} < \left(\sum_{j=0}^{N} (1-0)^2/(N+1) \right)^{\frac{1}{2}} = 1 \Rightarrow \sigma < 1.$$

Remark 4.4.1. *The definition of unstructuredness is designed by considering the discrete* L^2 *norm formula. It is a natural choice since the focus is on the error in the* L^2 *norm. Also, there are different ways to identify the unstructuredness of the mesh, such as through the variation of the mesh elements.*

4.4.2 SIAC Filtering Based on the Unstructuredness Parameter

After defining the unstructuredness σ, we now study the relation of σ and the filter scaling. However, the challenge is that due to the definition, it is nearly impossible to estimate the negative order norm exactly, let alone the effect of the divided differences over nonuniform meshes. Even for the first divided difference, since $u_h(x+\frac{H}{2})$ and $u_h(x-\frac{H}{2})$ are not in the same space, the traditional error estimates are not rigorous. Let us first consider the first divided difference, $\partial_H u_h$. Theorem 4.3.3 is based on the inequality that

$$\begin{aligned}\|\partial_H(u-u_h)\|_0 &\leq \frac{1}{H}\left\{\left\|(u-u_h)(x+\frac{H}{2})\right\|_0+\left\|(u-u_h)(x-\frac{H}{2})\right\|_0\right\}\\ &\leq 2\|u-u_h\|_0 H^{-1}.\end{aligned}$$

The above estimate bounds the first divided difference by considering the two parts separately instead of treating the divided difference as one component. This is the reason Theorem 4.3.3 does not consider the structure of the nonuniform meshes.

In this chapter, we propose a method based on relating the nonuniform mesh to its closest structured mesh (under definition (4.8)). That is

$$\underbrace{\|\partial_H(u-u_h)\|_0}_{\text{nonuniform mesh}} \leq \underbrace{\|\partial_H(u-u_h)\|_{0,f(\xi)}}_{\text{structured mesh}} + \underbrace{\|\partial_H(u-u_h)\|_{0,\text{diff}}}_{\text{difference}}.$$

As mentioned earlier, we know that the first divided difference over the structured mesh $\left\{f(\xi_{j+\frac{1}{2}})\right\}_{j=0}^{N}$ has nice properties. Then, we assume that the error of the first divided difference of the DG solution for the nonuniform mesh $\left\{x_{j+\frac{1}{2}}\right\}_{j=0}^{N}$ is dominated by the difference between the nonuniform mesh and its closest structured mesh.

Now, consider the difference term $\|\partial_H(u-u_h)\|_{0,\text{diff}}$ and denote that $\Omega_j=[x_{j+\frac{1}{2}}, f(\xi_{j+\frac{1}{2}})]$ (or $\Omega_j=[f(\xi_{j+\frac{1}{2}}), x_{j+\frac{1}{2}}]$) for $j=0,\ldots,N$. We have

$$\|\partial_H(u-u_h)\|_{0,\text{diff}} = \frac{2}{H}\left(\sum_{j=0}^{N}\|u-u_h\|_{0,\Omega_j}^2/(N+1)\right)^{\frac{1}{2}}.$$

Since the approximation u_h on the interval Ω_j cannot be estimated rigorously through the traditional error estimates, we assume that

$$\begin{aligned}\|u-u_h\|_{0,\Omega_j}^2 &= \int_{\Omega_j}(u-u_h)^2dx \leq C\,|\Omega_j|\,h^{2k+2}\\ &= C\left|x_{j+\frac{1}{2}}-f(\xi_{j+\frac{1}{2}})\right|h^{2k+2}. \qquad (4.9)\end{aligned}$$

Then, we have

$$\begin{aligned}
\|\partial_H(u-u_h)\|_{0,\text{diff}} &= \frac{2}{H}\left(\sum_{j=0}^{N}\|u-u_h\|_{0,\Omega_j}^2/(N+1)\right)^{\frac{1}{2}} \\
&\le Ch^{k+1}H^{-1}\left(\sum_{j=0}^{N}\left|x_{j+\frac{1}{2}}-f(\xi_{j+\frac{1}{2}})\right|/(N+1)\right)^{\frac{1}{2}} \\
&\le Ch^{k+1}H^{-1}\left\{\left((N+1)\sum_{j=0}^{N}\left(x_{j+\frac{1}{2}}-f(\xi_{j+\frac{1}{2}})\right)^2\right)^{\frac{1}{2}}/(N+1)\right\}^{\frac{1}{2}} \\
&= Ch^{k+1}H^{-1}\left\{\left(\sum_{j=0}^{N}\left(f(\xi_{j+\frac{1}{2}})-x_{j+\frac{1}{2}}\right)^2/(N+1)\right)^{\frac{1}{2}}\right\}^{\frac{1}{2}}
\end{aligned}$$

By using definition (4.8) and the assumption that $\|\partial_H(u-u_h)\|_{0,\text{diff}}$ is the dominant term, we obtain

$$\|\partial_H(u-u_h)\|_0 \le C\frac{\sqrt{\sigma}}{H}h^{k+1} = C\frac{h^{\frac{1}{2}\log_h\sigma}}{H}h^{k+1}, \tag{4.10}$$

with induction

$$\|\partial_H^{\alpha}(u-u_h)\|_0 \le C\frac{\sqrt{\sigma}}{H}h^{k+1} = C\left(\frac{h^{\frac{1}{2}\log_h\sigma}}{H}\right)^{\alpha}h^{k+1}. \tag{4.11}$$

Remark 4.4.2. *The above estimates are the reason we use formula* (4.8) *to define the unstructuredness. Also, we point out that the assumption* (4.9) *is empirical rather than a rigorous theoretical estimate. Furthermore, the assumption that* $\|\partial_H(u-u_h)\|_{0,diff}$ *dominates* $\|\partial_H(u-u_h)\|_0$ *is true only when the nonuniform mesh is not so close to the respective structured mesh (*$\sigma \gg 0$*).*

Based on the value of σ, we divided the nonuniform meshes into two groups and discuss them separately.

• **Nearly structured meshes**: $\log_h\sigma \ge 2$.

This definition is based on estimate (4.11), when

$$\frac{\sqrt{\sigma}}{h} \ge \frac{\sqrt{\sigma}}{H} \ge 1, \quad \Rightarrow \quad \sigma \ge h^2 \quad \Rightarrow \quad \log_h\sigma \ge 2.$$

Then, the nonuniform mesh is almost a structured mesh, and the effect of the difference is negligible. In other words, we can treat these almost structured meshes as structured meshes and use the conclusions in Chapter 2. Also, we note that the definition is not strict; when $\log_h\sigma \approx 2$ we can also treat these nonuniform meshes as structured meshes.

• **Unstructured meshes**: $\log_h\sigma < 2$.

Under the same conditions as in Lemma 4.3.1, we assume that for a nonuniform mesh with the unstructuredness parameter σ as defined in equation (4.8), the divided differences of DG solutions satisfy

$$\|\partial_H^\alpha (u - u_h)\|_{-(k+1),\Omega_0} \leq C h^{2k+1} \left(\frac{h^{\frac{1}{2}\log_h \sigma}}{H} \right)^\alpha, \tag{4.12}$$

when $H \leq h^{\frac{1}{2}\log_h \sigma}$.

Remark 4.4.3. *Hypotheses* (4.12) *is based on estimate* (4.11) *and the results in [62], which is why it is an empirical rather than a rigorous theoretical analysis.*

Theorem 4.4.1. *Under the same conditions as in Theorem 4.3.3, and suppose that hypotheses* (4.12) *holds. Then, for a given general nonuniform mesh, we have*

$$\|u - K_H^{(2k+1,k+1)} \star u_h\|_{0,\Omega_0} \leq C h^{\mu(2k+2)},$$

where the filter scaling is $H = h^{\mu_h}$ *with* $\mu_h = \frac{2k+1}{3k+3} + \frac{1}{6}\log_h \sigma$.

Proof. With the assumption (4.12), the divided differences of the approximation satisfy

$$\sum_{\alpha=0}^{k+1} \|\partial_H^\alpha (u - u_h)\|_{-(k+1)} \leq C \left(\frac{h^{\frac{1}{2}\log_h \sigma}}{H} \right)^{k+1} h^{2k+1},$$

and according to equations (4.4) - (4.6), let

$$H^{2k+2} = \left(\frac{h^{\frac{1}{2}\log_h \sigma}}{H} \right)^{k+1} h^{2k+1},$$

Then we have

$$H = h^{\mu_h} \quad \mu_h = \frac{2k+1}{3(k+1)} + \frac{1}{6}\log_h \sigma \approx \frac{2}{3} + \frac{1}{6}\log_h \sigma > \frac{1}{2}\log_h \sigma. \tag{4.13}$$

Here, we note that $H = h^{\mu_h} \leq h^{\frac{1}{2}\log_h \sigma}$. □

Theorem 4.4.1 demonstrates the relation of the scaling order μ_h and the unstructuredness σ. By using μ_h the filtered solution can obtain a better accuracy order compared to Theorem 4.3.3 for the given nonuniform mesh. This improvement is dependent on σ.

Computing σ

Formula (4.13) gives a relation between the scaling order μ and unstructuredness σ, but we still need to calculate the value of σ. Since σ can not be easily calculated by formula 4.8, here we present an alternative approximation σ_h for σ, which is calculated by implementing the least squares algorithm:

$$\sigma_h = \min_{f \in \mathbb{P}^n} \left(\sum_{j=0}^{N} \left(f(\xi_{j+\frac{1}{2}}) - x_{j+\frac{1}{2}} \right)^2 / (N+1) \right)^{\frac{1}{2}}, \tag{4.14}$$

where

$$f(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0,$$

and

$$\begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = A^{-1} \begin{pmatrix} \sum_{j=0}^{N} x_{j+\frac{1}{2}} \left(\xi_{j+\frac{1}{2}}\right)^0 \\ \sum_{j=0}^{N} x_{j+\frac{1}{2}} \left(\xi_{j+\frac{1}{2}}\right)^1 \\ \vdots \\ \sum_{j=0}^{N} x_{j+\frac{1}{2}} \left(\xi_{j+\frac{1}{2}}\right)^n \end{pmatrix},$$

with

$$A = \begin{pmatrix} \sum_{j=0}^{N} \left(\xi_{j+\frac{1}{2}}\right)^0 & \sum_{j=0}^{N} \left(\xi_{j+\frac{1}{2}}\right)^1 & \cdots & \sum_{j=0}^{N} \left(\xi_{j+\frac{1}{2}}\right)^n \\ \sum_{j=0}^{N} \left(\xi_{j+\frac{1}{2}}\right)^1 & \sum_{j=0}^{N} \left(\xi_{j+\frac{1}{2}}\right)^2 & \cdots & \sum_{j=0}^{N} \left(\xi_{j+\frac{1}{2}}\right)^{n+1} \\ \vdots & \vdots & \cdots & \vdots \\ \sum_{j=0}^{N} \left(\xi_{j+\frac{1}{2}}\right)^n & \sum_{j=0}^{N} \left(\xi_{j+\frac{1}{2}}\right)^{n+1} & \cdots & \sum_{j=0}^{N} \left(\xi_{j+\frac{1}{2}}\right)^{2n} \end{pmatrix}.$$

Remark 4.4.4. *We note that the condition $f' > 0$ in formula 4.8 is usually satisfied by requiring $n \ll N$.*

In order to give an idea of the computational cost of this algorithm, we present the CPU time of calculating σ_h with different n and N in Table 4.7. The CPU time is given by the average of 10000 times computation, using quadruple precision. The computational environment is a Intel(R) Core(TM)2 Duo CPU E8500 @3.16GHZ, Intel Fortran compiler 9.1. Table 4.7 shows that the computational cost of calculating σ_h is

Table 4.7: The CPU time (seconds) of calculating σ_h with polynomials of degree $n = 2, 3, 4, 5, 6$ for $N = 20, 40, 80, 160$.

time	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$
$N=20$	5.08e-5	6.40e-5	7.88e-5	9.48e-5	1.13e-4
$N=40$	8.68e-5	1.12e-4	1.38e-4	1.66e-4	2.00e-4
$N=80$	1.58e-4	2.05e-4	2.57e-4	3.09e-4	3.71e-4
$N=160$	3.00e-4	3.91e-4	4.93e-4	5.90e-4	7.12e-4

negligible.

Table 4.3 gives the values of σ_h for different nonuniform meshes with different number of elements N. Also, we calculate the value of μ_h by formula (4.13) and compare it with the value of $\mu^\star$ given in Table 4.3. Since we obtain σ_h from equation (4.14), it is a little larger than σ in equation (4.8), and μ_h is little smaller than the optimal scaling order. The numerical experiments using μ_h will be presented in the next section.

Table 4.8: The value of $\log_h \sigma_h$ and μ_h (4.13) with three different nonuniform meshes of domain $\Omega = [0, 1]$: structured meshes (4.7) with $f(\xi) = \xi + 0.1 \cdot \sin(2\pi\xi)$, Mesh 3.3.1 and Mesh 3.3.2. Here we use a polynomial of degree $n = 8$ in the least squares algorithm (4.14).

	Structured	Mesh 3.3.1					Mesh 3.3.2				
N	$\log_h \sigma_h$	$\log_h \sigma_h$	μ_h:	$\mathbb{P}^2, \mathbb{P}^3$	$\mu^\star$:	$\mathbb{P}^2, \mathbb{P}^3$	$\log_h \sigma_h$	μ_h:	$\mathbb{P}^2, \mathbb{P}^3$	$\mu^\star$:	$\mathbb{P}^2, \mathbb{P}^3$
20	3.65	1.53	0.81	0.84	0.90	0.97	1.49	0.80	0.83	0.82	0.90
40	2.96	1.40	0.79	0.82	0.85	0.91	1.29	0.77	0.80	0.81	0.87
80	2.50	1.33	0.78	0.81	0.84	0.88	1.15	0.75	0.78	0.82	0.85
160	2.17	1.30	0.77	0.80	0.81	0.86	1.09	0.74	0.76	0.75	0.80

Correction of Formula (4.13)

In the previous analysis, we used formula (4.13) to calculate the scaling H. However, using formula (4.13) the scaling order μ_h is still smaller than the optimal scaling order $\mu^\star$ (see Table 4.8) because when we obtain (4.13), we have not consider the effects of the constants in (4.5) and (4.6) which do not depend on the nonuniform mesh. Ignoring the constants is not a problem if we consider only the accuracy order, but if we want to achieve the optimal accuracy and make a better approximation, we have to take these constants into account as well. Here, we propose an alternative that accounts for the effect of the exact solution:

$$H = m \cdot h^\mu \quad \mu = \frac{2k+1}{3(k+1)} + \frac{1}{6}\log_h \sigma \approx \frac{2}{3} + \frac{1}{6}\log_h \sigma, \tag{4.15}$$

where m depends on the exact solution u, more precisely, $m = C\left(\frac{|u|_{H^{2k+2}}}{|u|_{H^{k+1}}}\right)^{\frac{1}{3k+3}}$. Usually $|u|_{H^{2k+2}} < |u|_{H^{k+1}}$ so that the value of m is usually < 1. In other words, we can also write (4.15) as

$$H = h^\mu \quad \mu = \frac{2k+1}{3(k+1)} + \frac{1}{6}\log_h \sigma \approx \frac{2}{3} + \frac{1}{6}\log_h \sigma + \log_h m. \tag{4.16}$$

The optimal scaling order is decided by three parts: the basic order, $\mu_0 = \frac{2k+1}{3(k+1)}$, the effect of the mesh, $\frac{1}{6}\log_h \sigma$, and the effect of the exact solution. The first two parts are calculable, but the third is usually decided by experience in practice. For example, we know that the exact solution of Table 4.3 is $\sin(2\pi(x - T))$. We can consider the m to be $(\pi)^{-1/3}$ to obtain a more precise result. However, in order to avoid this manufactured aspect, we do not consider m in the following examples.

Remark 4.4.5. *It is possible that the optimal scaling $H < h$, if the mesh is nearly structured or a structured mesh. In fact, if we consider a wave function* $\sin(\lambda\pi)$ *for uniform meshes, the optimal scaling is much smaller than h when λ is large, see Chapter 6.*

4.4.3 A Note on Computation

Aside from error reduction, the computational cost of using the filter is also an important factor in practical applications. As mentioned in previous sections, the scaling

H used in Theorem 4.3.3 or Theorem 4.4.1 is usually larger than h over nonuniform meshes, which means that the computational cost is higher than the uniform mesh case in Chapter 2 and [25]. Based on Figure 4.7, when $\mu \in [\mu^\star, 1]$, the final accuracy is directly related to the scaling order μ, which means one can sacrifice accuracy to improve computational efficiency. For example, if the mesh is not so unstructured, a naive choice of scaling $H = \max_j \Delta x_j$ (or $H = 1.5 \max_j \Delta x_j$, $H = 2 \max_j \Delta x_j$) can lead to acceptable results as obtained in [30, 48].

4.5 Numerical Results

In the previous section, we proposed using the scaling order μ_h given by (4.13). Using the scaling order μ_h can improve the accuracy order from the original discontinuous Galerkin solutions. Also, since μ_h is designed to approximate the optimal scaling order $\mu^\star$, the filtered solutions are expected to have a reduction in error compared to DG solutions. For numerical verification, we apply the newly designed scaling order μ_h for various differential equations over nonuniform meshes - Mesh 3.3.1 and Mesh 3.3.2 - and compare it with using scaling order μ_0 mentioned in Theorem 4.3.3.

4.5.1 Linear Equation

Consider a linear equation

$$\begin{aligned} u_t + u_x &= 0, \quad (x,t) \in [0,1] \times (0,T], \\ u(x,0) &= \sin(2\pi x), \end{aligned} \tag{4.17}$$

at time $T = 1$ over Mesh 3.3.1 and Mesh 3.3.2. Table 4.9 includes the L^2 and L^∞ norm errors of the DG solutions and two filtered solutions with scaling order μ_0 and μ_h. First we check the results of using scaling order μ_0 in Theorem 4.3.3. Although the filtered solutions have better accuracy order, both the L^2 and L^∞ errors are worse than the original DG solution! Theorem 4.3.3 only says something about the order, but not about the quality of the errors. For using a scaling order μ_h, SIAC filtering is able to reduce the errors in the L^2 and L^∞ norm and improve the accuracy order. Especially when using a higher order polynomials or a sufficiently refined mesh the filtered errors are reduced compared to the DG errors. Figure 4.8, the point-wise error plots, demonstrate the other feature of SIAC filtering as its name implies: smoothness-increasing. Both the filtered solutions are $\mathcal{C}^{k-1}$ functions, the smoothness is significantly improved compared to the weakly continuous DG solutions. This continuity is the reason we only consider a constant scaling. In Figure 4.8 both filtered solutions reduce the oscillations in the DG solution and using a scaling order μ_0 completely removes the oscillations due to the large filter support size.

Comparing the results between Mesh 3.3.1 and Mesh 3.3.2, we can see that the DG solutions and filtered solutions with scaling order μ_h are better over Mesh 3.3.1 than over Mesh 3.3.2. It is because that Mesh 3.3.1 is more structured than Mesh 3.3.2. However, using scaling order μ_0 generates almost the same result, evidence that μ_0 does not take advantage of the mesh structures.

Table 4.9: L^2- and $L^\infty-$errors for the DG approximation u_h together two filtered solutions (using scaling order $\mu = \mu_0$ and $\mu = \mu_h$) for linear equation (4.17) over Mesh 3.3.1 and Mesh 3.3.2

	u_h				$\mu = \mu_0$				$\mu = \mu_h$			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
Mesh 3.3.1					$\mathbb{P}^1$							
20	7.59E-03	–	3.00E-02	–	2.91E-02	–	4.12E-02	–	4.95E-03	–	8.26E-03	–
40	1.87E-03	2.02	9.51E-03	1.66	7.47E-03	1.96	1.06E-02	1.96	7.19E-04	2.78	1.35E-03	2.61
80	4.17E-04	2.16	2.23E-03	2.10	1.88E-03	1.99	2.66E-03	1.99	9.10E-05	2.98	1.86E-04	2.87
160	1.00E-04	2.06	5.95E-04	1.90	4.74E-04	1.99	6.71E-04	1.99	1.23E-05	2.89	2.67E-05	2.80
					$\mathbb{P}^2$							
20	2.62E-04	–	1.64E-03	–	5.13E-03	–	7.25E-03	–	7.19E-05	–	1.11E-04	–
40	3.26E-05	3.00	2.36E-04	2.80	5.86E-04	3.13	8.29E-04	3.13	3.97E-06	4.18	6.03E-06	4.21
80	3.23E-06	3.34	2.11E-05	3.49	6.21E-05	3.24	8.79E-05	3.24	1.99E-07	4.32	2.90E-07	4.38
160	4.03E-07	3.00	4.01E-06	2.39	6.36E-06	3.29	8.99E-06	3.29	9.23E-09	4.43	1.40E-08	4.37
					$\mathbb{P}^3$							
20	7.31E-06	–	4.16E-05	–	1.08E-03	–	1.52E-03	–	3.17E-06	–	4.50E-06	–
40	5.23E-07	3.80	3.23E-06	3.68	5.17E-05	4.38	7.31E-05	4.38	6.03E-08	5.72	8.72E-08	5.69
80	2.64E-08	4.31	1.60E-07	4.33	2.22E-06	4.54	3.14E-06	4.54	9.97E-10	5.92	1.49E-09	5.87
160	1.58E-09	4.07	1.16E-08	3.79	9.10E-08	4.61	1.29E-07	4.61	1.42E-11	6.13	2.44E-11	5.93
Mesh 3.3.2					$\mathbb{P}^1$							
20	1.00E-02	–	3.12E-02	–	3.16E-02	–	4.46E-02	–	7.90E-03	–	1.19E-02	–
40	1.99E-03	2.34	1.03E-02	1.60	7.60E-03	2.06	1.07E-02	2.05	9.35E-04	3.08	1.58E-03	2.91
80	6.38E-04	1.64	3.99E-03	1.37	1.90E-03	2.00	2.70E-03	1.99	1.41E-04	2.73	2.87E-04	2.46
160	1.43E-04	2.15	1.06E-03	1.92	4.79E-04	1.99	6.80E-04	1.99	2.38E-05	2.56	5.00E-05	2.52
					$\mathbb{P}^2$							
20	8.01E-04	–	5.52E-03	–	5.15E-03	–	7.28E-03	–	1.25E-04	–	2.98E-04	–
40	6.30E-05	3.67	5.42E-04	3.35	5.87E-04	3.13	8.30E-04	3.13	6.27E-06	4.32	1.14E-05	4.70
80	3.86E-06	4.03	2.67E-05	4.35	6.22E-05	3.24	8.79E-05	3.24	4.35E-07	3.85	6.50E-07	4.14
160	1.43E-06	1.44	2.23E-05	0.26	6.36E-06	3.29	8.99E-06	3.29	3.18E-08	3.78	1.44E-07	2.17
					$\mathbb{P}^3$							
20	2.07E-05	–	1.17E-04	–	1.08E-03	–	1.52E-03	–	3.80E-06	–	5.99E-06	–
40	9.49E-07	4.45	7.44E-06	3.97	5.17E-05	4.38	7.31E-05	4.38	1.03E-07	5.20	1.47E-07	5.35
80	7.12E-08	3.74	5.57E-07	3.74	2.22E-06	4.54	3.14E-06	4.54	2.84E-09	5.18	4.22E-09	5.12
160	5.77E-09	3.63	6.75E-08	3.04	9.10E-08	4.61	1.29E-07	4.61	5.98E-11	5.57	1.07E-10	5.30

4.5.2 Variable Coefficient Equation

After the linear equation (4.17), which has a constant coefficient, we consider the variable coefficient equation

$$\begin{aligned} u_t + (au)_x &= f, \quad (x,t) \in [0,1] \times (0,T] \\ u(x,0) &= \sin(2\pi x), \end{aligned} \tag{4.18}$$

where the variable coefficient $a(x,t) = 2+\sin(2\pi(x+t))$ and the right side term $f(x,t)$ is chosen to make the exact solution be $u(x,t) = \sin(2\pi(x-t))$.

Similar to the linear equation example, we compare the L^2 and L^∞ norm errors in Table 4.10, and the point-wise error plots are given in Figure 4.9. The results are similar to the previous results for the constant coefficient equation. Here we only point out the features that are different to the linear equation. Using a scaling order μ_0 does not reliably reduce the errors in the L^2 norm and the L^∞ norm errors are still worse than the DG solutions. However, using a scaling order μ_h reduces the errors in the L^2 norm and the L^∞ norm. The point-wise error plots in Figure 4.9 are more oscillatory compared to Figure 4.8 due to the effects of the variable coefficient.

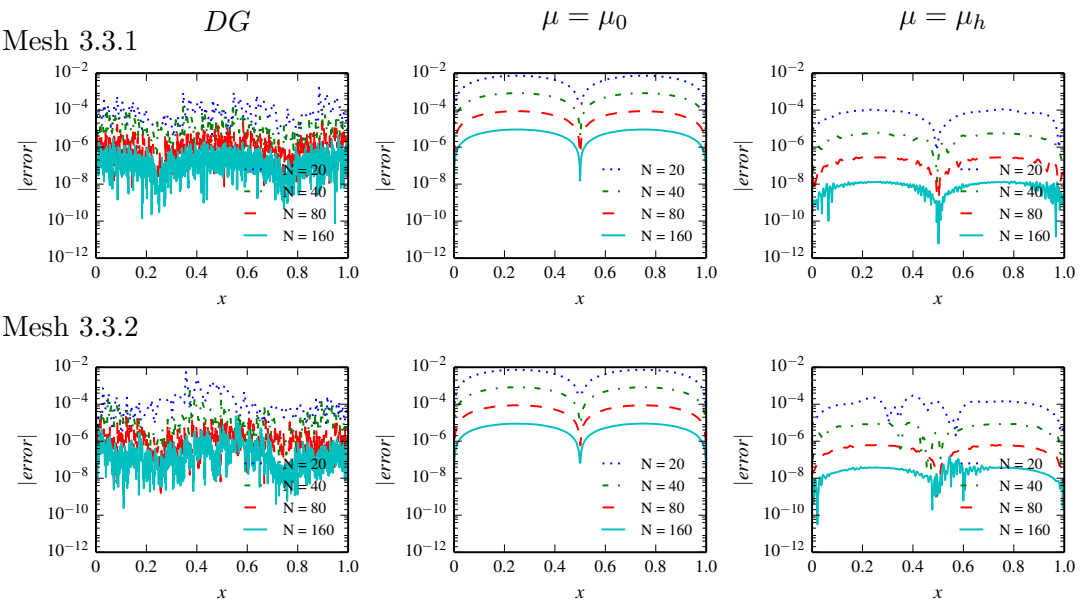


Figure 4.8: Comparison of the point-wise errors in log scale of the DG approximation together with two filtered solutions (using scaling order $\mu = \mu_0$ and $\mu = \mu_h$) for linear equation (4.17) over Mesh 3.3.1 and Mesh 3.3.2 with polynomial of degree $k = 2$.

4.5.3 Two-Dimensional Example

For the two-dimensional example, we consider a two-dimensional linear equation

$$\begin{aligned} u_t + u_x + u_y &= 0, \qquad (x, y) \in [0, 1] \times [0, 1], \\ u(x, y, 0) &= \sin(2\pi(x + y)), \end{aligned} \tag{4.19}$$

at time $T = 1$ over a two-dimensional quadrilateral extension of Mesh 3.3.1 and Mesh 3.3.2.

The L^2 and L^∞ norm errors are presented in Table 4.11 and Table 4.12, the point-wise error plots (pcolor plots) are included in Figure 4.10 and Figure 4.11. The results are very similar to the one-dimensional examples: the filtered solutions with scaling order μ_h reduce the errors in the L^2 norm; using a scaling order μ_0 increases the error in the L^2 norm over the DG error. In the two-dimensional case, the computational efficiency becomes more important compared to the one-dimensional case due the increased computational cost. As mentioned before, using a scaling order μ_0 is far more inefficient compared to using the scaling order μ_h. In particular, for a $\mathbb{P}^3$ polynomial basis with $N = 160 \times 160$ meshes, using a scaling order μ_0 is more than 8 times slower over Mesh 3.3.1 (5 times slower over Mesh 3.3.2) than using the scaling order μ_h.

Table 4.10: L^2- and $L^\infty-$errors for the DG approximation u_h together two filtered solutions (using scaling order $\mu = \mu_0$ and $\mu = \mu_h$) for variable coefficient equation (4.18) over Mesh 3.3.1 and Mesh 3.3.2.

	u_h				$\mu = \mu_0$				$\mu = \mu_h$			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
Mesh 3.3.1					$\mathbb{P}^1$							
20	6.93E-03	–	3.51E-02	–	2.50E-02	–	3.57E-02	–	1.61E-03	–	4.04E-03	–
40	1.83E-03	1.92	1.05E-02	1.74	6.83E-03	1.87	9.71E-03	1.88	2.32E-04	2.79	5.47E-04	2.89
80	4.15E-04	2.14	2.29E-03	2.20	1.82E-03	1.91	2.58E-03	1.91	3.72E-05	2.64	1.37E-04	2.00
160	1.00E-04	2.05	6.10E-04	1.91	4.66E-04	1.96	6.60E-04	1.97	6.00E-06	2.63	2.09E-05	2.71
					$\mathbb{P}^2$							
20	2.67E-04	–	1.71E-03	–	5.12E-03	–	7.25E-03	–	7.02E-05	–	1.32E-04	–
40	3.26E-05	3.03	2.25E-04	2.93	5.86E-04	3.13	8.29E-04	3.13	3.81E-06	4.20	6.82E-06	4.27
80	3.24E-06	3.33	2.11E-05	3.42	6.21E-05	3.24	8.79E-05	3.24	1.99E-07	4.26	3.23E-07	4.40
160	4.05E-07	3.00	4.01E-06	2.39	6.36E-06	3.29	8.99E-06	3.29	1.03E-08	4.27	2.78E-08	3.54
					$\mathbb{P}^3$							
20	7.43E-06	–	3.68E-05	–	1.08E-03	–	1.52E-03	–	3.18E-06	–	4.75E-06	–
40	5.25E-07	3.82	3.14E-06	3.55	5.17E-05	4.38	7.31E-05	4.38	6.07E-08	5.71	1.05E-07	5.50
80	2.65E-08	4.31	1.56E-07	4.33	2.22E-06	4.54	3.14E-06	4.54	1.01E-09	5.91	1.73E-09	5.93
160	1.58E-09	4.07	1.14E-08	3.78	9.10E-08	4.61	1.29E-07	4.61	1.53E-11	6.04	3.58E-11	5.59
Mesh 3.3.2					$\mathbb{P}^1$							
20	9.59E-03	–	4.42E-02	–	2.13E-02	–	3.00E-02	–	3.93E-03	–	7.08E-03	–
40	1.95E-03	2.30	1.14E-02	1.96	6.77E-03	1.65	9.62E-03	1.64	3.86E-04	3.35	1.09E-03	2.70
80	6.38E-04	1.61	4.19E-03	1.44	1.82E-03	1.90	2.60E-03	1.89	8.86E-05	2.12	2.85E-04	1.93
160	1.43E-04	2.15	1.09E-03	1.94	4.64E-04	1.97	6.60E-04	1.98	1.65E-05	2.42	5.72E-05	2.32
					$\mathbb{P}^2$							
20	7.90E-04	–	4.96E-03	–	5.08E-03	–	7.19E-03	–	1.71E-04	–	5.14E-04	–
40	6.33E-05	3.64	5.08E-04	3.29	5.86E-04	3.12	8.29E-04	3.12	8.54E-06	4.32	2.74E-05	4.23
80	3.88E-06	4.03	2.59E-05	4.29	6.21E-05	3.24	8.79E-05	3.24	4.40E-07	4.28	8.34E-07	5.04
160	1.44E-06	1.42	2.15E-05	0.27	6.36E-06	3.29	8.99E-06	3.29	1.28E-07	1.78	5.14E-07	0.70
					$\mathbb{P}^3$							
20	2.13E-05	–	1.12E-04	–	1.08E-03	–	1.52E-03	–	4.10E-06	–	8.22E-06	–
40	9.62E-07	4.47	6.98E-06	4.01	5.17E-05	4.38	7.31E-05	4.38	1.08E-07	5.24	2.02E-07	5.35
80	7.22E-08	3.74	5.24E-07	3.74	2.22E-06	4.54	3.14E-06	4.54	2.94E-09	5.20	5.31E-09	5.25
160	5.79E-09	3.64	6.05E-08	3.11	9.10E-08	4.61	1.29E-07	4.61	1.89E-10	3.96	9.78E-10	2.44

4.6 Conclusion

In this chapter, in order to apply a SIAC filter to DG solutions over nonuniform meshes, we have proposed implementing the filter with a scaling $H = h^{\mu_h}$. The scaling order μ_h is chosen according to the unstructuredness of the given nonuniform mesh. We have proved that by using the scaling $H = h^{\mu_h}$ with μ_h given in Theorem 4.4.1, the filtered solutions have an accuracy order of $\mu_h(2k+2)$, which is higher than the accuracy order of the DG solutions. In addition, since the scaling order μ_h is designed to approach the optimal scaling order $\mu^\star$, which minimizes the errors of the filtered solutions, the error reduction after filtering can be expected. The numerical results are promising: compared to the original DG errors, the filtered error (with scaling order μ_h) has significantly increased the accuracy as well as the accuracy order. Future work will concentrate on extending this scaling order μ_h to unstructured triangular meshes in two dimensions and tetrahedral meshes in three dimensions.

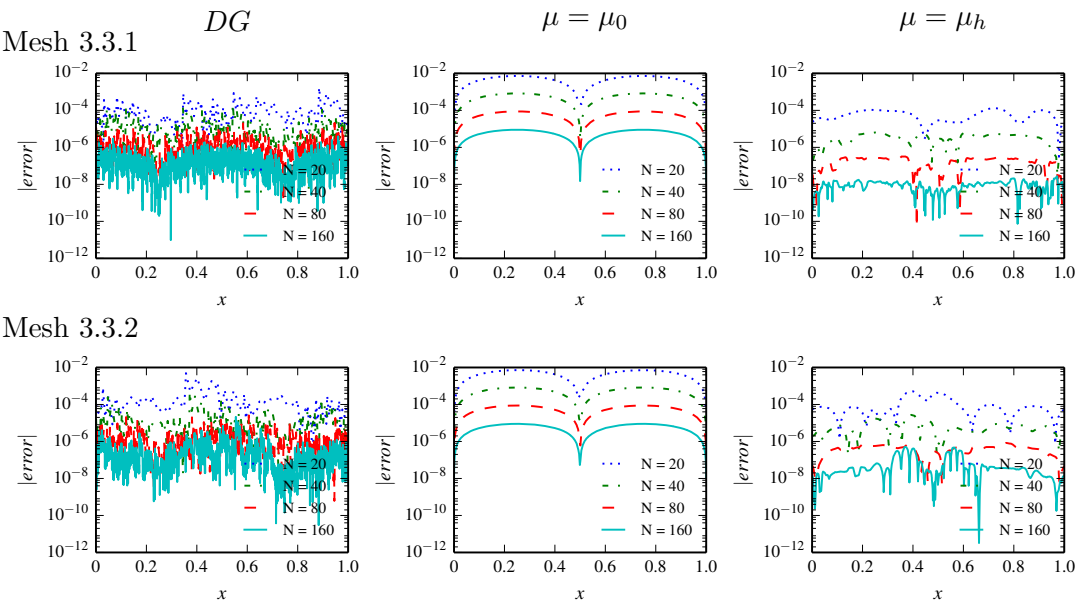


Figure 4.9: Comparison of the point-wise errors in log scale of the DG approximation together with two filtered solutions (using scaling order $\mu = \mu_0$ and $\mu = \mu_h$) for variable coefficient equation (4.17) over Mesh 3.3.1 and Mesh 3.3.2 with polynomial of degree $k = 2$

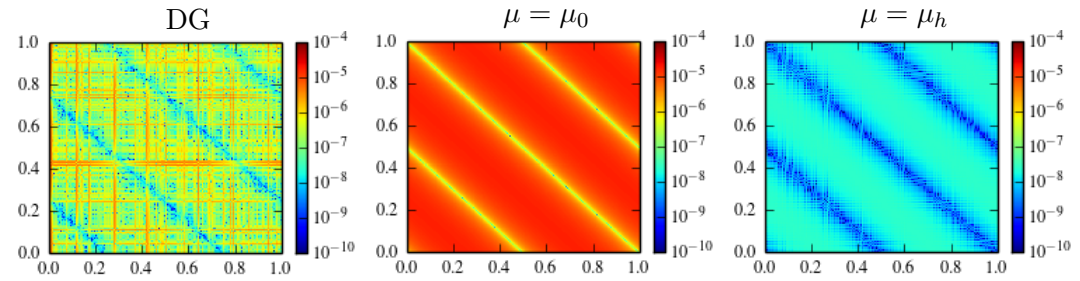


Figure 4.10: Comparison of the point-wise errors in log scale of the DG approximation together with two filtered solutions (using scaling order $\mu = \mu_0$ and $\mu = \mu_h$) for two-dimensional linear equation (4.19) over Mesh 3.3.1 (2D, $\mathbb{P}^2$ and $N = 160 \times 160$).

Table 4.11: L^2- and $L^\infty-$errors for the DG approximation u_h together two filtered solutions (using scaling order $\mu = \mu_0$ and $\mu = \mu_h$) for two-dimensional linear equation (4.19) over Mesh 3.3.1 (2D).

	DG				$\mu = \mu_0$				$\mu = \mu_h$			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\mathbb{P}^1$												
20×20	1.28E-02	–	6.09E-02	–	5.76E-02	–	8.20E-02	–	1.08E-02	–	1.86E-02	–
40×40	2.57E-03	2.31	1.86E-02	1.71	1.48E-02	1.96	2.11E-02	1.96	1.39E-03	2.96	2.55E-03	2.87
80×80	5.79E-04	2.15	4.94E-03	1.91	3.76E-03	1.98	5.33E-03	1.98	1.80E-04	2.94	3.62E-04	2.81
160×160	1.42E-04	2.03	1.26E-03	1.98	9.48E-04	1.99	1.34E-03	1.99	2.50E-05	2.85	5.27E-05	2.78
$\mathbb{P}^2$												
20×20	3.92E-04	–	3.19E-03	–	1.02E-02	–	1.45E-02	–	1.59E-04	–	2.37E-04	–
40×40	4.46E-05	3.13	4.85E-04	2.72	1.17E-03	3.12	1.66E-03	3.12	7.81E-06	4.34	1.19E-05	4.32
80×80	5.09E-06	3.13	5.29E-05	3.20	1.24E-04	3.24	1.76E-04	3.24	3.76E-07	4.38	5.69E-07	4.38
160×160	6.27E-07	3.02	7.49E-06	2.82	1.27E-05	3.29	1.80E-05	3.29	1.89E-08	4.31	3.22E-08	4.14
$\mathbb{P}^3$												
20×20	1.18E-05	–	8.74E-05	–	2.15E-03	–	3.04E-03	–	7.21E-06	–	1.03E-05	–
40×40	6.63E-07	4.16	6.65E-06	3.72	1.03E-04	4.38	1.46E-04	4.38	1.19E-07	5.92	1.74E-07	5.89
80×80	3.67E-08	4.17	4.03E-07	4.04	4.44E-06	4.54	6.28E-06	4.54	1.83E-09	6.02	2.82E-09	5.94
160×160	2.24E-09	4.04	2.53E-08	3.99	1.82E-07	4.61	2.57E-07	4.61	2.95E-11	5.96	5.08E-11	5.80

Table 4.12: L^2- and $L^\infty-$errors for the DG approximation u_h together two filtered solutions (using scaling order $\mu = \mu_0$ and $\mu = \mu_h$) for two-dimensional linear equation (4.19) over Mesh 3.3.2 (2D).

	DG				$\mu = \mu_0$				$\mu = \mu_h$			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\mathbb{P}^1$												
20×20	2.11E-02	–	1.72E-01	–	6.29E-02	–	9.05E-02	–	1.72E-02	–	3.69E-02	–
40×40	5.44E-03	1.96	6.98E-02	1.30	1.60E-02	1.97	2.32E-02	1.97	3.04E-03	2.50	9.63E-03	1.94
80×80	1.18E-03	2.21	1.42E-02	2.29	3.90E-03	2.04	5.59E-03	2.05	4.00E-04	2.93	1.17E-03	3.04
160×160	2.18E-04	2.43	2.90E-03	2.30	9.57E-04	2.03	1.36E-03	2.04	4.80E-05	3.06	1.11E-04	3.40
$\mathbb{P}^2$												
20×20	1.03E-03	–	7.74E-03	–	1.03E-02	–	1.45E-02	–	2.53E-04	–	5.66E-04	–
40×40	1.94E-04	2.41	2.17E-03	1.84	1.18E-03	3.13	1.67E-03	3.13	2.34E-05	3.43	8.41E-05	2.75
80×80	1.97E-05	3.30	3.55E-04	2.61	1.24E-04	3.24	1.76E-04	3.24	1.08E-06	4.44	5.13E-06	4.04
160×160	1.47E-06	3.74	2.77E-05	3.68	1.27E-05	3.29	1.80E-05	3.29	6.07E-08	4.15	1.26E-07	5.35
$\mathbb{P}^3$												
20×20	5.18E-05	–	6.23E-04	–	2.15E-03	–	3.04E-03	–	8.67E-06	–	1.63E-05	–
40×40	6.16E-06	3.07	9.20E-05	2.76	1.03E-04	4.38	1.46E-04	4.38	2.75E-07	4.98	1.01E-06	4.02
80×80	2.83E-07	4.44	3.84E-06	4.58	4.44E-06	4.54	6.28E-06	4.54	5.50E-09	5.64	1.57E-08	6.01
160×160	8.38E-09	5.08	1.40E-07	4.78	1.82E-07	4.61	2.57E-07	4.61	1.40E-10	5.30	2.85E-10	5.78

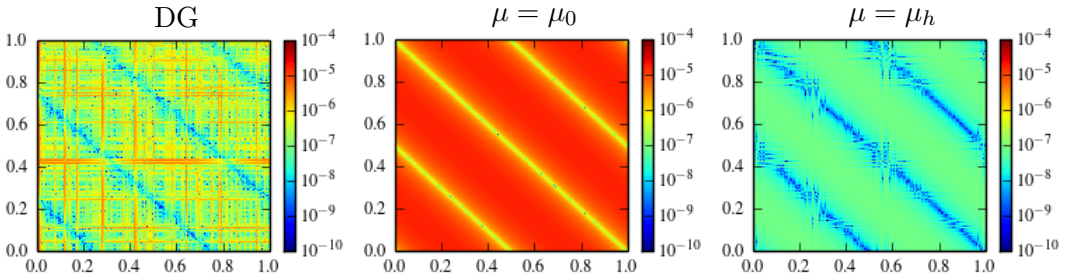


Figure 4.11: Comparison of the point-wise errors in log scale of the DG approximation together with two filtered solutions (using scaling order $\mu = \mu_0$ and $\mu = \mu_h$) for two-dimensional linear equation (4.19) over Mesh 3.3.2 (2D, $\mathbb{P}^2$ and $N = 160 \times 160$).

5 Applications of SIAC Filters in the Visualization

In the previous Chapters, we introduced different SIAC filters: symmetric filters, one-sided filters, derivative filters, etc. We demonstrated that by using SIAC filters, one can improve the results of DG approximation over uniform and nonuniform meshes with respect to both accuracy and smoothness. In this chapter, we apply SIAC filters in the visualization area and illustrate that the SIAC filter has great potential in these practical applications.

5.1 Introduction

Visualization is concerned with techniques that extract information from the results of simulations and computations, such as computational fluid dynamics simulations. The visualization technique is giving a picture to an approximation that helps one to understand the vector fields resulting from numerical simulations. Given a vector field, the streamlines of the field are curves that are tangential to the vector field at each point. Streamlines have been demonstrated in much of the literature, such as [12], to be a powerful and popular visualization method. However, application of streamline-based visualization to discontinuous field data represents a significant challenge due to the discontinuities in the fields. By applying SIAC filters to the discontinuous data prior to streamline integration, one can overcome the difficulties of the discontinuous nature of the data. This chapter is aimed at developing good performance algorithms of streamline integration by using different SIAC filters that were proposed in the previous chapters.

5.1.1 Streamline Integration

For a stationary vector field $\mathbf{u}$, a streamline is an integral curve that is given by the ordinary differential equation

$$\frac{d\mathbf{r}}{dt} = \mathbf{u}(\mathbf{r}), \quad \mathbf{r}(t_0) = \mathbf{x}_0. \tag{5.1}$$

Hence, streamline integration is often accomplished through the application of an ordinary differential equation (ODE) integrator such as Runge-Kutta (RK) schemes or

backward differentiation (BDF) methods. Since the theoretical foundation of these ODE schemes relies on a Taylor expansion, the errors of streamline integration are often related to the smoothness of the field through which the streamline is being integrated.

Discontinuous data fields generated by the finite volume and discontinuous Galerkin methods are very popular for numerical simulation, such as fluid dynamics simulation. However, calculating streamlines in such discontinuous data field presents a significant challenge to the classical streamline integration based on the Taylor expansion. Lacking smoothness at the inter-element level of DG data limits the accuracy of the streamline [61]. In the following sections, we will provide two different ways to use SIAC filters to enhance the performance of streamline integration over discontinuous data fields.

In this paper, we focus on two-dimensional vector fields with notation

$$\mathbf{u} = [u(x,y), v(x,y)]^T, \quad (x,y) \in \Omega$$

and the streamlines

$$\mathbf{r}(t) = [x(t), y(t)]^T.$$

5.2 Filtering the Entire Domain

The first method is straightforward and introduced in [61]. First we apply the SIAC filter over the entire domain Ω and obtain a smooth vector field $\mathbf{u}^\star = [u^\star(x,y), v^\star(x,y)]^T$, then uses ODE integrators, such as Runge-Kutta schemes, to calculate the streamline.

More precisely, consider a vector field given by the DG approximation,

$$\mathbf{u} = [u(x,y), v(x,y)]^T.$$

Then the filtered vector field is given by

$$\mathbf{u}^\star = [u^\star(x,y), v^\star(x,y)]^T,$$

with

$$u^\star(x,y) = \left(u \star K_h^{(2k+1,k+1)}\right)(x,y),$$
$$v^\star(x,y) = \left(v \star K_h^{(2k+1,k+1)}\right)(x,y).$$

Here, the two-dimensional SIAC filter is given by a tensor product of two one-dimensional filters that

$$K_h^{(2k+1,k+1)}(x,y) = K_{h_x}^{(2k+1,k+1)}(x) \cdot K_{h_y}^{(2k+1,k+1)}(y),$$

and

$$u_h^\star(x,y) = \int_{\mathbb{R}^2} K_{h_x}^{(2k+1,k+1)}(x-\xi) K_{h_y}^{(2k+1,k+1)}(y-\eta) u_h(\xi,\eta) d\xi d\eta,$$
$$v_h^\star(x,y) = \int_{\mathbb{R}^2} K_{h_x}^{(2k+1,k+1)}(x-\xi) K_{h_y}^{(2k+1,k+1)}(y-\eta) v_h(\xi,\eta) d\xi d\eta.$$

Remark 5.2.1. *In the original work [61], the authors only used the symmetric filter* (1.6) *in the interior region of the given domain over uniform meshes. Now, with the new position-dependent filter* (2.4) *introduced in Chapter 2, and the methods discussed in Chapter 4, we can treat the entire domain over both uniform and nonuniform meshes.*

After filtering the domain, we simply use an explicit Runge-Kutta scheme to calculate the streamlines from equation (5.1). More details of Runge-Kutta schemes and their error estimations can be found in standard numerical methods textbook, such as [38].

5.2.1 Numerical Results

For the numerical experiments, we consider three analytic fields given in [61], which have form:

$$\begin{aligned} z &= x + iy, \\ u &= Re(r), \\ v &= -Im(r). \end{aligned}$$

Field 1

$$\begin{aligned} r =&(z - (0.74 + 0.35i))(z - (0.68 - 0.59i)) \\ &(z - (-0.11 - 0.72i))(\bar{z} - (-0.58 + 0.64i)) \\ &(\bar{z} - (0.51 - 0.27i))(\bar{z} - (-0.12 + 0.84i))^2. \end{aligned}$$

Field 2

$$\begin{aligned} r =&(z - (0.74 + 0.35i))(\bar{z} + (-0.18 - 0.19i)) \\ &(z - (-0.11 - 0.72i))(\bar{z} - (-0.58 + 0.64i)) \\ &(\bar{z} - (0.51 - 0.27i)). \end{aligned}$$

Field 3

$$\begin{aligned} r =&(z - (0.74 + 0.35i))(z - (0.11 - 0.11i))^2 \\ &(z - (-0.11 + 0.72i))(\bar{z} - (-0.58 + 0.64i)) \\ &(\bar{z} - (0.51 - 0.27i)). \end{aligned}$$

The domain of interest is $[-1, 1] \times [-1, 1]$ for the above three vector fields. The DG approximations of these fields are given by the L^2 projection onto linear piecewise polynomial basis functions over a uniform 40×40 mesh. Once the approximation fields are obtained, we apply SIAC filters through the entire domain. Here, we point out that by using the new one-sided filter near the boundaries, we are able to deal with the entire domain $[-1, 1] \times [-1, 1]$ compared to the original results in [61] which considered only the interior region of the domain. For the streamline integration, in this example we use the second order Runge-Kutta scheme with a time step $dt = 0.01$.

For convenience, we refer to the streamlines based on the analytic fields to the "exact" streamlines, the streamlines based on the L^2 projection fields as DG streamlines, and the streamlines based on the SIAC filtered fields as filtered streamlines. Streamline integration examples based on the given vector fields are presented in Figure 5.1. Note that in Figure 5.1, the filtered streamline more closely follows the exact streamline. In some cases, one can observe that in regions where bifurcations occur, the filtered streamline follows the exact streamline, but the DG streamline is diverging away. This behavior can be attributed to enhanced accuracy and smoothness of the filtered approximation due to applying SIAC filters, and is similar to those found in Steffan et al. [61].

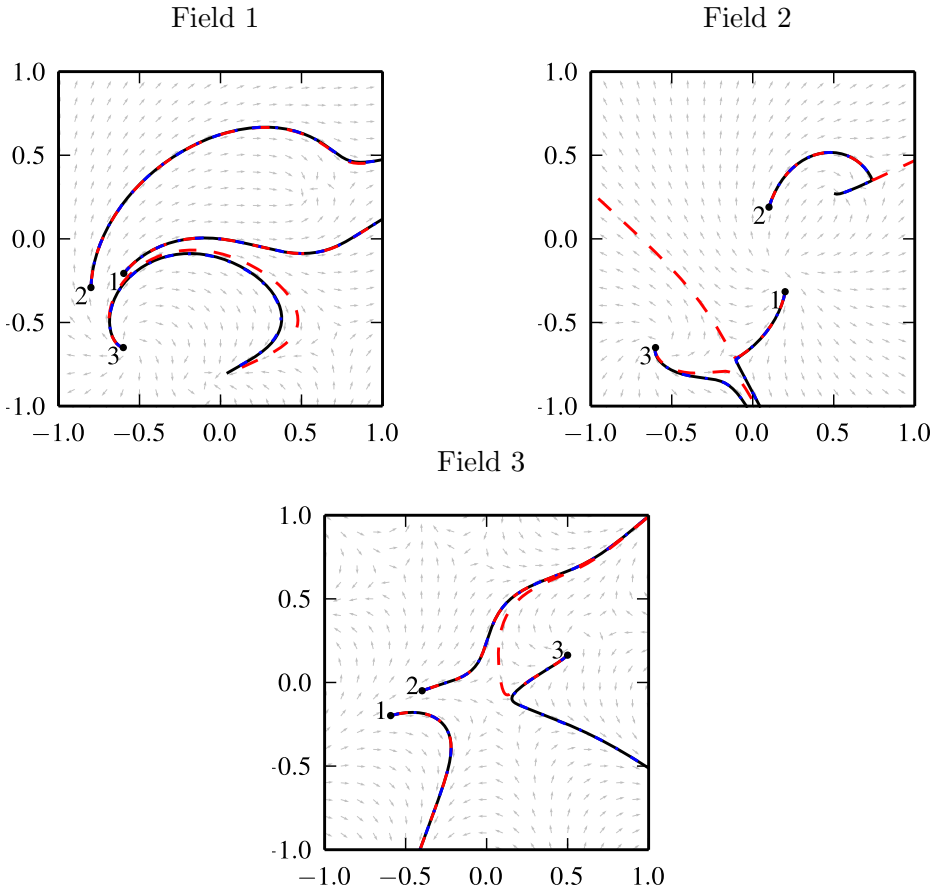


Figure 5.1: Streamlines based upon vector fields: Field 1, Field 2 and Field 3. Black solid lines denote the "exact" solution, red dashed lines are the DG streamlines and blue dashed dot lines are the filtered streamlines. Note the black lines are overlapped by the blue lines.

In addition, we use the *Hausdorff distance* metric to measure the error of the DG streamline and the filtered streamline with respect to the "exact" streamline. The Hausdorff distance is a very popular technique to define a distance between two curves.

Given two curves X and Y, their Hausdorff distance $d_H(X, Y)$ is defined by

$$d(X,Y) = \max\left\{\sup_{x\in X}\inf_{y\in Y} d(x,y), \sup_{y\in Y}\inf_{x\in X} d(x,y)\right\},$$

where $d(x, y)$ denotes the standard euclidean distance between two points. Table 5.1 presents the Hausdorff distances of the streamlines given in Figure 5.1. In Table 5.1, we observe that the filtered streamlines are close to the exact streamlines compared to the DG streamlines.

Table 5.1: The error (Hausforff distance) for the DG streamlines and the filtered streamlines in Figure 5.1.

DG / Filtered	line 1	line 2	line 3
Field 1	2.71E-2 / 1.28E-3	2.83E-2 / 9.80E-3	9.75E-1 / 1.32E-2
Field 2	1.37E-0 / 1.73E-3	2.89E-1 / 2.90E-3	7.33E-2 / 1.83E-3
Field 3	4.63E-3 / 4.48E-3	7.71E-3 / 4.90E-3	1.00E-1 / 4.70E-3

5.3 Filtering Along the Streamline

The first streamline integration method demonstrated good performance. However, the algorithm of filtering the entire domain is computationally expensive, especial for higher dimensional vector fields. In order to develop an alternative efficient algorithm, Walfisch et al. [68] proposed the idea of filtering along the streamline. This idea uses the information from the streamline instead of information from the entire domain. To implement this idea, one needs to use one-sided filters. In [68], the authors used the RS filter (1.12). In this section, we will use the newly defined position-dependent filter (2.4) which has proved to have better performance than the RS filter.

5.3.1 Backward-Differentiation Methods

In [68], the authors used the 1D one-sided filtering with Runge-Kutta schemes for the streamline integration. However, we point out these Runge-Kutta schemes require sampling the field from positions that are not along the streamline. Since the filtering proposed in the algorithm is done along the streamline, evaluating the points which are not on the streamline is inconsistent. To explain this, we give the formula for the second order Runge-Kutta scheme

$$\begin{aligned} k_1 &= hf(t_n, y_n) \\ k_2 &= hf(t_n + \frac{h}{2}, y_n + \frac{k_1}{2}) \\ y_{n+1} &= y_n + k_2. \end{aligned}$$

In the above formula, it is clear that the point $(t_n + \frac{h}{2}, y_n + \frac{k_1}{2})$ in step 2 usually does not belong to the streamline, and then we can not use filtering at this point along the streamline. The same thing occurs with any other multi-stage methods.

Also, we want the algorithm to work reasonably well for other integration curves, such as streaklines and pathlines. Based on these reasons, we use backward differentiation methods as the integrator for the streamline integration. By choosing backward differentiation for integration, it allows us to use a larger time-step which will be outside an explicit integrator's stability region.

Consider the general formula for multi-step schemes and select backward differentiation as the time-stepping method, one gets a backward differentiation formula (BDF) of the form

$$\sum_{i=0}^{p} \alpha_i \mathbf{r}^{n+1-i} = \Delta t \beta \mathbf{u}^{n+1}$$

where the coefficients for a fixed time step are given as in Table 5.3.1, see [38].

Table 5.2: Backward Differentiation Coefficients

Order p	β	α_1	α_2	α_3	α_4
1	1	-1			
2	$\frac{2}{3}$	$-\frac{4}{3}$	$\frac{1}{3}$		
3	$\frac{6}{11}$	$-\frac{18}{11}$	$\frac{9}{11}$	$-\frac{2}{11}$	
4	$\frac{12}{25}$	$-\frac{48}{25}$	$\frac{36}{25}$	$-\frac{16}{25}$	$\frac{3}{25}$

The standard practice for reaching an order (*i.e.* p) scheme is to start with a $p = 1$ scheme and "jump-start" once calculations up through the orders as more positional data becomes available. In this section, we initiate the time-stepping with the trapezoidal rule version of the BDF given by:

$$\mathbf{r}^{n+1} = \mathbf{r}^n + \Delta t \left(\frac{1}{2}\mathbf{u}^n + \frac{1}{2}\mathbf{u}^{n+1} \right).$$

Note that this is an implicit formula and hence requires a root finding technique. In this section, we use the multi-dimensional Newton's method which requires us to re-write the scheme in root finding form as:

$$\mathbf{g}(\mathbf{r}^{n+1}) = \mathbf{r}^{n+1} - \mathbf{r}^n - \Delta t \left(\frac{1}{2}\mathbf{u}^n + \frac{1}{2}\mathbf{u}^{n+1} \right) = \mathbf{0}.$$

The goal is to find the Newton's method iterate (denoted by k) such that $\mathbf{g}(\mathbf{r}^{n+1,k+1}) = 0$ (or in our case, machine zero).

To implement Newton's method, we define the Jacobian matrix

$$J\mathbf{g}(\mathbf{r}^{n+1,k}) = \begin{pmatrix} \frac{\partial g_1}{\partial x} & \frac{\partial g_1}{\partial y} \\ \frac{\partial g_2}{\partial x} & \frac{\partial g_2}{\partial y} \end{pmatrix}_{r^{n+1,k}}$$

and iterate Newton's formula given by:

$$\mathbf{r}^{n+1,k+1} = \mathbf{r}^{n+1,k} - [J\mathbf{g}(\mathbf{r}^{n+1,k})]^{-1}\mathbf{g}(\mathbf{r}^{n+1,k}). \tag{5.2}$$

Once the time-stepping has been initiated, one can work their way up the different orders p using the following formula:

$$\mathbf{r}^{n+1} = \Delta t \beta u(\mathbf{r}^{n+1}) - \sum_{i=1}^{p} \alpha_i \mathbf{r}^{n+1-i}.$$

To be able to solve the multi-dimensional Newton's method, we do the same method and construct a function:

$$\mathbf{g}(r^{n+1}) = \Delta t \beta u(\mathbf{r}^{n+1}) - \mathbf{r}^{n+1} - \sum_{i=1}^{p} \alpha_i \mathbf{r}^{n+1-i} = 0.$$

The goal is to find the Newton's method iterate (denoted by k) such that $\mathbf{g}(\mathbf{r}^{n+1,k+1}) = \mathbf{0}$.

5.3.2 Algorithm

To give the complete algorithm of filtering along the streamline, we consider a vector field Ω (triangles or quadrilaterals in two-dimension and tetrahedra, hexahedra, prisms or pyramids in three-dimensions).

First, we describe the outline of the algorithm of filtering along the streamline. Once we obtain $r^0, r^1, \ldots, r^{n-1}, r^n$, like Figure 5.2, we can evaluate the new position r^{n+1} in the following steps:

1. Construct a streamline based on the known information by using interpolation methods.
2. Rewrite the streamline from Cartesian coordinates to arc-length coordinates.
3. Apply a one-sided filter along the streamline to obtain a filtered vector field.
4. Use backward-differentiation methods to obtain the new position r^{n+1}.

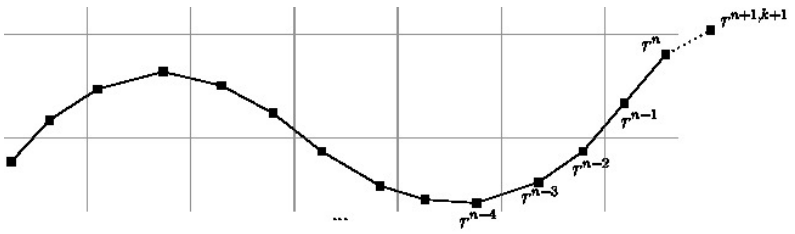


Figure 5.2: Structure of the algorithm: filtering along the streamline.

The discontinuous Galerkin approximation of the vector filed is given by

$$\mathbf{u} = [u(x, y), v(x, y)]^T.$$

A streamline $\mathbf{r}(t) = [x(t), y(t)]^T$ of this vector field $\mathbf{u}$ is an integration curve that satisfies

$$\frac{d\mathbf{r}}{dt} = \mathbf{u}(\mathbf{r}).$$

We now present the details of how to filtering along the streamline. To filtering along the streamline, we need to consider using the arc-length coordinates instead of the Cartesian coordinates as mentioned above.

By considering the arc-length coordinates, we can write the filtered vector field as

$$\mathbf{u}^\star(\mathbf{r}(s_n)) = K_h \star \mathbf{u}(s_n) = \int_{-\infty}^{\infty} K_h(s_n - s)\mathbf{u}(\mathbf{r}(s))ds,$$

where $\mathbf{r}(s_n)$ is the new position of the streamline. Here, we assume that there is enough history of the data to apply the one-sided filter (2.4) proposed in Chapter 4. Denote the support of the filter included in segment $I_1, \ldots, I_n$, where $I_j = [s_{j-1}, s_j]$, then we can rewrite the above formula as

$$\mathbf{u}^\star(\mathbf{r}(s_n)) = \sum_{j=1}^{n} \int_{I_j} K_h(s_n - s)\mathbf{u}(\mathbf{r}(s))ds.$$

For each segment I_j, let $\tau \in [0, 1]$ be the variable which we parameterize the curve, so that

$$\int_{I_j} K_h(s_n - s)\mathbf{u}(\mathbf{r}(s))ds = \int_0^1 K_h(s_n - s(\tau))\mathbf{u}(\mathbf{r}_j(\tau))\frac{ds}{d\tau}d\tau.$$

In each segment I_j, the curve $\mathbf{r}_j(\tau)$ has form

$$\mathbf{r}_j(\tau) = (x(\tau), y(\tau)),\ \tau \in [0, 1],$$

and

$$\frac{ds}{d\tau} = \sqrt{\left(\frac{dx(\tau)}{d\tau}\right)^2 + \left(\frac{dy(\tau)}{d\tau}\right)^2}, \quad s(\tau) = \int_0^\tau \frac{ds}{d\tau}d\tau.$$

In particular, for linear interpolation

$$\mathbf{r}_j(\tau) = \mathbf{x}_{j-1} + \tau(\mathbf{x}_j - \mathbf{x}_{j-1}),$$

with

$$\frac{ds}{d\tau} = \sqrt{(x_j - x_{j-1})^2 + (y_j - y_{j-1})^2} = \Delta s_j, \quad s(\tau) = \Delta s_j \tau,$$

then we have

$$\begin{aligned}&\int_{I_j} K_h(\Delta s_{m-j} + s_j - s)\mathbf{r}(s)ds \\ &= \int_0^1 K_h(\Delta s_{m-j} + (1-\tau)\Delta s_j)\mathbf{r}_j(\tau)\Delta s_j d\tau.\end{aligned}$$

In order to use backward differentiation methods and Newton iteration (5.2), we also need to calculate the Jacobian matrix

$$J\mathbf{g}(\mathbf{r}^{n+1,k}) = \begin{pmatrix} \frac{\partial g_1}{\partial x} & \frac{\partial g_1}{\partial y} \\ \frac{\partial g_2}{\partial x} & \frac{\partial g_2}{\partial y} \end{pmatrix}_{r^{n+1,k}} = \begin{pmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} \end{pmatrix}_{r^{n+1,k}} - \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

Since the above Jacobian matrix includes the derivatives, we also need to use the one-sided derivative filter proposed in Chapter 3. In Chapter 3, the formula of the filtered solution for αth derivative was given by:

$$\frac{d^\alpha u^\star}{dx^\alpha} = \frac{d^\alpha}{dx^\alpha}\left(K_h^{(2k+1,k+1+\alpha)} \star u\right) = \left(\partial_h^\alpha \tilde{K}_{h\mathbf{T}}^{(2k+1,k+1,\alpha)}\right) \star u,$$

where the filter K_h is one-sided derivative filter, $K_{h\mathbf{T}}^{(2k+1,k+1,\alpha)}$, given in (3.6).

First, we consider the $\frac{\partial u^\star}{\partial x}$ term. The filtered solution is

$$u^\star(\mathbf{r}(s_n)) = K_h \star u(s_n).$$

Using the same notation as in previous steps, we get

$$\begin{aligned}\frac{\partial}{\partial x}u^\star &= \sum_{j=1}^{n}\int_{I_j}\frac{\partial}{\partial x}K_h^{k+1+1}(s_n - s)u(\mathbf{r}(s))ds \\ &= \sum_{j=1}^{n}\int_{I_j}\left(\partial_h \tilde{K}_h^{k+1}\right)(s_n - s)\frac{\partial s}{\partial x}u(\mathbf{r}(s))ds.\end{aligned}$$

For linear interpolation, we will have

$$\begin{aligned}\frac{\partial}{\partial x}u^\star &= \sum_{j=1}^{n}\int_0^1\left(\partial_h \tilde{K}_h^{k+1}\right)(s_n - s(\tau))\frac{ds}{d\tau}\frac{d\tau}{dx}u(\mathbf{r_j}(\tau))\frac{ds}{d\tau}d\tau \\ &= \sum_{j=1}^{n}\int_0^1\left(\partial_h \tilde{K}_h^{k+1}\right)(s_n - s(\tau))u(\mathbf{r_j}(\tau))\frac{(\Delta s_j)^2}{x_j - x_{j-1}}d\tau.\end{aligned}$$

Similarly, for the remaining three terms,

$$\frac{\partial}{\partial y}u^\star = \sum_{j=1}^{n}\int_0^1\left(\partial_h \tilde{K}_h^{k+1}\right)(s_n - s(\tau))u(\mathbf{r_j}(\tau))\frac{(\Delta s_j)^2}{y_j - y_{j-1}}d\tau$$

$$\frac{\partial}{\partial x}v^\star = \sum_{j=1}^{n}\int_0^1\left(\partial_h \tilde{K}_h^{k+1}\right)(s_n - s(\tau))v(\mathbf{r_j}(\tau))\frac{(\Delta s_j)^2}{x_j - x_{j-1}}d\tau$$

$$\frac{\partial}{\partial y}v^\star = \sum_{j=1}^{n}\int_0^1\left(\partial_h \tilde{K}_h^{k+1}\right)(s_n - s(\tau))v(\mathbf{r_j}(\tau))\frac{(\Delta s_j)^2}{y_j - y_{j-1}}d\tau$$

Remark 5.3.1. *The integration can be calculated by using Gauss quadrature, but we need to consider both the DG breaks and the filter breaks as mentioned in Chapter 1.*

5.3.3 Preliminary Results

Time Cost

First, we check the computational cost for the algorithm of filtering along the streamline and compare it with the method of filtering the entire field. To avoid interruption, we test two algorithms for the simplest vector field, a constant vector field $\mathbf{u} = [1, 1]^T$. Then the streamlines will be straight lines, and the numerical approximations should be exactly the same, see Figure 5.3. Also, we provide Table 5.3, which shows the computational cost (time) of using these two algorithms. From the table, we can clearly see that filtering along the streamline can significantly reduce the computational cost, which is almost 70 times faster than filtering the entire field.

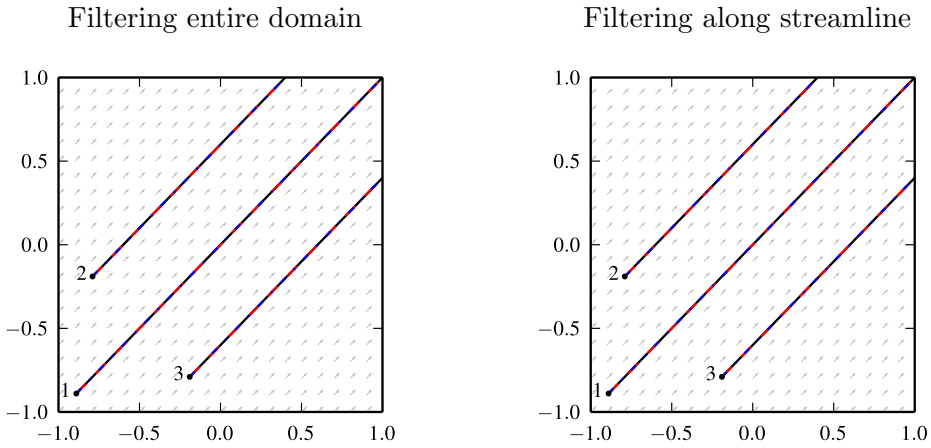


Figure 5.3: Straight-line streamlines based upon vector field $\mathbf{u} = [1, 1]^T$. We note that the numerical approximations are exact.

Table 5.3: Number of steps, time per integration step and ratio of filtering along the streamline to filtering the entire field time per step required to calculate three streamlines on the constant vector field $\mathbf{u} = [1, 1]^T$.

	Filtering entire field		Filtering along streamline		
streamline	Steps	Time/Step(sec)	Steps	Time/Step(sec)	Ratio
1	190	2.64e-02	190	3.79e-04	69.72
2	120	2.61e-02	120	3.67e-04	71.14
3	120	2.62e-02	120	3.67e-04	71.36

Then, we consider the same three analytic fields used in the previous section. The domain is $[-1, 1] \times [-1, 1]$, and the DG approximations are given by the L^2 projection, polynomial $\mathbb{P}^1$, onto a uniform 40×40 mesh. Here, the filtered streamline is obtained by using the algorithm of filtering along the streamline. For streamline integration, we use the second order backward-differentiation method with time step $dt = 0.01$. Streamline

integration examples based the given vector fields are presented in Figure 5.4. Unlike Figure 5.1, the difference between the filtered streamline and the DG streamline is quite small and hard to observe from the plots (unless zoom in). However, we can still compare the error by using the Hausdorff distance. Table 5.4 presents the Hausdorff distances of the streamlines given in Figure 5.4. From Table 5.1, we observe that the filtered streamlines are slightly close to the exact streamlines compared to the DG streamlines, but the advantage is not obvious.

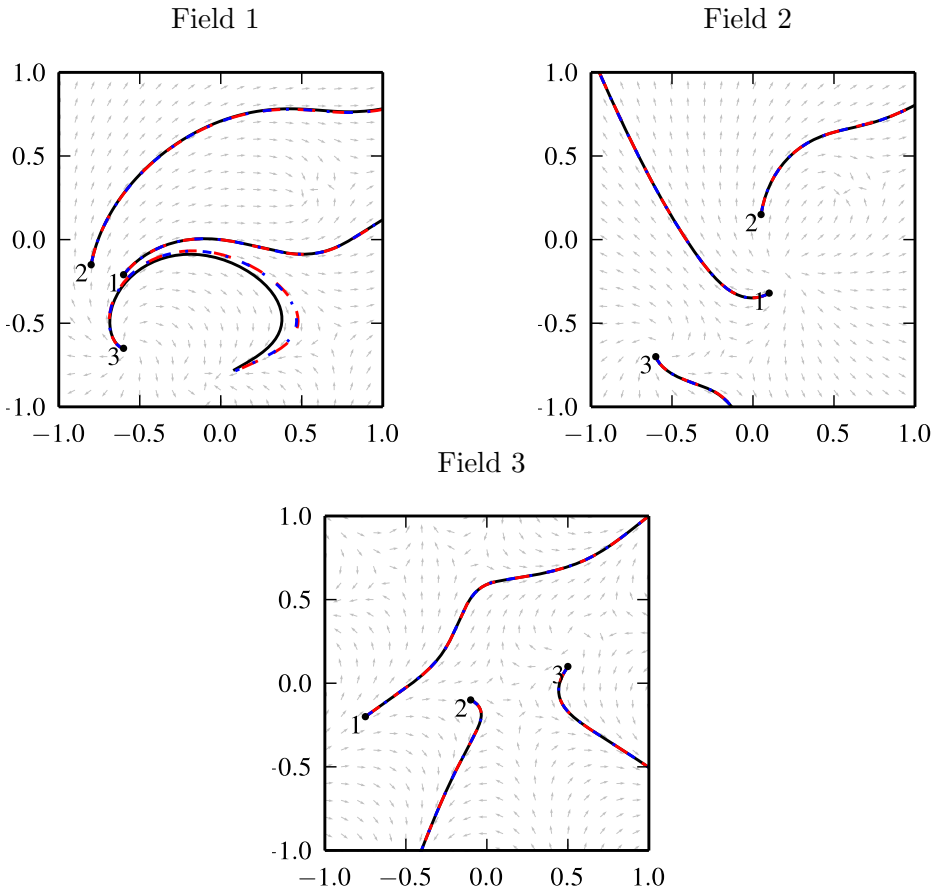


Figure 5.4: Streamlines based upon vector fields: Field 1, Field 2 and Filed 3. Black solid lines denote the "exact" solution, red dashed lines are the DG streamlines and blue dashed dot lines are the filtered streamlines. Using one-sided filtering along the streamline.

Remark 5.3.2. *The filtered streamline obtained by applying the method of filtering along the streamline is less accurate compared to the method of filtering the entire domain. It is a method to sacrifice accuracy for speed.*

Remark 5.3.3. *The results presented in this section are only preliminary results. In order to obtain optimal results, we must answer the following questions:*

Table 5.4: The error (Hausforff distance) for the DG streamlines and the filtered streamlines in Figure 5.4.

Before / After	line 1	line 2	line 3
Field 1	4.72E-3 / 3.49E-3	3.45E-2 / 2.37E-3	1.00E-1 / 9.08E-2
Field 2	8.09E-2 / 2.00E-2	2.26E-3 / 1.17E-3	5.00E-3 / 3.34E-3
Field 3	2.70E-3 / 4.03E-3	1.54E-2 / 8.24E-4	1.85E-2 / 1.26E-2

- *Which method should be used for reconstructing the streamline: linear interpolation, Hermite interpolation or others?*
- *Are backward-differentiation methods the best integration method when filtering along the streamline?*
- *Which one-sided filter is the most suitable one for filtering along the streamline?*
- *How should the filter scaling be chosen for filtering along the streamline? When filtering along the streamline, in order to use the one-dimensional one-sided filter, we must consider arc-length coordinates. It follows that we have to use the filter over a nonuniform mesh. As mentioned in Chapter 4, the choice of the filtering scaling is directly related to the final accuracy.*

5.3.4 Which One-Sided Filter?

For filtering along the streamline, one must use a one-sided filter. It follows that an interesting question is which one-sided filter is the most suitable one for this algorithm. As introduced earlier, there are three one-sided filters: the RS filter (1.12), the SRV filter (1.14) and the new filter (2.4). In the previous parts of this chapter, we use only the new filter since the other two filters do not acquire any information at the point where the filtered value needs to be computed. However, it is still interesting to compare the behaviors of using these three one-sided filters for streamline integration.

In Figure 5.5 and Table 5.5, we compare the results of using these three one-sided filters over Field 2. One can observe that the new filter provides the most accurate results, then the RS filter. For the SRV filter, the results are significantly worse than the other two one-sided filters and the DG solutions.

Table 5.5: The error (Hausforff distance) for the DG streamlines and the filtered streamlines with three different one-sided filters in Figure 5.5.

	DG	RS filter	SRV filter	New filter
line 1	8.09E-2	6.71E-2	5.11E-2	2.00E-2
line 2	2.26E-3	1.41E-3	2.21E-1	1.17E-3
line 3	5.00E-3	3.61E-3	4.36E-2	3.34E-3

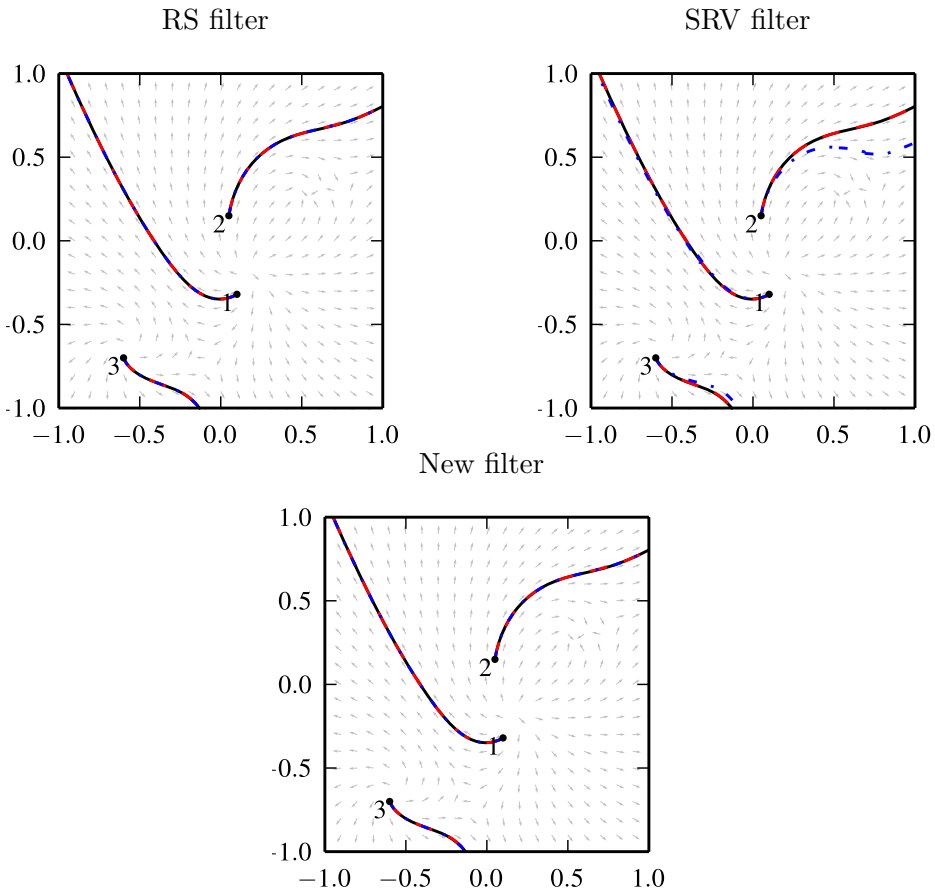


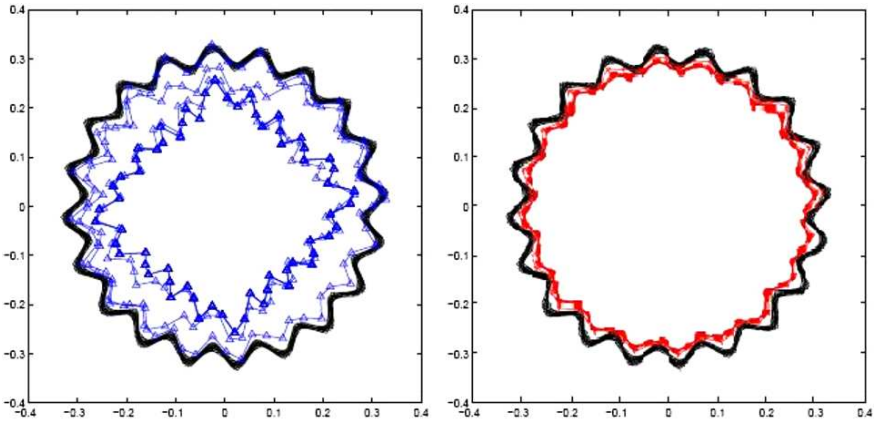
Figure 5.5: Streamlines based upon Field 2. Black solid lines denote the "exact" solution, red dashed lines are the DG streamlines and blue dashed dot lines are the filtered streamlines. Using three different one-sided filters.

Then, we consider an interesting field given by [68] which was defined in polar coordinates by

$$\begin{bmatrix} u(r,\theta) \\ v(r,\theta) \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\cos(20\theta)\cos(\theta) - r\sin(\theta) \\ \frac{1}{2}\cos(20\theta)\sin(\theta) + r\cos(\theta) \end{bmatrix}. \tag{5.3}$$

This field has streamlines which are oscillating closed circuits. In [68], the author compared the results of the DG streamline and the filtered streamline with using the RS filter, see Figure 5.6.

In order to compare the results of using different one-sided filters, we redo the example with the same settings by using SRV filter (1.14) and the new filter (2.4), see Figure 5.7. Comparing the results in Figures 5.6 and 5.7, the new filter is still the winner. However, we point out the results of using the RS filter are also acceptable, and the SRV filter is diverging away from the actual streamline. Through the above examples, we can clearly see that for filtering along the streamline one can use either the new filter or the RS filter, but the SRV filter is not a suitable choice.



(a) Without filtering (b) With filtering

Figure 5.6: Streamline integrations [68] based upon vector field (5.3). Using the RS filter.

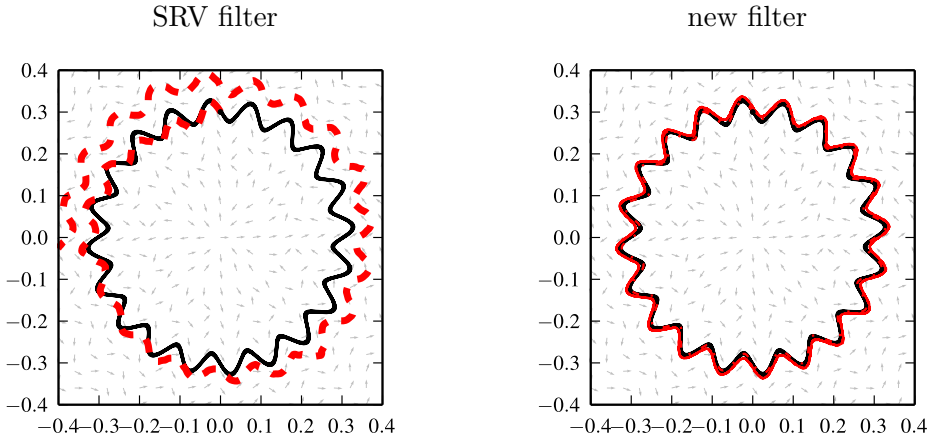


Figure 5.7: Streamline integrations based upon vector field (5.3). Black solid lines denote the "exact" solution and red dashed dot lines are the filtered streamlines. Using the SRV filter and the new filter.

5.4 Conclusion

The SIAC filtering technique has demonstrated its ability to improve the continuity of the discontinuous Galerkin solution and maintain the accuracy. Therefore, SIAC filtering has potential value as a preprocessing tool prior to other techniques such as visualization techniques. In this chapter, we presented the preliminary results of applying SIAC filters for streamline visualization. We briefly discussed how to apply SIAC filters for the discontinuous Galerkin approximations that results in more accurate

streamline placements. There are two ways to use SIAC filters for streamline integration. The first one is using the filter introduced in Chapter 2 to filter the entire vector field, and then perform the streamline integration. This method is mature and robust, and the filtered streamlines are closer to the exact streamlines compared to the DG streamlines. Also, applying SIAC filtering before the streamline integration, to a great extent, has avoided diverging situations for the streamlines. The second method is proposed based on computational considerations. This method considers filtering only along the streamline rather than the entire domain. Clearly, filtering along the streamline is more efficient compared to filtering the entire domain. However, as mentioned in this chapter, the method of filtering along the streamline is on-going research, and it is not robust as the method of filtering the entire domain. Through some preliminary results, we indicate there are several points that need to be discussed or resolved:

- The first question is: which one-sided filter to use? The algorithm of filtering along the streamline requires using a one-sided SIAC filter. Based on numerical experiments, we point out we should make a choice between the RS filter (1.12) and the new filter (2.4). The SRV filter (1.14) is not a suitable choice for streamline integration.

- The second question is: how should the filter scaling be chosen? As mentioned in Chapter 4, the filter scaling is the crucial factor for filtering over nonuniform meshes, and algorithm of filtering along the streamline is always performed over nonuniform meshes no matter the original tessellation.

- There are some remaining details about the implementation of the algorithm, such as which interpolation method to reconstruct the streamline, and which integration method for the streamline integration.

If these are solved, this algorithm could be of great value to visualization techniques. Additionally, the idea of this algorithm that is dealing with multi-dimensional data in a one-dimensional operation has great potential value for many other applications.

Further works relate this chapter are:

- Further investigation of the algorithm of filtering along the streamline and develop it into a robust algorithm.

- Extend the algorithm to other visualization techniques, such as other integration curves (streakline, pathline, etc.).

- Theoretical analysis of the algorithm, such as error estimates.

6

Further Inverstigation of SIAC Filter

In Chapter 1, we introduced the original definition and properties of the SIAC filter. In this chapter, we will investigate further details of the SIAC filter and its variations. For convenience, without specification the filters in this chapter are symmetric filters. However, one can easily extend the results in this chapter to one-sided filters.

6.1 Structure of SIAC Filter

In Chapter 1, we discussed the structure of the original symmetric SIAC filter - a linear combination of central B-splines. Following this construction an interesting question is: is there another structure that allows one to extract the superconvergence order of $2k+1$ from DG solutions?

To answer this question, we recall the necessary components to extract the superconvergence order of $2k+1$, Theorem 1.3.4. Through the proof of Theorem 1.3.4 given in [25], one can see that there are four main components behind Theorem 1.3.4, namely, Lemma 1.2.2, Theorem 1.2.1, Property 1.3.2 and Property 1.3.3:

- Lemma 1.2.2 shows that the L^2 norm of a function can be bounded by the negative order norm of the derivatives of the same function. It is dependent on the regularity of the function.
- Theorem 1.2.1 proves that the DG solution and its divided differences have superconvergence order of $2k+1$ in the negative order norm. It is the superconvergence of the DG solution.
- Property 1.3.2 demonstrates that the filter has the ability to reproduce polynomials by convolution. It is due to equation (1.8).
- Property 1.3.3 allows us to express derivatives in term of divided difference quotients. It is due to the structure of the SIAC filter, more precisely, the central B-splines.

Through the above analysis, we know the key structure of the filter is to provide the ability to express derivatives in term of divided differences. Therefore, the task is to

find a basis function $\phi^{(\ell)}$, such that

$$\frac{d}{dx}\phi_H^{(\ell)}(x) = \partial_H \phi^{(\ell-1)}(x) = \frac{1}{h}\left(\phi^{(\ell-1)}(x+h/2) - \phi^{(\ell-1)}(x-h/2)\right), \tag{6.1}$$

then we can construct the filter by

$$K^{(2k+1,k+1)} = \sum_{\gamma=0}^{2k} c_\gamma^{(2k+1,k+1)} \phi^{(k+1)}(x-(-k+\gamma)), \tag{6.2}$$

where $c_\gamma^{(2k+1,k+1)}$ is decided by (1.8).

In order to find such a basis function $\phi^{(\ell)}$, we take the Fourier transform of equation (6.1)

$$2\pi i\xi\hat{\phi}^{(\ell)} = \hat{\phi}^{(\ell-1)}e^{\pi i\xi} - \hat{\phi}^{(\ell-1)}e^{-\pi i\xi} \implies \hat{\phi}^{(\ell)} = \hat{\phi}^{(\ell-1)}\frac{\sin(\pi\xi)}{\pi\xi}.$$

It is easy to verify that $\hat{\chi}_{[-1/2,1/2]} = \frac{\sin(\pi\xi)}{\pi\xi}$, then we have

$$\phi^{(\ell)} = \phi^{(\ell-1)} \star \chi_{[-1/2,1/2]}. \tag{6.3}$$

Once we decide the initial function $\phi^{(1)}$, the filter, constructed by (6.2) and (6.3), can extract the superconvergence order of $2k+1$ from the DG solution. The simplest and natural choice is $\phi^{(1)} = \chi_{[-1/2,1/2]}$, which leads to the central B-spline filter. Of course, we can choose another initial function, such as $\phi^{(1)} = \cos(\pi x)\chi_{[-1/2,1/2]}$, $\phi^{(k)}$ with $k = 2, 3, 4$. These are shown in Figure 6.1.

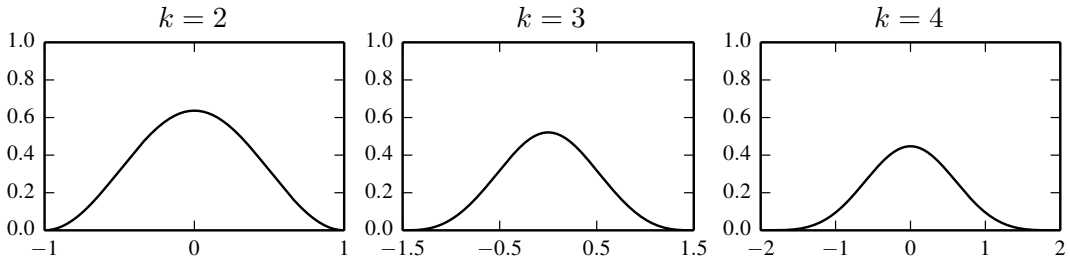


Figure 6.1: Basis $\phi^{(k)}$, which satisfies (6.3), with $\phi^{(1)} = \cos(\pi x)\chi_{[-1/2,1/2]}$.

We can see that the filter given in (6.2) constructed by the basis function $\phi^{(k+1)}$, which satisfies (6.3), also has Property 1.3.3. Then, the following question is: do there exist constants $c_\gamma^{(2k+1,k+1)}$ such that filter (6.2) reproduces polynomials by convolution until degree up to degree $2k$? To answer this question, we present Theorem 6.1.1 for the more general situation.

Theorem 6.1.1. *Assume ϕ_γ, $\gamma = 0, \ldots, r$ are $r+1$ normalized linear independent functions, each with has compact support. Then the linear system*

$$\sum_{\gamma=0}^{r} c_\gamma \int_{-\infty}^{\infty} \phi_\gamma(\xi)(x-\xi)^m d\xi = x^m, \quad m = 0, 1, \ldots, r \tag{6.4}$$

has a unique solution.

Proof. For convenience, we denote that

$$\begin{aligned} A(\gamma, m)(x) &= \int_{-\infty}^{\infty} \phi_\gamma(\xi)(x-\xi)^m d\xi, \quad \gamma, m = 0, 1, \ldots, r, \\ &= x^m + \sum_{j=0}^{m-1} \lambda_{\gamma,m}^j x^j, \end{aligned}$$

where $\lambda_{\gamma,m}^j = \int_{-\infty}^{\infty} \phi_\gamma(\xi)\,(-\xi)^j \binom{m}{j} d\xi$. Then, the conclusion in Theorem 6.1.1 is equivalent to matrix $A(\gamma, m)$ is nonsingular, in other words, we only need to prove the $r+1$ rows $\{A(\gamma, m)\}_{\gamma=0}^r$ are linear independent.

Assume there exist constants $b_m, m = 0, \ldots, r$ such that

$$\sum_{m=0}^{r} b_m A(\gamma, m)(x) = 0, \quad \gamma = 0, \ldots, r.$$

By substituting the form of $A(\gamma, m)$, we have

$$\sum_{m=0}^{r} b_m \left(x^m + \sum_{j=0}^{m-1} \lambda_{\gamma,m}^j x^j \right) = 0 \Rightarrow \sum_{m=0}^{r} \left(b_m + \sum_{j=m+1}^{r} b_j \lambda_{\gamma,i}^m \right) x^m = 0.$$

Since the above relation is true for all $x \in \mathbb{R}$, we know

$$b_m + \sum_{j=m+1}^{r} b_j \lambda_{\gamma,i}^m = 0 \Rightarrow \begin{cases} b_r = 0 \\ b_{r-1} = -\lambda_{\gamma,r}^{r-1} b_r = 0 \\ \cdots \\ b_0 = -\left(\sum_{j=1}^{r} \lambda_{\gamma,j}^0 b_j \right) = 0. \end{cases}.$$

It follows that the $r+1$ rows of linear system (6.4) is linear independent. □

Since $\phi^{(k+1)}(x+k-\gamma), \gamma = 0, \ldots, r$, are $r+1$ linear independent compact functions, Theorem 6.1.1 indicates that there exists unique constants $c_\gamma^{(2k+1,k+1)}$ such that filter (6.2) reproduces polynomials by convolution up to degree $2k$. For example, the filters with $\phi^{(1)} = \cos(\pi x)\chi_{[-1/2,1/2]}$ are presented in Figure 6.2.

Finally, we know that there are infinitely many structures of the SIAC filter that allow us to extract the superconvergence order of $2k+1$ from the DG solutions.

Theorem 6.1.2. *Under the same conditions in Theorem 1.3.4, the filter given in* (6.2)*, then*

$$\|u - K_h^{(2k+1,k+1)} \star u_h\|_{0,\Omega_0} \leq C h^{2k+1}.$$

Proof. The proof is the same as in Theorem 1.3.4 □

Here, we point out that if we limit a SIAC filter to be a piecewise polynomial, then the original symmetric filter constructed by central B-splines has the simplest formula.

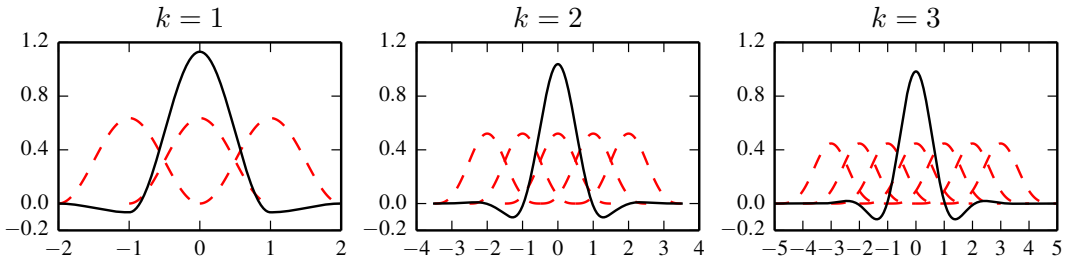


Figure 6.2: $K^{(2k+1,k+1)}$ given in (6.2) constructed by basis $\phi^{(k+1)}$, which satisfies (6.3), with $\phi^{(1)} = \cos(\pi x)\chi_{[-1/2,1/2]}$.

Remark 6.1.1. *Through the technique discussed in this section, we can create an infinitely smooth SIAC filter to enhance the smoothness of the filtered solutions. For example, we can choose*

$$\phi^{(1)} = \begin{cases} \exp\left(-\frac{1}{1-4x^2}\right), & |x| < \frac{1}{2} \\ 0, & |x| \geq \frac{1}{2} \end{cases}.$$

Then we can obtain a filter which has the same support size as the original filter, but it belongs to $\mathcal{C}_0^\infty$. It follows that the filtered solution $u_h^\star \in \mathcal{C}^\infty$ and also has accuracy order of $2k+1$.

6.2 The Order of B-splines

As mentioned in the previous chapters, the order of the SIAC filter plays a very import role, such as for derivative filters. According to the error estimates, one must use a certain order of B-splines to convert all the required derivatives into divided differences. This is the reason we can not use a linear filter to filter a cubic DG approximations.

6.2.1 The Lowest Order of B-splines

This leads to the following question: what is the lowest order of B-spline that ensures obtaining accuracy order of $2k+1$. Usually, we choose the order of B-splines to be $k+1$, which means the filter $K^{(2k+1,k+1)}$ is a piece-wise polynomial of degree k.

One can verify that the filtered solution, $u_h^\star = K_h^{(2k+1,k+1)} \star u_h$, is a piecewise polynomial of degree $2k+1$. Based on the approximation theorem, as a polynomial of degree $2k+1$, the filtered solution does not obtain its optimal accuracy order $2k+2$. In fact, due to the superconvergence property of the DG solution, Theorem 1.2.1, the accuracy order of $2k+2$ is usually impossible unless u_h is a L^2 projection of the exact solution.

However, it leads to a conjecture that we can use filter $K^{(2k+1,k)}$, order k filter, to approach the accuracy order of $2k+1$.

Theorem 6.2.1. *Under the same conditions in Theorem 1.3.4, then*

$$\|u - K_h^{(2k+1,k)} \star u_h\|_{0,\Omega_0} \leq Ch^{2k+1}.$$

Proof. The proof is the same as in Theorem 1.3.4. □

Remark 6.2.1. *Theorem 6.2.1 also explains the behavior that after using the filter $K^{(2k+1,k+1)}$, the first derivative of the filtered solution has accuracy order $2k+1$ not $2k$ as theorem in [56] suggested.*

$$\left\|\partial_x u - \partial_x \left(K_h^{(2k+1,k+1)} \star u_h\right)\right\|_{0,\Omega_0} \leq Ch^{2k+1}.$$

Since $K_h^{(2k+1,k)} \star u_h$ is only a piecewise polynomial of degree $2k$, we can claim that the accuracy order $2k+1$ is already optimal. Also, the approximation theorem tells us the order of filter can not be lower than k, and order k is already the lowest order if an accuracy order of $2k+1$ is desired. Now, we compare the differences between using the $k+1$th order filter and the kth order filter.

Example 6.2.2. *Consider a linear hyperbolic equation*

$$\begin{aligned} u_t + u_x &= 0, \quad (x,t) \in [0,1] \times (0,T] \\ u(x,0) &= \sin(2\pi x) \end{aligned}$$

with final time $T = 1$ over uniform meshes. The L^2 and L^∞ norm errors and respective accuracy order are given in Table 6.1. Figure 6.3 shows the point-wise errors in log scale. The respective results of the DG approximation can be found in Example 1.3.5.

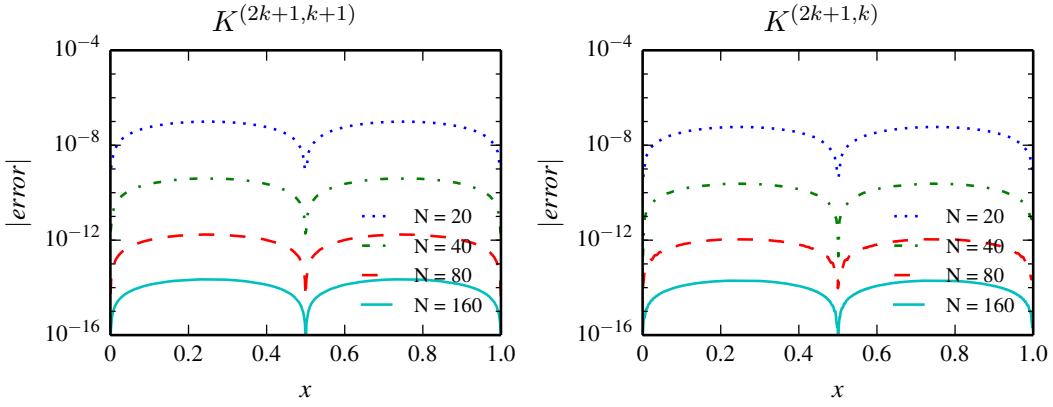


Figure 6.3: Comparison of the point-wise errors in log scale of the filtered solutions with the kth and $(k+1)$th symmetric filter. The DG basis are polynomials $\mathbb{P}^3$ for a linear advection equation.

Table 6.1: L^2- and $L^\infty-$errors for the filtered solutions $u_h^\star$ with the kth and $(k+1)$th symmetric filter for a linear advection equation.

	$K^{(2k+1,k+1)}$				$K^{(2k+1,k)}$			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\mathbb{P}^1$								
20	1.97E-03	–	2.80E-03	–	1.94E-03	–	2.75E-03	–
40	2.44E-04	3.02	3.46E-04	3.02	2.42E-04	3.00	3.44E-04	3.00
80	3.02E-05	3.01	4.28E-05	3.01	3.02E-05	3.00	4.29E-05	3.00
160	3.76E-06	3.01	5.33E-06	3.01	3.77E-06	3.00	5.35E-06	3.00
$\mathbb{P}^2$								
20	4.11E-06	–	5.82E-06	–	3.09E-06	–	4.39E-06	–
40	9.49E-08	5.44	1.34E-07	5.44	7.88E-08	5.29	1.12E-07	5.29
80	2.49E-09	5.25	3.52E-09	5.26	2.24E-09	5.14	3.19E-09	5.14
160	7.75E-11	5.00	1.10E-10	5.00	7.38E-11	4.92	1.05E-10	4.93
$\mathbb{P}^3$								
20	6.97E-08	–	9.86E-08	–	4.15E-08	–	5.87E-08	–
40	2.83E-10	7.95	4.00E-10	7.95	1.70E-10	7.93	2.40E-10	7.93
80	1.23E-12	7.85	1.73E-12	7.85	7.82E-13	7.76	1.11E-12	7.76
160	1.59E-14	6.27	2.25E-14	6.27	1.41E-14	5.79	2.00E-14	5.79

Table 6.1 shows that both the kth and $(k+1)$th order filters can give us accuracy order of $2k+1$ for the filtered solutions. Here, we point out although the difference is quite small, and the filtered solutions obtain by using the kth order filter have better accuracy compared to using the $(k+1)$th order filter. These results are consistent with studies in Chapter 4 that with the higher order filter, the optimal scaling is increased.

Figure 6.3 reveals that for high order polynomials, such as $\mathbb{P}^3$, the filtered solutions look almost the same as filters $K^{(2k+1,k)}$ and $K^{(2k+1,k+1)}$. However, if we check the results for $\mathbb{P}^1$ in Figure 6.4, we can see that using the $(k+1)$th order filter generates a smoother result than using the kth order filter.

The conclusion is simple, using kth order filter is computationally more efficient but the filtered solutions are less smooth compared to using $(k+1)$th order filter. Also, we note that for higher order cases, $k \geq 3$, the difference is negligible.

6.2.2 Inexact Gaussian Quadrature Approach

After discussing using lower order B-splines, we now present the benefits of using higher order B-splines. As discussed in Chapter 1, the basic operation of the filtering process is the convolution of the DG solution against the filter. Usually, this convolution operator is calculated by using Gaussian quadrature. As mentioned in Section 1.3.4, since the DG solution is discontinuous at the element interface, and the filter is also a piecewise polynomial, one has to consider the breaks in integration of both. In other words, one needs to divide the integration region into several subintervals such that in each subinterval both the DG solution and the filter are C^∞ polynomials (not piecewise polynomials). However, the Gaussian quadrature is quite costly especially

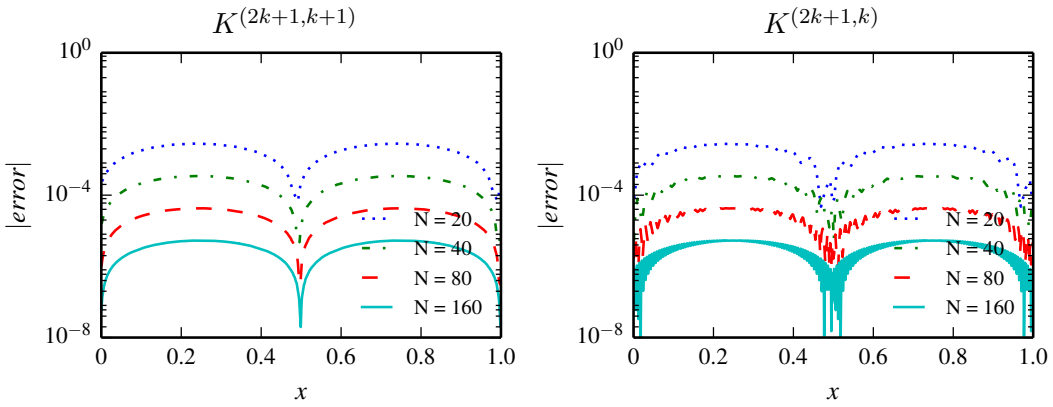


Figure 6.4: Comparison of the point-wise errors in log scale of the filtered solutions with the kth and $(k+1)$th symmetric filter. The DG basis are polynomials $\mathbb{P}^1$ for a linear advection equation.

for unstructured meshes and high-dimensional cases. Therefore, [50] proposed an idea of using inexact integration to overcome this issue by ignoring the breaks of the filter. Compared to the DG solution which is only weakly continuous at the DG breaks, the filter still has $\mathcal{C}^{k-1}$ continuity at the filter breaks. Therefore, numerically we can use some techniques to overcome the filter breaks, but not the DG breaks. We note that, in this section, the inexact Gaussian quadrature represents using the Gaussian quadrature to calculate the convolution without considering the filter breaks.

First, we present Example 6.2.3 to show what happens if we ignore the filter breaks.

Example 6.2.3. *Consider a linear hyperbolic equation*

$$\begin{aligned} u_t + u_x &= 0, \quad (x,t) \in [0,1] \times (0,T] \\ u(x,0) &= \sin(2\pi x) \end{aligned}$$

with final time $T = 1$ over uniform meshes. The L^2 and L^∞ norm errors and respective accuracy orders are given in Table 6.2, and Figure 6.5 shows the point-wise errors in log scale.

In Table 6.2, we can see that although the filtered solutions still have better accuracy compared to the DG solutions, the accuracy order drops down from $2k+1$ due to the inexact Gaussian quadrature. However, we note that using the inexact Gaussian quadrature leads to less accurate filtered solutions compared to using the exact Gaussian quadrature, see Table 1.1. Also, the filtered solutions have many oscillations in the plots in Figure 6.5.

In [50], the authors increased the Gaussian quadrature points to overcome the drawbacks of using the inexact Gaussian quadrature. In this section, we propose an alternative way, which is more natural. Increasing the regularity of the filter by using higher order B-splines. The losts of accuracy of using inexact Gaussian quadrature is

Table 6.2: L^2- and $L^\infty-$errors for the DG approximation u_h and the filtered solution $u_h^\star$ with the inexact Gaussian quadrature for a linear advection equation.

	DG error				After filtering			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
				$\mathbb{P}^1$				
20	4.60E-03	–	1.13E-02	–	1.75E-03	–	2.72E-03	–
40	1.09E-03	2.08	3.21E-03	1.82	1.89E-04	3.21	3.30E-04	3.04
80	2.67E-04	2.02	8.49E-04	1.92	1.73E-05	3.45	3.92E-05	3.07
160	6.65E-05	2.01	2.18E-04	1.96	1.64E-06	3.40	4.44E-06	3.14
				$\mathbb{P}^2$				
20	1.07E-04	–	3.67E-04	–	4.38E-06	–	6.57E-06	–
40	1.34E-05	3.00	4.62E-05	2.99	2.12E-07	4.37	3.96E-07	4.05
80	1.67E-06	3.00	5.78E-06	3.00	2.37E-08	3.16	4.64E-08	3.09
160	2.09E-07	3.00	7.23E-07	3.00	2.95E-09	3.01	5.78E-09	3.01
				$\mathbb{P}^3$				
20	2.06E-06	–	6.04E-06	–	5.34E-08	–	9.92E-08	–
40	1.29E-07	4.00	3.80E-07	3.99	1.18E-09	5.50	2.40E-09	5.37
80	8.07E-09	4.00	2.38E-08	4.00	8.63E-11	3.77	1.73E-10	3.79
160	5.04E-10	4.00	1.49E-09	4.00	5.44E-12	3.99	1.09E-11	3.99

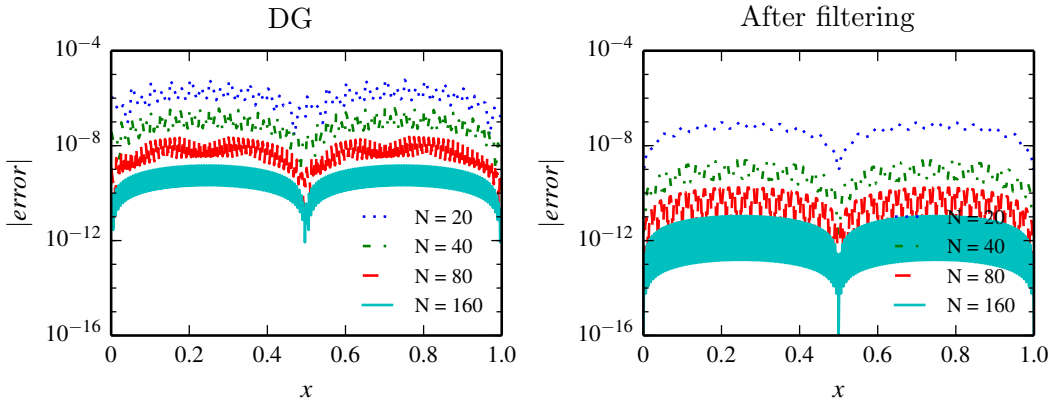


Figure 6.5: Comparison of the point-wise errors in log scale of the DG approximation u_h and the filtered solution $u_h^\star$ with the inexact Gaussian quadrature. The DG basis are polynomials $\mathbb{P}^3$ for a linear advection equation.

due to the regularity (or continuity) of the filter not being sufficient at the filter breaks. One can simply increase the order of B-splines to increase the regularity of the filter. In Table 6.3 and Figure 6.6, we present the filtered solutions using B-splines of order $k+2$ and order $k+3$. The results suggest that by increasing the order of B-splines, the filtered solutions with inexact Gaussian quadrature can be improved to the same level of using exact quadrature. In addition, we point out that increasing the order

of B-splines also slightly increases the support size of the filter. However, the extra cost of the increased support size is negligible compare to using the exact Gaussian quadrature.

Table 6.3: L^2- and $L^\infty-$errors for the filtered solution $u_h^\star$ with the inexact Gaussian quadrature for a linear advection equation. The filters used are $K^{(2k+1,k+2)}$ and $K^{(2k+1,k+3)}$.

	$K^{(2k+1,k+2)}$				$K^{(2k+1,k+3)}$			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
				$\mathbb{P}^1$				
20	2.03E-03	–	2.87E-03	–	2.09E-03	–	2.96E-03	–
40	2.47E-04	3.04	3.50E-04	3.04	2.51E-04	3.06	3.56E-04	3.06
80	3.05E-05	3.02	4.31E-05	3.02	3.07E-05	3.03	4.35E-05	3.03
160	3.78E-06	3.01	5.34E-06	3.01	3.80E-06	3.02	5.37E-06	3.02
				$\mathbb{P}^2$				
20	5.47E-06	–	7.76E-06	–	7.26E-06	–	1.03E-05	–
40	1.16E-07	5.56	1.65E-07	5.55	1.45E-07	5.64	2.05E-07	5.64
80	2.76E-09	5.39	4.01E-09	5.37	3.28E-09	5.47	4.64E-09	5.47
160	7.85E-11	5.14	1.17E-10	5.09	8.99E-11	5.19	1.27E-10	5.19
				$\mathbb{P}^3$				
20	1.08E-07	–	1.53E-07	–	1.59E-07	–	2.25E-07	–
40	4.38E-10	7.95	6.20E-10	7.95	6.44E-10	7.95	9.10E-10	7.95
80	1.85E-12	7.89	2.63E-12	7.88	2.65E-12	7.93	3.74E-12	7.93
160	1.93E-14	6.58	2.89E-14	6.51	2.14E-14	6.95	3.03E-14	6.95

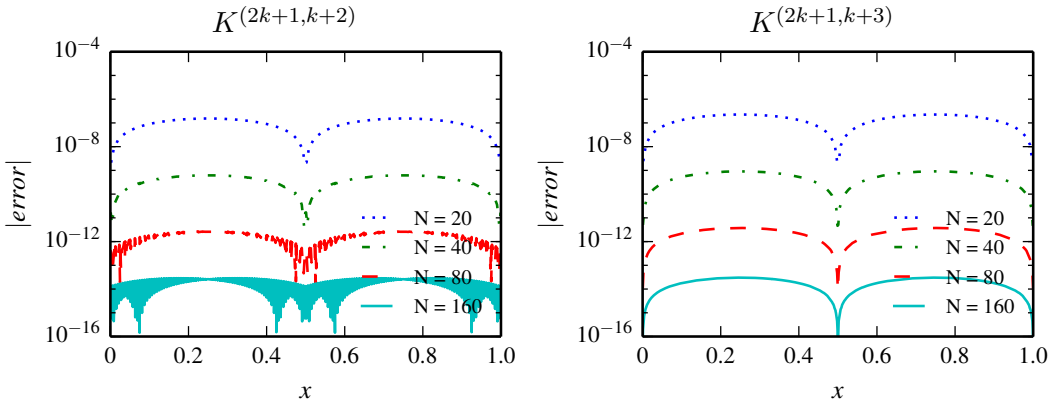


Figure 6.6: Comparison of the point-wise errors in log scale of the filtered solutions with the inexact Gaussian quadrature. The filters used are $K^{(2k+1,k+2)}$ and $K^{(2k+1,k+3)}$. The DG basis are polynomials $\mathbb{P}^3$ for a linear advection equation.

The idea of the inexact Gaussian quadrature is intended to reduce the compu-

tational cost of using SIAC filters. The benefits are not obvious for one-dimensional uniform mesh cases. However, this idea has great potential value for multi-dimensional applications especially for unstructured meshes. For example, consider the two dimensional unstructured triangular meshes, it is computationally expensive to divide the integration region into subregions, and numerical quadrature over these irregular subregions is also very expensive. The computational costs will become more expensive in three-dimensional cases, such as tetrahedral meshes. This is the reason a method that allows us to ignore the filter breaks has great potential values with respect to the computational considerations, and it is a subject of future work.

6.3 SIAC Filtering for Wave Functions

Wave functions are used in many branches of mathematics, science and engineering. Mathematically, wave propagation problems are usually described by hyperbolic equations, and developing efficient methods and algorithms to solve wave-related application problems becomes an interesting task. In this section, we will introduce the preliminary methods of using SIAC filters to enhance the discontinuous Galerkin solutions that relate to wave functions.

6.3.1 Sufficient Elements of the DG Approximation

It is well known that when using piecewise polynomials to approximate a wave function, one should at least use a certain number elements-per-wavelength. In other words, in order to approximate a wave function, we need sufficient resolution for the DG approximation.

Consider a simple example, a sine function with λ wavenumber in the domain $[0, 1]$, $\sin(2\lambda\pi x)$. First we expand $\sin(2\lambda\pi x)$ into Legendre polynomials. Denote $\xi = 2x - 1$, then we have

$$\sin(2\lambda\pi x) = \sin(\lambda\pi(\xi + 1)), \qquad \xi \in [-1, 1].$$

The Legendre expansion can be written as

$$\sin(\lambda\pi(\xi + 1)) = \sum_{n=0}^{\infty} a_n P_n(\xi),$$

where P_n is the Legendre polynomials of degree n over $[-1, 1]$ and

$$a_n = \frac{1}{\sqrt{2\lambda}}(2n + 1)J_{n+1/2}(\lambda\pi)\sin(\lambda\pi + n\pi/2).$$

Here, $J_{n+1/2}$ is the Bessel function of the first kind of order $n + 1/2$. If the approximation is given as a polynomial of degree k, then the truncation error $E(\lambda, k)$ is

$$|E(\lambda, k)| \leq \sum_{n=k+1}^{\infty} |a_n|.$$

Here, the coefficient a_n satisfies

$$|a_n| \leq \frac{1}{\sqrt{2\lambda}}(2n + 1)|J_{n+1/2}(\lambda\pi)|.$$

If N elements are used for the approximation, which means the wavenumber on each element is λ/N, then the new error is

$$|E(\lambda,k,N)| \leq \sum_{n=k+1}^{\infty} \frac{1}{\sqrt{2\lambda/N}}(2n+1)\left|J_{n+1/2}\left(\frac{\lambda\pi}{N}\right)\right|.$$

Using the asymptotic form for the Bessel function (as $x \to 0$) (Eq. (9.1.7) in [1])

$$J_v(x) \sim \frac{1}{\Gamma(v+1)}\left(\frac{x}{2}\right)^v,$$

we have

$$\begin{aligned}|a_n| &\leq \frac{1}{\sqrt{2\lambda/N}}(2n+1)\left(\frac{\lambda\pi}{2N}\right)^{n+1/2}\frac{1}{\Gamma(n+3/2)}\\ &= \frac{1}{\sqrt{2\lambda/N}}(2n+1)\left(\frac{\lambda\pi}{2N}\right)^{n+1/2}\frac{4^{n+1}(n+1)!}{\sqrt{\pi}(2n+2)!}\\ &= \frac{1}{(2n-1)!!}\left(\frac{\lambda\pi}{N}\right)^n\end{aligned}$$

Since $h = \frac{2}{N}$, we obtain the lowest order term of h for the truncation error,

$$|E(\lambda,k,N)| \leq \frac{(\lambda\pi)^{k+1}}{(2k+1)!!}h^{k+1}.$$

The required number of elements, $N \gg \lambda\pi$, is needed for this desired accuracy.

Then, we check the details of the condition $N \gg \lambda\pi$. This condition is required by using the asymptotic form for the Bessel function of the first kind that

$$J_v(z) \sim (\frac{1}{2}z)^v/\Gamma(v+1).$$

The above asymptotic form requires $z \to 0$ when v is fixed. In the above error estimate, $z = \frac{\lambda\pi}{N}$, then we get $N \gg \lambda\pi$. In order to understand further details of this requirement, we write the exact formula of the Bessel function of the first kind (Eq (9.1.10) in [1]),

$$J_v(z) = \left(\frac{z}{2}\right)^v \sum_{n=0}^{\infty} b_n(z) = \left(\frac{z}{2}\right)^v \sum_{n=0}^{\infty} \frac{\left(-\frac{z^2}{4}\right)^n}{n!\Gamma(v+n+1)}.$$

Comparing the first term $b_0(z)$ and the second $b_1(z)$ with $z = \frac{\lambda\pi}{N}$ and $v = k+3/2$,

$$|b_0(z)/b_1(z)| = \frac{4\Gamma(v+2)}{z^2\Gamma(v+1)} = \frac{4(v+1)}{z^2} = \frac{(4k+6)N^2}{(\lambda\pi)^2}.$$

Therefore, we can require $b_0(z)/b_1(z) \geq N$ to guarantee a reasonable accuracy and accuracy order. This means,

$$\frac{(4k+6)N^2}{(\lambda\pi)^2} \geq N \quad \Longrightarrow \quad N \geq \frac{(\lambda\pi)^2}{4k+6}.$$

The above formula gives the requirement of the number of elements for approximating a wave function. Furthermore, the divided differences of the approximations are also needed in the process of SIAC filtering. Consider the first divided difference of the wave function, $\sin(2\lambda\pi x)$,

$$\begin{aligned}\partial_h \sin(2\lambda\pi x) &= \frac{1}{h}\left(\sin(2\lambda\pi x + \lambda\pi h) - \sin(2\lambda\pi x - \lambda\pi h)\right)\\ &= \frac{2}{h}\sin(\lambda\pi h)\cos(2\lambda\pi x).\end{aligned}$$

According to the above formula, the wave number does not change for the divided differences. Therefore, we know that once the number of elements is sufficient for the DG approximation, then there is no problem with its divided differences.

6.3.2 SIAC Filtering for Wave Functions

Through the analysis in the previous section, we know that the theoretical foundations of using SIAC filters for wave functions are solidly established. However, things are not going as well as we expected.

Consider a linear hyperbolic equation

$$\begin{aligned} &u_t + u_x = 0, \quad (x,t) \in [0,1] \times (0,T]\\ &u(x,0) = \sin(2\lambda\pi x)\end{aligned} \tag{6.5}$$

with final time $T = 1$ over uniform meshes. In this example, we test $\lambda = 1, 2, \ldots 8$ cases. In order to save space, we only present the L^2 and L^∞ norm errors for $\lambda = 8$ case of equation (6.5) in Table 6.4. Here, the filter used is the standard symmetric filter given in (1.6) with scaling $H = h$ (h is the mesh size).

Table 6.4: L^2- and $L^\infty-$errors for the DG approximation u_h and the filtered solution $u_h^\star$ for a function that has 8 waves for a linear advection equation.

	DG error				After filtering			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\mathbb{P}^2$								
20	6.37E-01	–	9.41E-01	–	6.71E-01	–	9.96E-01	–
40	6.29E-02	3.34	1.04E-01	3.17	9.50E-02	2.82	1.87E-01	2.41
80	3.21E-03	4.29	1.08E-02	3.27	3.23E-03	4.88	6.24E-03	4.91
160	2.98E-04	3.43	1.53E-03	2.82	8.97E-05	5.17	1.59E-04	5.30
$\mathbb{P}^3$								
20	1.02E-01	–	1.82E-01	–	4.87E-01	–	9.34E-01	–
40	2.45E-03	5.38	1.07E-02	4.09	3.68E-02	3.72	9.42E-02	3.31
80	1.36E-04	4.17	6.73E-04	3.99	3.43E-04	6.75	9.21E-04	6.68
160	8.54E-06	4.00	4.16E-05	4.02	1.69E-06	7.66	4.63E-06	7.64

From Table 6.4, we can see that only if the mesh is sufficiently refined, $N = 80$ for $\mathbb{P}^2$ and $\mathbb{P}^3$, the filtered solution has better accuracy compared to the original DG solution.

Also, we point out that the improvement after filtering is not as impressive compared to the single wave example. To further investigate this problem, we present Figure 6.7 that shows the variation of the errors with the wave number λ. We can see that if the standard filter is used, when the wave number λ is large, the filtered solutions does not behave well, it can behave even worse than the original DG solution. In this section, we propose two possible ways to solve this problem. The first method is considering the idea of optimal scaling $H^\star$ introduced in Chapter 4. By using the optimal scaling $H^\star$, the filtered solutions are at least as good as the DG approximation. The second method is to use the compressed filter (6.7) introduced in the next section. One can use a compression factor $\eta = 1/\lambda$ by considering the concept of splines-per-wavelength, the same number of B-splines for each wavelength as the standard filter for the single wave function. The results of using two methods are shown in Figure 6.7. From Figure 6.7, the results suggest that both methods work well and better than the DG solution and using the standard filter.

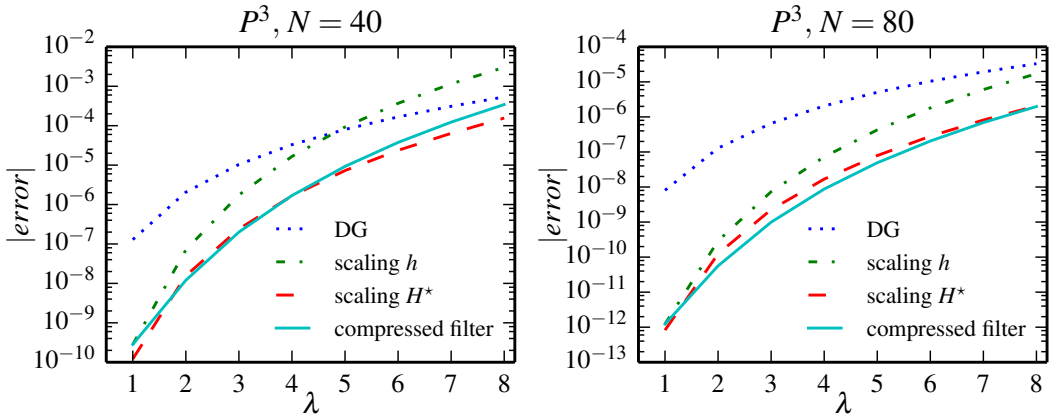


Figure 6.7: Point-wise errors in log scale of the DG solution and filtered solutions: the standard filter with scaling h, the standard filter with optimal scaling $H^\star$ and the compressed filter ($\eta = 1/\lambda$) with scaling h, for a function with wave number λ for a linear advection equation.

Remark 6.3.1. *We note that this section is a very preliminary study of SIAC filtering for wave functions. Further research is needed to develop a rigorous conclusion for this problem.*

6.4 Compressed SIAC Filter

During the previous discussions, we addressed different components of SIAC filters, such as the order of B-splines and the number of B-splines. In this section, we treat another basic component of the filter: the sampling of the B-splines.

Recall the formula of the symmetric filter,

$$K^{(2k+1,k+1)}(x) = \sum_{\gamma=0}^{r} c_\gamma^{(2k+1,k+1)} \psi^{(k+1)}(x + k - \gamma). \tag{6.6}$$

In formula (6.6), the B-splines are uniformly distributed with distance 1. However, the proof of Theorem 1.3.4 suggests that the sampling of the $2k+1$ B-splines has no effect on the final accuracy order, $2k+1$. The theoretical results allow us to sample in different ways. It is then important to sample the B-splines in a way that we can obtain additional benefits.

As mentioned earlier, one important computational factor of the filter is its support size. A large support size usually leads to many computational problems. Based on this, we can sample the B-splines more closely to each other and reduce the support size of the filter.

$$K_\eta^{(2k+1,k+1)}(x) = \sum_{\gamma=0}^{r} c_\gamma^{(2k+1,k+1)} \psi^{(k+1)}(x - x_\gamma), \tag{6.7}$$

where the sampling is

$$x_\gamma = \eta(-k + \gamma).$$

Here, we refer to the filter given in (6.7) as the compressed filter and η as the compression factor. Figure 6.8 shows the compressed filters with $\eta = 0.5$, see the original filters in Figure 1.2. The support size of the compressed filter is $(2\eta+1)k+1$.

Remark 6.4.1. *If one changes the scaling of the original filter in* (6.6) *by a scaling* $H = \frac{(2\eta+1)k+1}{3k+1}$, *then the scaled filter* $K_H^{(2k+1,k+1)}$ *has the same support size as the compressed filter* $K_\eta^{(2k+1,k+1)}$. *However, these two filters are not equivalent. Only the compressed filter is able to obtain supperconvergence.*

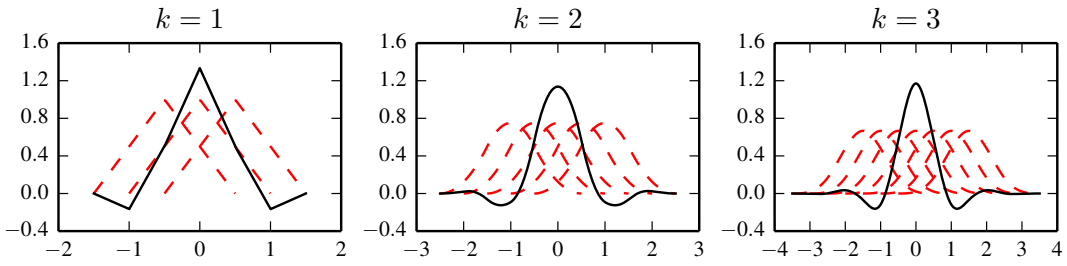


Figure 6.8: Solid black lines represent the compressed filter $K_\eta^{(2k+1,k+1)}$ given in (6.7) with $k = 1, 2, 3$, dashed red lines represent the respect central B-splines. The compression factor $\eta = 0.5$.

For the filtered solutions, we denote the scaled compressed filter as $K_{h,\eta}^{(2k+1,k)}(x) = \frac{1}{h} K_\eta^{(2k+1,k+1)}$, then one can prove

Theorem 6.4.1. *Under the same conditions in Theorem 1.3.4, the compressed filter defined in* (6.7) *with* $\eta > 0$*, then*

$$\|u - K_{h,\eta}^{(2k+1,k)} \star u_h\|_{0,\Omega_0} \leq Ch^{2k+1}.$$

Proof. The proof is the same as in Theorem 1.3.4. □

Example 6.4.2. *Consider a linear hyperbolic equation*

$$\begin{aligned} u_t + u_x &= 0, \quad (x,t) \in [0,1] \times (0,T] \\ u(x,0) &= \sin(2\pi x) \end{aligned}$$

with final time $T = 1$ *over uniform meshes. The* L^2 *and* L^∞ *norm errors and respective accuracy orders are given in Table 6.5. Figure 6.9 shows the point-wise errors in log scale. The respective results of the DG approximation and the original filtered solution can be found in Example 1.3.5.*

Table 6.5: L^2- and $L^\infty-$errors for the filtered solution $u_h^\star$ with compressed filters ($\eta = 0.5$ and $\eta = 0.25$) for a linear advection equation.

	$K_{0.5}^{(2k+1,k+1)}$				$K_{0.25}^{(2k+1,k+1)}$			
Mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
$\mathbb{P}^1$								
20	1.94E-03	–	2.78E-03	–	1.93E-03	–	2.78E-03	–
40	2.42E-04	3.00	3.44E-04	3.01	2.41E-04	3.00	3.44E-04	3.01
80	3.01E-05	3.00	4.27E-05	3.01	3.01E-05	3.00	4.27E-05	3.01
160	3.76E-06	3.00	5.32E-06	3.01	3.75E-06	3.00	5.32E-06	3.01
$\mathbb{P}^2$								
20	2.51E-06	–	3.66E-06	–	2.27E-06	–	3.47E-06	–
40	6.95E-08	5.17	1.00E-07	5.19	6.57E-08	5.11	9.72E-08	5.16
80	2.09E-09	5.06	2.98E-09	5.07	2.03E-09	5.02	2.94E-09	5.05
160	7.12E-11	4.87	1.01E-10	4.88	7.03E-11	4.85	1.01E-10	4.87
$\mathbb{P}^3$								
20	1.11E-08	–	1.57E-08	–	5.89E-09	–	9.15E-09	–
40	4.81E-11	7.84	6.86E-11	7.84	2.76E-11	7.74	4.25E-11	7.75
80	3.04E-13	7.30	4.33E-13	7.31	2.24E-13	6.95	3.30E-13	7.01
160	1.23E-14	4.63	1.74E-14	4.64	1.20E-14	4.23	1.70E-14	4.28

Through Example 6.4.2, we can see the compressed filter works as good as the original filter, same accuracy order, same error level and same smoothness. The compressed filter has reduced support size and maintains the same accuracy level as the original filter. The compressed filter idea can help solve many issues caused by the large support size of the filter, such as filtering over nonuniform meshes and boundaries. It is a subject of future work.

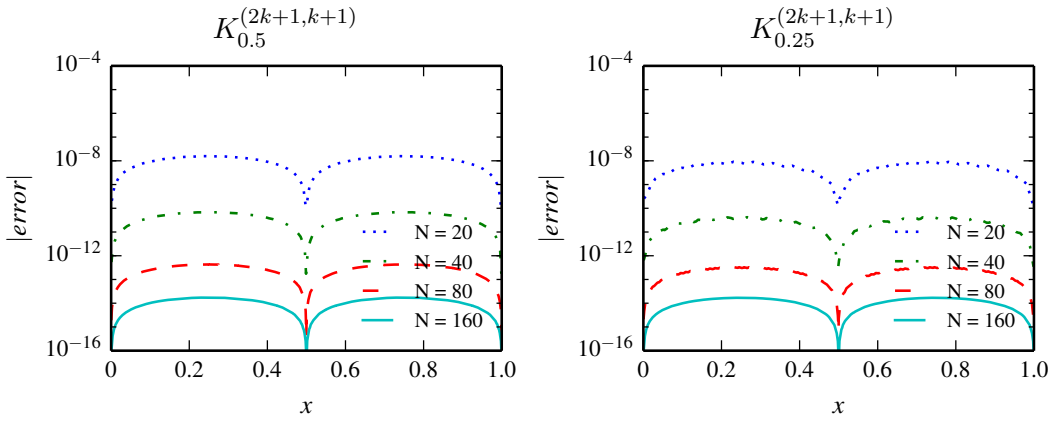


Figure 6.9: Point-wise errors in log scale of the filtered solutions with compressed filters ($\eta = 0.5$ and $\eta = 0.25$). The DG basis are polynomials $\mathbb{P}^3$ for a linear advection equation.

6.5 Conclusion

In this chapter, we have discussed several interesting topics of SIAC filters. Each topic can lead us to a deeper understanding of SIAC filters and can be further studied. Here, we just name a few future problems that are based on the topics in this chapter:

- Fast SIAC filtering for unstructured meshes and high-dimensional problems by combining the idea of the inexact Gaussian quadrature and the compressed filter.
- SIAC filtering for wave-related application problems by using the compressed filter or the optimal scaling concept.
- New structured filters based on Section 6.1, such as infinity smooth filter, trigonometric filter, etc.

7

Conclusion and Future Work

In this dissertation, we focus on exploiting superconvergence for discontinuous Galerkin solutions and constructing a superconvergence extraction techniques, in particular, Smoothness-Increasing Accuracy-Conserving filtering. We contributed to a series of studies of SIAC filters for a variety of circumstances by proving theoretical analysis and numerical experiments. The particular contributions of this thesis are the following:

- **One-Sided SIAC Filtering Over Uniform and Nonuniform Meshes.** Typically, most of the studies of SIAC filtering are confined to the interior of the underlying domain. For boundary regions, a one-sided filter is needed. The existing one-sided filters are not directly useful for most applications since they were limited to uniform meshes, linear equations, using multi-precision pages in computation. Also, the theoretical proof relied on a periodic boundary assumption. We aimed to overcome these deficiencies and develop a new fast one-sided filter for both uniform and nonuniform meshes. By studying B-splines and the negative order norm analysis, we generalized the structure of SIAC filters from a combination of central B-splines to using more general B-splines. Then, a "boundary shape" B-spline (using multiplicity knots at the boundary) was used to construct a new one-sided filter. We also presented the first theoretical proof of convergence for SIAC filtering over nonuniform meshes (smoothly-varying meshes). Details are given in Chapter 2.

- **Derivative Filtering Over Nonuniform Meshes and Near Boundaries.** One advantage of SIAC filtering is that it improves the smoothness of DG solutions. Because of the increased smoothness, we can obtain a better approximation of the derivatives of DG solutions. The derivative filtering over the interior region of uniform meshes was previously studied. However, nonuniform meshes and boundary regions still remain a big challenge. We extended the one-sided filter to a one-sided derivative filter. Nonuniform meshes are a difficult area, by investigating the negative order norm over arbitrary meshes, we proposed to scale the one-sided derivative filter with scaling h^{μ}. For arbitrary nonuniform rectangle meshes, we proved that the one-sided derivative filter can enhance the order of convergence for αth derivative of DG solution from $k+1-\alpha$ to $\mu(2k+2)$,

where $\mu \approx \frac{2}{3}$. Details are in Chapter 3.

- **Superconvergence Extraction Over Nonuniform Meshes.** The most challenging part of this project is recovering the superconvergence of a DG solution over nonuniform meshes through SIAC filtering. Typically, most theoretical proofs for the SIAC filter are limited to uniform meshes (or translation invariant meshes). The few theoretical investigations for nonuniform meshes were given in the one-sided and derivative filtering studies. Although our early research for nonuniform meshes was able to provide good engineering accuracy, we want to do better mathematically. This is not an easy task since unstructured meshes give DG solutions irregular performance under the negative order norm. In our work, we introduced a parameter to measure the "unstructuredness" of a given nonuniform mesh. Then by adjusting the scaling of SIAC filter based on this "unstructuredness" parameter, we are able to obtain the optimal filtered approximation (best accuracy) over a given nonuniform mesh. Details are in Chapter 4.

- **Application to Streamline Integration.** After introducing the new one-sided filter, we aimed to verify its usage in realistic engineering applications. The topic we chose was streamline integration. By taking advantage of the one-sided property of the new filter, we designed an efficient algorithm which filters the velocity field along the streamline, then uses a backward differentiation formula (BDF) for integration. Compared to the traditional method that filters the entire field (multi-dimensions algorithm), the computational cost drops dramatically since it is only a one-dimensional algorithm. Details can be found in Chapter 5.

- **Further Topics of SIAC Filters.** After studying SIAC filters for a broad range of applications, we returned to further investigations of SIAC filters themselves. Further topics such the uniqueness of the structure SIAC filters, the effects of the order of B-splines to SIAC filters and the compressed SIAC filters are included in Chapter 6. These topics give us in-depth insight into SIAC filters and reveal some future directions for the development of SIAC filters.

Future Work

- This dissertation studies the superconvergence and SIAC filters for DG methods. The extension is also possible to other numerical methods, such as finite element methods, finite volume methods, finite difference methods and spectral methods.

- The theoretical error estimates for SIAC filters are only established for linear hyperbolic problems. Nevertheless, the numerical results suggest that the filters work well for a broad range of problems, such as variable coefficient equations and linear hyperbolic conservation laws. Theoretical support for the nonlinear problems could be a significant step towards real-world applications.

- Uniform or nonuniform quadrilateral meshes were considered in this dissertation. The extension of SIAC filters for unstructured triangular or tetrahedral tessellation is still a problem that is not perfectly solved. Developing an efficient

and accurate way to use SIAC filters over these unstructured meshes has great practical values.

- The mathematical foundation of SIAC filters has been well established for many situations. It is time to evolve SIAC filters widely to real-world applications. For example, the use of SIAC filters for streamline integration is a subject of ongoing research.

Bibliography

[1] Milton Abramowitz and Irene A. Stegun, editors. *Handbook of mathematical functions with formulas, graphs, and mathematical tables*. Dover Publications, Inc., New York, 1992. Reprint of the 1972 edition. http://people.math.sfu.ca/∼cbm/aands/intro.htm.

[2] Slimane Adjerid and Mahboub Baccouch. The discontinuous Galerkin method for two-dimensional hyperbolic problems. II. A posteriori error estimation. *J. Sci. Comput.*, 38(1):15–49, 2009.

[3] Slimane Adjerid and Mahboub Baccouch. A superconvergent local discontinuous Galerkin method for elliptic problems. *J. Sci. Comput.*, 52(1):113–152, 2012.

[4] Slimane Adjerid, Karen D. Devine, Joseph E. Flaherty, and Lilia Krivodonova. A posteriori error estimation for discontinuous Galerkin solutions of hyperbolic problems. *Comput. Methods Appl. Mech. Engrg.*, 191(11-12):1097–1112, 2002.

[5] Slimane Adjerid and Idir Mechai. A superconvergent discontinuous Galerkin method for hyperbolic problems on tetrahedral meshes. *J. Sci. Comput.*, 58(1):203–248, 2014.

[6] Slimane Adjerid and Helmi Temimi. A discontinuous Galerkin method for the wave equation. *Comput. Methods Appl. Mech. Engrg.*, 200(5-8):837–849, 2011.

[7] Rick Archibald, Anne Gelb, Sigal Gottlieb, and Jennifer K. Ryan. One-sided post-processing for the discontinuous Galerkin method using ENO type stencil choosing and the local edge detection method. *J. Sci. Comput.*, 28(2-3):167–190, 2006.

[8] Mahboub Baccouch. The local discontinuous Galerkin method for the fourth-order Euler-Bernoulli partial differential equation in one space dimension. Part I: Superconvergence error analysis. *J. Sci. Comput.*, 59(3):795–840, 2014.

[9] Mahboub Baccouch. The local discontinuous Galerkin method for the fourth-order Euler-Bernoulli partial differential equation in one space dimension. Part II: *A posteriori* error estimation. *J. Sci. Comput.*, 60(1):1–34, 2014.

[10] Mahboub Baccouch. Superconvergence and *a posteriori* error estimates of a local discontinuous Galerkin method for the fourth-order initial-boundary value problems arising in beam theory. *Int. J. Numer. Anal. Model. Ser. B*, 5(3):188–216, 2014.

[11] Mahboub Baccouch and Slimane Adjerid. Discontinuous Galerkin error estimation for hyperbolic problems on unstructured triangular meshes. *Comput. Methods Appl. Mech. Engrg.*, 200(1-4):162–177, 2011.

[12] Georges-Pierre Bonneau, Thomas Ertl, and Gregory M. Nielson, editors. *Scientific visualization: the visual extraction of knowledge from data*. Mathematics and Visualization. Springer-Verlag, Berlin, 2006.

[13] J. H. Bramble and A. H. Schatz. Higher order local accuracy by averaging in the finite element method. *Math. Comp.*, 31(137):94–111, 1977.

[14] Susanne C. Brenner and L. Ridgway Scott. *The mathematical theory of finite element methods*, volume 15 of *Texts in Applied Mathematics*. Springer, New York, third edition, 2008.

[15] Waixiang Cao, Zhimin Zhang, and Qingsong Zou. Superconvergence of discontinuous Galerkin methods for linear hyperbolic equations. *SIAM J. Numer. Anal.*, 52(5):2555–2573, 2014.

[16] Fatih Celiker and Bernardo Cockburn. Element-by-element post-processing of discontinuous Galerkin methods for Timoshenko beams. *J. Sci. Comput.*, 27(1-3):177–187, 2006.

[17] Yingda Cheng and Chi-Wang Shu. Superconvergence and time evolution of discontinuous Galerkin finite element solutions. *J. Comput. Phys.*, 227(22):9612–9627, 2008.

[18] Yingda Cheng and Chi-Wang Shu. Superconvergence of discontinuous Galerkin and local discontinuous Galerkin schemes for linear hyperbolic and convection-diffusion equations in one space dimension. *SIAM J. Numer. Anal.*, 47(6):4044–4072, 2010.

[19] Philippe G. Ciarlet. *The finite element method for elliptic problems*, volume 40 of *Classics in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2002. Reprint of the 1978 original [North-Holland, Amsterdam; MR0520174 (58 #25001)].

[20] Bernardo Cockburn. An introduction to the discontinuous Galerkin method for convection-dominated problems. In *Advanced numerical approximation of nonlinear hyperbolic equations (Cetraro, 1997)*, volume 1697 of *Lecture Notes in Math.*, pages 151–268. Springer, Berlin, 1998.

[21] Bernardo Cockburn. Discontinuous Galerkin methods for convection-dominated problems. In *High-order methods for computational physics*, volume 9 of *Lect. Notes Comput. Sci. Eng.*, pages 69–224. Springer, Berlin, 1999.

[22] Bernardo Cockburn. Discontinuous Galerkin methods. *ZAMM Z. Angew. Math. Mech.*, 83(11):731–754, 2003.

[23] Bernardo Cockburn, George E. Karniadakis, and Chi-Wang Shu, editors. *Discontinuous Galerkin methods*, volume 11 of *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, Berlin, 2000. Theory, computation and applications, Papers from the 1st International Symposium held in Newport, RI, May 24–26, 1999.

[24] Bernardo Cockburn, George E Karniadakis, and Chi-Wang Shu. The development of discontinuous Galerkin methods. In *Discontinuous Galerkin methods (Newport, RI, 1999)*, pages 3–50. Springer, Berlin, 2000.

[25] Bernardo Cockburn, Mitchell Luskin, Chi-Wang Shu, and Endre Süli. Enhanced accuracy by post-processing for finite element methods for hyperbolic equations. *Math. Comp.*, 72(242):577–606, 2003.

[26] Bernardo Cockburn and Chi-Wang Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws. II. General framework. *Math. Comp.*, 52(186):411–435, 1989.

[27] Bernardo Cockburn and Chi-Wang Shu. The local discontinuous Galerkin method for time-dependent convection-diffusion systems. *SIAM J. Numer. Anal.*, 35(6):2440–2463 (electronic), 1998.

[28] Bernardo Cockburn and Chi-Wang Shu. The Runge-Kutta discontinuous Galerkin method for conservation laws. V. Multidimensional systems. *J. Comput. Phys.*, 141(2):199–224, 1998.

[29] Bernardo Cockburn and Chi-Wang Shu. Runge-Kutta discontinuous Galerkin methods for convection-dominated problems. *J. Sci. Comput.*, 16(3):173–261, 2001.

[30] Sean Curtis, Robert M. Kirby, Jennifer K. Ryan, and Chi-Wang Shu. Postprocessing for the discontinuous Galerkin method over nonuniform meshes. *SIAM J. Sci. Comput.*, 30(1):272–289, 2007/08.

[31] Carl de Boor. *A practical guide to splines*, volume 27 of *Applied Mathematical Sciences*. Springer-Verlag, New York, revised edition, 2001.

[32] Daniele Antonio Di Pietro and Alexandre Ern. *Mathematical aspects of discontinuous Galerkin methods*, volume 69 of *Mathématiques & Applications (Berlin) [Mathematics & Applications]*. Springer, Heidelberg, 2012.

[33] Jim Douglas, Jr. and Todd Dupont. Some superconvergence results for Galerkin methods for the approximate solution of two-point boundary problems. In *Topics*

in numerical analysis (Proc. Roy. Irish Acad. Conf., University Coll., Dublin, 1972), pages 89–92. Academic Press, London, 1973.

[34] Jim Douglas, Jr., Todd Dupont, and Mary Fanett Wheeler. Some superconvergence results for an H^1-Galerkin procedure for the heat equation. In *Computing methods in applied sciences and engineering (Proc. Internat. Sympos., Versailles, 1973), Part 1*, pages 288–311. Lecture Notes in Comput. Sci., Vol. 10. Springer, Berlin, 1974.

[35] Lawrence C. Evans. *Partial differential equations*, volume 19 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 1998.

[36] Xiaobing Feng, Ohannes Karakashian, and Yulong Xing, editors. *Recent developments in discontinuous Galerkin finite element methods for partial differential equations*, volume 157 of *The IMA Volumes in Mathematics and its Applications*. Springer, Cham, 2014. 2012 John H. Barrett Memorial Lectures, Selected papers from the workshop held at the University of Tennessee, Knoxville, TN, May 9–11, 2012.

[37] Casey Hufford and Yulong Xing. Superconvergence of the local discontinuous Galerkin method for the linearized Korteweg-de Vries equation. *J. Comput. Appl. Math.*, 255:441–455, 2014.

[38] Arieh Iserles. *A first course in the numerical analysis of differential equations*. Cambridge Texts in Applied Mathematics. Cambridge University Press, Cambridge, second edition, 2009.

[39] Liangyue Ji, Paulien van Slingerland, Jennifer K. Ryan, and Kees Vuik. Superconvergent error estimates for position-dependent smoothness-increasing accuracy-conserving (SIAC) post-processing of discontinuous Galerkin solutions. *Math. Comp.*, 83(289):2239–2262, 2014.

[40] Liangyue Ji, Yan Xu, and Jennifer K. Ryan. Accuracy-enhancement of discontinuous Galerkin solutions for convection-diffusion equations in multiple-dimensions. *Math. Comp.*, 81(280):1929–1950, 2012.

[41] Liangyue Ji, Yan Xu, and Jennifer K. Ryan. Negative-order norm estimates for nonlinear hyperbolic conservation laws. *J. Sci. Comput.*, 54(2-3):531–548, 2013.

[42] C. Johnson and J. Pitkäranta. An analysis of the discontinuous Galerkin method for a scalar hyperbolic equation. *Math. Comp.*, 46(173):1–26, 1986.

[43] James King, Hanieh Mirzaee, Jennifer K. Ryan, and Robert M. Kirby. Smoothness-increasing accuracy-conserving (SIAC) filtering for discontinuous Galerkin solutions: improved errors versus higher-order accuracy. *J. Sci. Comput.*, 53(1):129–149, 2012.

[44] Michal Křížek and Pekka Neittaanmäki. Bibliography on superconvergence. In *Finite element methods (Jyväskylä, 1997)*, volume 196 of *Lecture Notes in Pure and Appl. Math.*, pages 315–348. Dekker, New York, 1998.

[45] P. Lasaint and P.-A. Raviart. On a finite element method for solving the neutron transport equation. In *Mathematical aspects of finite elements in partial differential equations (Proc. Sympos., Math. Res. Center, Univ. Wisconsin, Madison, Wis., 1974)*, pages 89–123. Publication No. 33. Math. Res. Center, Univ. of Wisconsin-Madison, Academic Press, New York, 1974.

[46] Randall J. LeVeque. *Finite volume methods for hyperbolic problems.* Cambridge Texts in Applied Mathematics. Cambridge University Press, Cambridge, 2002.

[47] Xiaozhou Li, Yan Xu, and Yishen Li. Investigation of multi-soliton, multi-cuspon solutions to the Camassa-Holm equation and their interaction. *Chin. Ann. Math. Ser. B*, 33(2):225–246, 2012.

[48] Hanieh Mirzaee, James King, Jennifer K. Ryan, and Robert M. Kirby. Smoothness-increasing accuracy-conserving filters for discontinuous Galerkin solutions over unstructured triangular meshes. *SIAM J. Sci. Comput.*, 35(1):A212–A230, 2013.

[49] Hanieh Mirzaee, Jennifer K. Ryan, and Robert M. Kirby. Quantification of errors introduced in the numerical approximation and implementation of smoothness-increasing accuracy conserving (SIAC) filtering of discontinuous Galerkin (DG) fields. *J. Sci. Comput.*, 45(1-3):447–470, 2010.

[50] Hanieh Mirzaee, Jennifer K. Ryan, and Robert M. Kirby. Efficient implementation of smoothness-increasing accuracy-conserving (SIAC) filters for discontinuous Galerkin solutions. *J. Sci. Comput.*, 52(1):85–112, 2012.

[51] Michael S. Mock and Peter D. Lax. The computation of discontinuous solutions of linear hyperbolic equations. *Comm. Pure Appl. Math.*, 31(4):423–430, 1978.

[52] Kassem Mustapha and Jennifer K. Ryan. Post-processing discontinuous Galerkin solutions to Volterra integro-differential equations: analysis and simulations. *J. Comput. Appl. Math.*, 253:89–103, 2013.

[53] Todd E. Peterson. A note on the convergence of the discontinuous Galerkin method for a scalar hyperbolic equation. *SIAM J. Numer. Anal.*, 28(1):133–140, 1991.

[54] W.H. Reed and T.R. Hill. Triangular mesh methods for the neutron transport equation. *Los Alamos Report LA-UR-73-479*, 1973.

[55] Jennifer K. Ryan. Local Derivative Post-processing: Challenges for a non-uniform mesh. *Delft University of Technology Report 10-18*, 2013.

[56] Jennifer K. Ryan and Bernardo Cockburn. Local derivative post-processing for the discontinuous Galerkin method. *J. Comput. Phys.*, 228(23):8642–8664, 2009.

[57] Jennifer K. Ryan and Chi-Wang Shu. On a one-sided post-processing technique for the discontinuous Galerkin methods. *Methods Appl. Anal.*, 10(2):295–307, 2003.

[58] Jennifer K. Ryan, Chi-Wang Shu, and Harold Atkins. Extension of a postprocessing technique for the discontinuous Galerkin method for hyperbolic equations with application to an aeroacoustic problem. *SIAM J. Sci. Comput.*, 26(3):821–843, 2005.

[59] Alfred H. Schatz. Pointwise error estimates and asymptotic error expansion inequalities for the finite element method on irregular grids. I. Global estimates. *Math. Comp.*, 67(223):877–899, 1998.

[60] Larry L. Schumaker. *Spline functions: basic theory.* Cambridge Mathematical Library. Cambridge University Press, Cambridge, third edition, 2007.

[61] M Steffen, S Curtis, R M Kirby, and J K Ryan. Investigation of Smoothness-Increasing Accuracy-Conserving Filters for Improving Streamline Integration Through Discontinuous Fields. *Visualization and Computer Graphics, IEEE Transactions on*, 14(3):680–692, 2008.

[62] Vidar Thomée. High order local approximations to derivatives in the finite element method. *Math. Comp.*, 31(139):652–660, 1977.

[63] Vidar Thomée. Negative norm estimates and superconvergence in Galerkin methods for parabolic problems. *Math. Comp.*, 34(149):93–113, 1980.

[64] Paulien van Slingerland, Jennifer K. Ryan, and C Vuik. Smoothness-Increasing Convergence-Conserving Spline Filters Applied to Streamline Visualization of DG Approximations. *Delft University of Technology Report 09-06*, 2009. http://ta.twi.tudelft.nl/nw/users/vuik/papers/Sli09RV.pdf.

[65] Paulien van Slingerland, Jennifer K. Ryan, and C. Vuik. Position-dependent smoothness-increasing accuracy-conserving (SIAC) filtering for improving discontinuous Galerkin solutions. *SIAM J. Sci. Comput.*, 33(2):802–825, 2011.

[66] M.J. Vuik. Limiting and shock detection for discontinuous Galerkin solutions using multiwavelet. *TU Delft MSc Thesis*, 2012. http://ta.twi.tudelft.nl/users/vuik/numanal/vuik_afst.pdf.

[67] Lars B. Wahlbin. *Superconvergence in Galerkin finite element methods*, volume 1605 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 1995.

[68] David Walfisch, Jennifer K. Ryan, Robert M. Kirby, and Robert Haimes. One-sided smoothness-increasing accuracy-conserving filtering for enhanced streamline integration through discontinuous fields. *J. Sci. Comput.*, 38(2):164–184, 2009.

[69] Ziqing Xie and Zhimin Zhang. Uniform superconvergence analysis of the discontinuous Galerkin method for a singularly perturbed problem in 1-D. *Math. Comp.*, 79(269):35–45, 2010.

[70] Yan Xu and Chi-wang Shu. Local discontinuous Galerkin methods for three classes of nonlinear wave equations. *J. Comput. Math.*, 22(2):250–274, 2004. Special issue dedicated to the 70th birthday of Professor Zhong-Ci Shi.

[71] Yan Xu and Chi-Wang Shu. Local discontinuous Galerkin methods for nonlinear Schrödinger equations. *J. Comput. Phys.*, 205(1):72–97, 2005.

[72] Yan Xu and Chi-Wang Shu. Local discontinuous Galerkin methods for the Kuramoto-Sivashinsky equations and the Ito-type coupled KdV equations. *Comput. Methods Appl. Mech. Engrg.*, 195(25-28):3430–3447, 2006.

[73] Yan Xu and Chi-Wang Shu. A local discontinuous Galerkin method for the Camassa-Holm equation. *SIAM J. Numer. Anal.*, 46(4):1998–2021, 2008.

[74] Yang Yang and Chi-Wang Shu. Analysis of optimal superconvergence of discontinuous Galerkin method for linear hyperbolic equations. *SIAM J. Numer. Anal.*, 50(6):3110–3133, 2012.

[75] Yang Yang and Chi-Wang Shu. Discontinuous Galerkin method for hyperbolic equations involving δ-singularities: negative-order norm error estimates and applications. *Numer. Math.*, 124(4):753–781, 2013.

[76] Tie Zhang and Shun Yu. The derivative patch interpolation recovery technique and superconvergence for the discontinuous Galerkin method. *Appl. Numer. Math.*, 85:128–141, 2014.

[77] Zuozheng Zhang, Ziqing Xie, and Zhimin Zhang. Superconvergence of discontinuous Galerkin methods for convection-diffusion problems. *J. Sci. Comput.*, 41(1):70–93, 2009.

Curriculum vitae

Xiaozhou Li

- 2010.12 - 2015.7: Ph.D., Delft Institute of Applied Mathematics, Delft University of Technology, The Netherlands.
  - Supervisors: Dr. Jennifer Ryan and Prof. Kees Vuik
  - Thesis Title: Smoothness-Increasing Accuracy-Conserving Filters for Discontinuous Galerkin Methods: Challenging the Assumptions of Symmetry and Uniformity
- 2007 - 2010: B.S., Mathematics and Applied Mathematics, University of Science and Technology of China, China.
  - Supervisors: Prof. Qing Chen
  - Thesis Title: Geometric Measure Theory
- 2006 - 2007: School of Information Science and Technology, University of Science and Technoloty of China, China.
- Born on 24-05-1987 in Chongqing, China.
- See more on my website, http://xiaozhouli.com.

List of publications

Journal papers

- Smoothness-Increasing Accuracy-Conserving (SIAC) Filtering for Discontinuous Galerkin Solutions over Nonuniform Meshes: Superconvergence and Optimal Accuracy. X. Li, J.K. Ryan, R.M. Kirby, and C. Vuik. SIAM Journal on Scientific Computing, submitted.
- Smoothness-Increasing Accuracy-Conserving (SIAC) Filters for Derivative Approximations of DG solutions over Nonuniform Meshes and Near Boundaries. X. Li, J.K. Ryan, R.M. Kirby, and C. Vuik. Journal of Computational and Applied Mathematics, submitted.
- One-Sided Position-Dependent Smoothness-Increasing Accuracy-Conserving (SIAC) Filtering Over Uniform and Non-uniform Meshes. J.K. Ryan, X. Li, R.M. Kirby, and C. Vuik. Journal of Scientific Computing, accepted.

Others

- SIAC Filtering for Nonlinear Hyperbolic Equations. X. Li and J.K. Ryan. Proceedings of AMMCS 2013.
- One-Sided SIAC Filtering for Streamline with BDF Time Integrator. X. Li, J.K. Ryan, R.M. Kirby, and C. Vuik. Preprint.

Acknowledgements

At the end of my thesis, I would like to acknowledge the people who helped and supported me a lot in various aspects of my research and stayed in Delft.

First of all, my most sincere gratitude goes to my supervisor Dr. Jennifer Ryan. During my PhD, I have benefited greatly from her guidance and encouragement. Without her help, the research work would have never been finished. Beside research, she also helped me to adapt the life in the Netherlands and United Kingdom, and created the comfortable research environment. It was a great pleasure for me to do research under her supervision. I thank her for her patience and care that were of enormous importance to me.

I would also like to thank Prof. Kees Vuik for accepting me to work in the Group of Numerical Analysis at Delft Institute of Applied Mathematics, and for his wise suggestions during our meeting.

The present research was done cooperated with Prof. Mike Kirby at the University of Utah, Unite States. The cooperation was valuable for me and resulted in interesting research achievements.

My gratitude also goes to my committee, for taking their time to evaluate this thesis and providing helpful comments and suggestions.

I am also grateful to my first supervisor Prof. Yan Xu at the University of Science and Technology of China. She played an important role in my research in the primary stage, and without her my PhD in Delft would not be started.

I am thankful to Liangyue Ji, for having delightful discussions and offering useful suggestions. Also, I am grateful to members of our research group: Thea, Daniel, Julia and Xiong for their helping and discussions. Special thanks go to Thea Vuik, for helping me translate part of this thesis into Dutch.

I am thankful to Theda Olsder, who has managed with an amount of formal procedures related to my arrival and stay in Delft. Deborah Dongor, who has helped with official affairs. Kees Lemmens, who has provided technical support.

It was a pleasure to share the office with friendly office mates Pavel and Reijer, and then Guido and Joost, and then Jing, Thea and Peiyao. During my study in Delft, I met a lot of friendly and pleasant colleagues in Numerical Analysis group. Many

thanks to Fred, Duncan, Domenico, Matthias, Neil, Fons, Martin, etc. Moreover, I would also like to thank former and current PhD students Bowen, Menel, Manuel, Dennis, Daniël, Rohit, Martijn, Fahim, Abdul, Behrouz, Reinaldo, Fei, Yue, Fei, Lisa, Xin, Jiao, Gabriela, Luis, etc.

Finally, I would like to give my greatest gratitude to my parents.

Xiaozhou Li
Delft, June 2015