



Eidgenössische Technische Hochschule Zürich  
Swiss Federal Institute of Technology Zurich



# Implementation and numerical analysis of the hp discontinuous Galerkin FEM for convection-diffusion problems in one space dimension

Bachelor Thesis

written by

Tim Gyger

Department of Mathematics  
ETH Zürich

**Supervisor:**

Prof. Dr. Christoph Schwab

**Co-Supervisor:**

Fernando Henriquez Barraza

July 8, 2020

# Abstract

We analyze the theoretical and computational aspects of the hp-DG-FEM for the convection-diffusion problem in one space dimension introduced in [2] and [3]. In a last step, the convection-diffusion problem in one space dimension is extended to include random coefficients and we compute statistical moments of the solution.

# Contents

<b>Abstract</b>	<b>i</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 The Convection-Diffusion Problem</b>	<b>3</b>
2.1 The Problem Formulation . . . . .	3
2.2 Existence, Uniqueness and Regularity of the Solution . . . . .	4
<b>3 The hp discontinuous Galerkin Finite Element Method</b>	<b>14</b>
3.1 Finite Element Spaces for the DG-FEM . . . . .	14
3.2 Variational Formulation . . . . .	17
3.2.1 Galerkin Discretization . . . . .	17
3.3 Existence and Uniqueness of the DG-FE Solution . . . . .	19
<b>4 Stability and Convergence of the hp-DG-FEM</b>	<b>28</b>
4.1 Stability . . . . .	28
4.2 Convergence . . . . .	33
<b>5 Numerical Scheme and Implementations</b>	<b>36</b>
5.1 Numerical Scheme . . . . .	36
5.1.1 Computing the elementwise Matrices and Vectors . . . . .	37
5.1.2 Assembly of the global Stiffness Matrix and Load Vector . . . . .	42
5.1.3 Imposing the Boundary Conditions . . . . .	42
5.2 Implementations . . . . .	44
5.2.1 hp-DG-FEM Solution . . . . .	44
5.2.2 hp-DG-FEM with constant Coefficients and 2-element Mesh . . . . .	56
5.2.3 Error Estimators . . . . .	63

<b>6</b>	<b>Numerical Experiments</b>	<b>67</b>
6.1	The Model Problem . . . . .	67
6.2	Numerical Results . . . . .	68
<b>7</b>	<b>Stochastic Differential Equations and the hp-DG-FEM</b>	<b>71</b>
7.1	The Stochastic Problem Formulation . . . . .	72
7.2	Monte Carlo Method . . . . .	76
7.2.1	Convergence . . . . .	77
7.3	Random Number Generator . . . . .	78
7.3.1	Mathematical Formulation . . . . .	78
7.3.2	Basic Example . . . . .	79
7.4	Implementation . . . . .	79
7.5	Numerical Experiments . . . . .	84
7.5.1	The Model Problem . . . . .	84
7.5.2	Approximation of the Mean and the Variance Function . .	85
7.5.3	Convergence Analysis . . . . .	86
	<b>Bibliography</b>	<b>88</b>

# Introduction

---

The Finite Element Method (FEM) is a widely used tool for the numerical solution of Partial Differential Equations (PDEs). The idea behind the FEM is to divide the computational domain into smaller subdomains and, using this partition, construct a finite dimensional space in which a numerical approximation to the exact solution of the problem is computed. This procedure and the basic algorithmic concepts of the FEM are discussed in [1].

Different variants of the FEM may be found in the scientific literature, which are tailored to the particular challenges posed by specific PDEs. For example, in advection dominated equations it has been reported that the FEM with discontinuous basis functions provides good stability properties. Furthermore, the hp-version of the FEM, in which the refinement of the computational domain and the selection of polynomial degree is done simultaneously, yields exponential convergence rates provided the solution satisfies specific regularity estimates [5].

In [2] and [3], the convection-diffusion problem, which describe the flow of physical quantities in a system due to convection and diffusion, in one space dimension is considered and the hp-discontinuous Galerkin FEM (hp-DG-FEM) is analyzed. In this model, the main challenge is that the smoothness of the solution to this problem is not uniform in the diffusion parameter. Hence, if not appropriately addressed, small values of this parameter may deteriorate the performance of the FEM. Previous to [2, 3], J.M. Melenk and Ch. Schwab [6] analyzed the regularity of the solution to convection-diffusion problems in one space dimension, with an explicit dependence on the diffusion parameter. Moreover, it was proved that hp-FEM with a carefully designed set of test functions and appropriate stabilization parameters deliver robust, i.e. independent of the diffusion parameter, exponential convergence rates. In [2, 3], the hp-DG-FEM has been shown to deliver robust exponential convergence rates in the approximation of the solution to the convection-diffusion problems in one space dimension, however, without any further stabilization.

In this work, we present in the second chapter the convection-diffusion equation on the basis of a model problem and prove the existence, uniqueness and regularity of its solution making certain assumptions about the coefficients. Chapter 3 introduces the hp discontinuous Galerkin Finite Element Method by stating the variational formulation of the model problem and by proving the existence and uniqueness of the DG-FE solution. In chapter 4, we analyse the stability and convergence of the hp-DG-FEM and prove the main statement in [2], the robust exponential convergence. In section 5, we present the numerical scheme of the hp-DG-FEM with the corresponding implementations in MATLAB which we use in chapter 6 to perform some numerical experiments illustrating the convergence results. In the last chapter 7, we extend the convection-diffusion model problem to include one random coefficient and compute statistical moments of the solution, such as the “ensemble average”, by using the Monte Carlo method.

# The Convection-Diffusion Problem

---

In this chapter we present a classical convection-diffusion equation in one space dimension and prove some regularity properties of its solution.

This chapter is mainly based on the detailed explanations in [7].

The behavior of many parameters in flow phenomena follows the convection-diffusion equation, such as momentum and heat. The convection-diffusion equation is also used to describe the diffusion process in environmental science, such as the pollutant transport in the atmosphere, oceans, lakes, rivers or groundwater. Therefore, the research of this problem is of great importance.

## 2.1 The Problem Formulation

We consider the following model problem which is also stated in [2]:

Find a function

$$u : \Omega = (-1, 1) \longrightarrow \mathbb{R} \quad (2.1)$$

such that

$$\begin{aligned} \mathcal{L}_\epsilon u &\equiv -\epsilon u'' + a(x)u' + b(x)u = f(x) \quad \text{in } \Omega \\ u(\pm 1) &= g^\pm. \end{aligned} \quad (2.2)$$

Here,  $\mathcal{L}_\epsilon$  is called the corresponding linear operator,  $0 < \epsilon \ll 1$  is the diffusion coefficient which represents for example the mass diffusivity for particle motion or the thermal diffusivity for heat transport,  $u(x)$  is, for example, the concentration of a substance for mass transfer or the temperature for heat transfer,  $a \in C^\infty(\bar{\Omega})$  is the velocity that the quantity  $u$  is moving with,  $b \in C^\infty(\bar{\Omega})$  describes losses respectively sources of the quantity and  $f \in C^\infty(\bar{\Omega})$  is an external source term.

Without loss of generality, we analyse the problem (2.2) with homogeneous Dirichlet boundary conditions:

$$g^\pm = 0.$$

In this work, we make following assumptions for the functions and coefficient as in [2] which we use in later proofs concerning the solution of the problem and the stability and convergence of the method:

There are constants  $b_0 \in \mathbb{R}$  and  $a_0, c_0, \bar{c} > 0$  such that for all  $x \in \Omega$  and all  $\epsilon \in (0, 1]$ :

$$a(x) \geq a_0 \tag{2.3}$$

$$b(x) \geq b_0 \tag{2.4}$$

$$\bar{c} \geq b(x) - \frac{1}{2}a'(x) \geq c_0 \tag{2.5}$$

$$a_0^2 + 4\epsilon b_0 > 0 \tag{2.6}$$

Furthermore, there are constants  $C_a, C_b, C_f, \gamma_a, \gamma_b, \gamma_f > 0$  independent of  $\epsilon$  and  $n$  such that for all  $n \in \mathbb{N}_0$  the following analytic regularity estimates are satisfied:

$$\|a^{(n)}\|_{L^\infty(\Omega)} \leq C_a \gamma_a^n n! \tag{2.7}$$

$$\|b^{(n)}\|_{L^\infty(\Omega)} \leq C_b \gamma_b^n n! \tag{2.8}$$

$$\|f^{(n)}\|_{L^\infty(\Omega)} \leq C_f \gamma_f^n n!. \tag{2.9}$$

## 2.2 Existence, Uniqueness and Regularity of the Solution

In this section, we show that under the assumptions made above (2.3) - (2.9) our model problem (2.2) has an unique solution and some regularity properties.

We need these properties in the end to proof the robust exponential convergence of the hp-DG-FEM.

Since the functions  $a, b$  and  $f$  are smooth on  $\bar{\Omega}$ , and since  $\epsilon > 0$ , we know by the extension of the Picard-Lindelöf theorem to second order differential equation that there exists an unique analytic solution  $u_\epsilon$  for the initial value problem (2.2).

If we choose a very small  $\epsilon$  the model problem characterizes a situation where the diffusion is dominated by the convection and the analytic solution has a so called boundary layer at  $x = 1$ . This phenomenon occurs in narrow regions near the boundary on which there is a rapid change in the solution to the boundary condition. This behavior can be described by asymptotic expansions which approximate a function in terms of a sequence of functions.



To this effect, we decompose the analytic solution as in [7]:

$$u_\epsilon = w_m + C_m u_\epsilon^+ + r_m, \quad \text{for some } m \in \mathbb{N}_0. \quad (2.10)$$

Here,  $m$  is an arbitrary expansion order,  $w_m$  is the smooth asymptotic part:

$$w_m := \sum_{j=0}^m \epsilon^j u_j, \quad (2.11)$$

where

$$u_{j+1}(x) := e^{-\Lambda(x)} \int_{-1}^x \frac{e^{\Lambda(t)}}{a(t)} u_j''(t) dt \quad \text{for } j = 0, \dots, m-1 \quad (2.12)$$

$$u_0(x) := e^{-\Lambda(x)} \int_{-1}^x \frac{e^{\Lambda(t)}}{a(t)} f(t) dt \quad (2.13)$$

$$\Lambda(x) := \int_{-1}^x \lambda(t) dt \quad (2.14)$$

$$\lambda(x) := \frac{b(x)}{a(x)}, \quad (2.15)$$

$u^+$  is the outflow boundary part and solves the problem:

$$\mathcal{L}_\epsilon u_\epsilon^+ = 0 \quad \text{on } \Omega, \quad u_\epsilon^+(-1) = 0, \quad u_\epsilon^+(1) = 1, \quad (2.16)$$

$C_m$  is just the negative, asymptotic part evaluated at the boundary:

$$C_m := -w_m(1) \quad (2.17)$$

and the left over remainder  $r_m$  is the solution of the problem:

$$\mathcal{L}_\epsilon r_m = \epsilon^{m+1} u_m'' \quad \text{on } \Omega, \quad r_m(\pm 1) = 0. \quad (2.18)$$

Under this decomposition (2.10) we can prove some regularity properties of the analytic solution  $u_\epsilon$  of our model problem (2.2).

For this purpose, we proceed as in [7] and state firstly three auxiliary lemmas.

To do so, we define some constants and auxiliary functions:

$$\begin{aligned}\lambda^- &:= \max\left\{\frac{a_0 - \sqrt{a_0^2 + 4b_0\epsilon}}{2\epsilon}, 0\right\} \\ \lambda^+ &:= \min\left\{\frac{a_0 + \sqrt{a_0^2 + 4b_0\epsilon}}{2\epsilon}, \frac{a_0}{\epsilon}\right\} \\ \theta_\pm(x) &:= \frac{(x+1)}{\sqrt{a_0^2 + 4b_0\epsilon}} e^{\lambda^-(x+1)} \|f\|_{L^\infty(\bar{\Omega})} \pm u_\epsilon(x) \\ \phi_\pm(x) &:= u_\epsilon^+(1) e^{-\lambda^+(1-x)} \pm u_\epsilon^+(x).\end{aligned}$$

**Lemma 2.1** (First auxiliary lemma). *There exists a constant  $C > 0$  depending on the constants  $C_a, C_b, C_f, \gamma_a$  and  $\gamma_b$  from (2.7) - (2.9) and  $a_0$  and  $b_0$  from (2.3) and (2.4), such that the analytic solution  $u_\epsilon$  of (2.2) satisfies:*

$$\|u_\epsilon\|_{L^\infty(\Omega)} \leq C \quad (2.19)$$

$$\|u'_\epsilon\|_{L^\infty(\Omega)} \leq \frac{C}{\epsilon} \quad (2.20)$$

*Proof.* We use the maximum principle in differential equations which is a generalization of the elementary fact of calculus that any function  $f(x)$  which satisfies the inequality  $f'' > 0$  on an interval  $[a, b]$  achieves its maximum value at one of the endpoints of the interval. The theory of the maximum principle in second order differential equations and its use of approximating solutions is explained detailed in [8] (Chap. 1, Sec. 4 & 5).

It holds on the boundary of  $\Omega$ :

$$\theta_\pm(\pm 1) = \frac{(\pm 1 + 1)}{\sqrt{a_0^2 + 4b_0\epsilon}} e^{\lambda^-(\pm 1 + 1)} \|f\|_{L^\infty(\bar{\Omega})} \geq 0, \quad (2.21)$$

because of the non-negativity of the norm  $\|\cdot\|_{L^\infty}$  and since  $u_\epsilon(\pm 1)$  vanishes due to the boundary conditions (2.2)<sub>2</sub>.

For the differential equation defined by the linear operator  $\mathcal{L}_\epsilon$ , it holds for  $x \in (-1, 1)$ :

$$\begin{aligned}
\mathcal{L}_\epsilon \theta_\pm &= \mathcal{L}_\epsilon \left( \frac{(x+1)}{\sqrt{a_0^2 + 4b_0\epsilon}} e^{\lambda^-(x+1)} \|f\|_{L^\infty(\bar{\Omega})} \right) \pm \underbrace{\mathcal{L}_\epsilon(u_\epsilon(x))}_{=f} \\
&\stackrel{(2.2)}{\geq} \frac{1}{\sqrt{a_0^2 + 4b_0\epsilon}} e^{\lambda^-(x+1)} \|f\|_{L^\infty(\bar{\Omega})} (-\epsilon\lambda^-(\lambda^-x + \lambda^- + 2) \\
&\quad + a(x)(\lambda^-x + \lambda^- + 1) + b(x)(x+1)) - \|f\|_{L^\infty(\bar{\Omega})} \\
&\stackrel{(2.3),(2.4)}{\geq} \frac{1}{\sqrt{a_0^2 + 4b_0\epsilon}} e^{\lambda^-(x+1)} \|f\|_{L^\infty(\bar{\Omega})} (-\epsilon\lambda^-(\lambda^-x + \lambda^- + 2) \\
&\quad + a_0(\lambda^-x + \lambda^- + 1) + b_0(x+1)) - \|f\|_{L^\infty(\bar{\Omega})} \\
&\geq \frac{1}{\sqrt{a_0^2 + 4b_0\epsilon}} e^{\lambda^-(x+1)} \|f\|_{L^\infty(\bar{\Omega})} \underbrace{(x(-\epsilon(\lambda^-)^2 + b_0 + a_0\lambda^-))}_{=0} \\
&\quad \underbrace{-\epsilon(\lambda^-)^2 + b_0 + a_0\lambda^-}_{=0} + \underbrace{a_0 - 2\epsilon\lambda^-}_{=\sqrt{a_0^2 + 4b_0\epsilon}} - \|f\|_{L^\infty(\bar{\Omega})} \\
&\geq \underbrace{(e^{\lambda^-(x+1)} - 1)}_{\geq 1} \|f\|_{L^\infty(\bar{\Omega})} \geq 0.
\end{aligned}$$

Therefore, it holds by the maximum principle that:

$$|u_\epsilon(x)| \leq \frac{(x+1)}{\sqrt{a_0^2 + 4b_0\epsilon}} e^{\lambda^-(x+1)} \|f\|_{L^\infty(\bar{\Omega})} \stackrel{x \in [-1, 1]}{\leq} \frac{2}{\sqrt{a_0^2 + 4b_0\epsilon}} e^{2\lambda^-} \|f\|_{L^\infty(\bar{\Omega})}$$

and as claimed in (2.19) there exists a constant  $C > 0$  depending on the desired constants in (2.3) - (2.9):

$$\|u_\epsilon\|_{L^\infty(\Omega)} \leq C.$$

For the second estimate (2.20), we define the function:

$$A(x) := \frac{1}{\epsilon} \int_x^1 a(t) dt$$

with derivative:

$$A'(x) = -\frac{a(x)}{\epsilon}.$$

Firstly, we multiply the differential equation (2.2)<sub>1</sub> by  $e^{A(x)}$  and integrate from  $x$  to 1:

$$\underbrace{-\epsilon \int_x^1 u''_\epsilon e^{A(t)} dt + \int_x^1 a(t) u'_\epsilon e^{A(t)} dt + \int_x^1 b(t) u_\epsilon e^{A(t)} dt}_{=-\epsilon([u'_\epsilon(t)e^{A(t)}]_x^1 - \int_x^1 u'_\epsilon A'(t)e^{A(t)} dt) = \epsilon(u'_\epsilon(x)e^{A(x)} - u'_\epsilon(1)) - \int_x^1 a(t) u'_\epsilon e^{A(t)} dt} = \int_x^1 f(t) e^{A(t)} dt.$$

Integrating the first term by parts yields:

$$\epsilon(u'_\epsilon(x)e^{A(x)} - u'_\epsilon(1)) + \int_x^1 b(t) u_\epsilon e^{A(t)} dt = \int_x^1 f(t) e^{A(t)} dt.$$

We multiply again, but now by  $e^{-A(x)}$ , then we divide by  $\epsilon$  and rearrange:

$$u'_\epsilon(x) = u'_\epsilon(1)e^{-A(x)} - \frac{1}{\epsilon} \int_x^1 b(t) u_\epsilon e^{A(t)-A(x)} dt + \frac{1}{\epsilon} \int_x^1 f(t) e^{A(t)-A(x)} dt \quad (2.22)$$

and integrate this equation from  $-1$  to  $1$ :

$$0 = u'_\epsilon(1) \int_{-1}^1 e^{-A(x)} dx - \frac{1}{\epsilon} \int_{-1}^1 \int_x^1 e^{A(t)-A(x)} (b(t) u_\epsilon + f(t)) dt dx. \quad (2.23)$$

We estimate the terms:

$$\begin{aligned} \int_{-1}^1 e^{-A(x)} dx &= \int_{-1}^1 e^{-\frac{1}{\epsilon} \int_x^1 a(t) dt} dx \leq \underbrace{\int_{-1}^1 e^{-\frac{1}{\epsilon} \int_x^1 a_0 dt} dx}_{= \frac{\epsilon}{a_0} (1 - e^{-2\frac{a_0}{\epsilon}})} \leq \frac{\epsilon}{a_0} \\ \int_{-1}^1 e^{-A(x)} dx &= \int_{-1}^1 e^{-\frac{1}{\epsilon} \int_x^1 a(t) dt} dx \geq \underbrace{\int_{-1}^1 e^{-\frac{1}{\epsilon} \int_x^1 \|a\|_{L^\infty} dt} dx}_{= \frac{\epsilon}{\|a\|_{L^\infty}} (1 - e^{-2\frac{\|a\|_{L^\infty}}{\epsilon}})} \geq \frac{\epsilon}{\|a\|_{L^\infty}} (1 - e^{-2\frac{a_0}{\epsilon}}) \\ \frac{1}{\epsilon} \int_{-1}^1 \int_x^1 e^{A(t)-A(x)} dt dx &= \frac{1}{\epsilon} \int_{-1}^1 e^{-A(x)} \int_x^1 e^{A(t)} dt dx \leq \frac{2}{a_0}. \end{aligned}$$

Therefore, we can estimate the term  $u'_\epsilon(1)$  in (2.23):

$$|u'_\epsilon(1)| \leq \frac{2\|a\|_{L^\infty}}{\epsilon a_0} (1 - e^{-2\frac{a_0}{\epsilon}})^{-1} (\|b\|_{L^\infty} \|u_\epsilon\|_{L^\infty} + \|f\|_{L^\infty})$$

and insert this inequality in (2.22):

$$\begin{aligned}
|u'_\epsilon(x)| &\leq |u'_\epsilon(1)| \underbrace{|e^{-A(x)}|}_{\leq 1} + \frac{1}{\epsilon} \|b\|_{L^\infty} \|u_\epsilon\|_{L^\infty} \underbrace{\left| \int_x^1 e^{A(t)-A(x)} dt \right|}_{\leq 2} \\
&\quad + \frac{1}{\epsilon} \|f\|_{L^\infty} \underbrace{\left| \int_x^1 e^{A(t)-A(x)} dt \right|}_{\leq 2} \\
&\leq |u'_\epsilon(1)| + \frac{2}{\epsilon} (\|b\|_{L^\infty} \|u_\epsilon\|_{L^\infty} + \|f\|_{L^\infty}) \\
&\leq \frac{2}{\epsilon} \left( 1 + \frac{\|a\|_{L^\infty}}{a_0} (1 - e^{-2\frac{a_0}{\epsilon}})^{-1} \right) (\|b\|_{L^\infty} \|u_\epsilon\|_{L^\infty} + \|f\|_{L^\infty})
\end{aligned}$$

Hence, we can estimate for a constant  $C \geq 0$  as desired:

$$\begin{aligned}
\|u_\epsilon\|_{L^\infty(\Omega)} &\leq C \\
\|u'_\epsilon\|_{L^\infty(\Omega)} &\leq \frac{C}{\epsilon}
\end{aligned}$$

and the lemma yields.  $\square$

**Lemma 2.2** (Second auxiliary lemma). *Let  $u_\epsilon^+$  be the outflow boundary layer defined in (2.16). Then there exists a constant  $C > 0$  depending on  $C_a$ ,  $C_b$ ,  $\gamma_a$  and  $\gamma_b$  from (2.7) - (2.8) and  $a_0$  and  $b_0$  from (2.3) and (2.4) such that:*

$$|u_\epsilon^+(x)| \leq e^{-\frac{a_0}{2\epsilon}(1-x)} \quad (2.24)$$

$$|u_\epsilon^{'+}(x)| \leq \frac{C}{\epsilon} e^{-\frac{a_0}{2\epsilon}(1-x)} \quad (2.25)$$

*Proof.* We can proceed similarly to the proof of Lemma 2.1 with the auxiliary function  $\phi_\pm$  instead of  $\theta_\pm$  and get by the maximum principle:

$$|u_\epsilon^+(x)| \leq \underbrace{|u_\epsilon^+(1)|}_{\stackrel{(1.16)}{=} 1} e^{-\lambda^+(1-x)} \leq e^{-\frac{a_0}{2\epsilon}(1-x)}$$

and as for the derivative estimate in the proof of Lemma 2.1, it holds for a constant  $C > 0$ :

$$|u_\epsilon^{'+}(x)| \leq \frac{2}{\epsilon} \left( 1 + \frac{\|a\|_{L^\infty}}{a_0} (1 - e^{-2\frac{a_0}{\epsilon}})^{-1} \right) \|b\|_{L^\infty} \|u_\epsilon^+\|_{L^\infty} \leq \frac{C}{\epsilon} e^{-\frac{a_0}{2\epsilon}(1-x)}$$

and the lemma yields.  $\square$

**Lemma 2.3** (Third auxiliary lemma). *Let  $G$  be an open, complex neighborhood of  $[-1,1]$ . Assume that the functions  $\Lambda, a, u_0 : G \rightarrow \mathbb{C}$  are holomorphic (i.e. at every point of  $[-1,1]$ , complex differentiable in a neighbourhood of the point) and bounded on  $G$ . Assume additionally that  $|a| \geq a_0 > 0$  on  $G$ . Then there are constants  $C, K_1, K_2 > 0$  depending only on  $a_0, \|a'\|_{L^\infty(G)}, \|\Lambda\|_{L^\infty(G)}, \|\Lambda'\|_{L^\infty(G)}$  and  $G$  such that the functions  $u_j$  defined in (2.12)-(2.15) satisfy*

$$\|u_j^{(n)}\|_{L^\infty([-1,1])} \leq CK_1^j K_2^n j! n! \|u_0\|_{L^\infty(G)}, \quad \forall j, n \in \mathbb{N}_0$$

*Proof.* Without loss of generality we assume that  $G$  is star shaped with respect to  $z = -1$ . For  $\delta > 0$  we define  $G_\delta := \{z \in G \mid \text{dist}(z, \partial G) > \delta\}$ .

Claim:  $\exists C, K > 0 : \|u_j\|_{L^\infty(G_\delta)} \leq CK^j \delta^{-j} j! \|u_0\|_{L^\infty(G)}, \quad \forall j \in \mathbb{N}_0$  The proof of Lemma 2.3 follows immediately from this inequality and the Cauchy's integral theorem for derivatives.

**Cauchy's Integral Formula for Derivatives:** Let  $A \in \mathbb{C}$  be open and let  $f : A \rightarrow \mathbb{C}$  be analytic on  $A$ . Let  $\gamma$  be any simple closed, piecewise smooth and positively oriented curve contained in  $A$  and such that the inside of  $\gamma$  is contained in  $A$ . Then if  $z_0$  is inside  $\gamma$  we have that:

$$f^{(k)}(z_0) = \frac{k!}{2\pi i} \int_\gamma \frac{f(z)}{(z - z_0)^{(k+1)}} dz, \quad k \in \mathbb{N}_0$$

We proof the claim by induction over  $j$ . Obviously, the estimation is true for  $j = 0$  and  $C \geq 1$ .

We use our recursion in (2.12) and get by integrating by parts:

$$\begin{aligned} u_{j+1}(z) &= e^{-\Lambda(z)} \int_{-1}^z e^{\Lambda(t)} \frac{u_j''(t)}{a(t)} dt \\ &= e^{-\Lambda(z)} \left[ e^{\Lambda(t)} \frac{u_j'(t)}{a(t)} \right]_{-1}^z - e^{-\Lambda(z)} \int_{-1}^z e^{\Lambda(t)} \frac{\Lambda'(t)a(t) + a'(t)}{a(t)^2} u_j'(t) dt \\ &= \frac{u_j'(z)}{a(z)} - e^{-\Lambda(z)} \frac{u_j'(-1)}{a(-1)} - e^{-\Lambda(z)} \int_{-1}^z e^{\Lambda(t)} \frac{\Lambda'(t)a(t) + a'(t)}{a(t)^2} u_j'(t) dt \end{aligned}$$

and by tacking the absolute value we can estimate:

$$\begin{aligned} |u_{j+1}(z)| &\leq \frac{\|u_j'\|_{L^\infty(G_\delta)}}{a_0} (1 + |e^{-\Lambda(z)}|) \\ &\quad + \underbrace{|(z+1)|}_{< \infty \text{ since } G_\delta \text{ is bounded}} \frac{\|\Lambda'\|_{L^\infty(G_\delta)} \|a\|_{L^\infty(G_\delta)} + \|a'\|_{L^\infty(G_\delta)}}{a_0^2} \|u_j'\|_{L^\infty(G_\delta)}. \end{aligned}$$

Therefore, it follows that  $\exists C_1 > 0$  such that:

$$\|u_{j+1}\|_{L^\infty(G_\delta)} \leq C_1 \|u_j'\|_{L^\infty(G_\delta)}.$$

By Cauchy's integral theorem with  $A = G_{(1-k)\delta}$ ,  $\gamma(t) = z_0 + k\delta e^{2\pi it}$  for  $0 < k < 1$  and the induction hypothesis, we can estimate:

$$\begin{aligned}
\|u_{j+1}\|_{L^\infty(G_\delta)} &\leq C_1 \|u'_j\|_{L^\infty(G_\delta)} \leq C_1 \left\| \frac{1}{2\pi i} \int_\gamma \frac{u_j(z)}{(z - z_0)^2} dz \right\|_{L^\infty(G_\delta)} \\
&\leq C_1 \|u_j\|_{L^\infty(G_{(1-k)\delta})} \left| \frac{1}{2\pi i} \int_\gamma \frac{1}{(z - z_0)^2} dz \right| \\
&= C_1 \|u_j\|_{L^\infty(G_{(1-k)\delta})} \left| \frac{1}{2\pi i} \int_0^1 \frac{1}{(\gamma(t) - z_0)^2} \dot{\gamma}(t) dt \right| \\
&= C_1 \|u_j\|_{L^\infty(G_{(1-k)\delta})} \left| \frac{1}{2\pi i} \int_0^1 \frac{e^{-4\pi it}}{(k\delta)^2} 2\pi i k \delta e^{2\pi it} dt \right| \\
&= C_1 \frac{k\delta}{(k\delta)^2} \|u_j\|_{L^\infty(G_{(1-k)\delta})} \underbrace{\left| \int_0^1 e^{-2\pi it} dt \right|}_{\leq 1} \\
&\leq C_1 \frac{1}{k\delta} \|u_j\|_{L^\infty(G_{(1-k)\delta})} \\
&\leq C_1 C j! K^j (1-k)^{-j} \delta^{-j} \frac{1}{k\delta} \|u_0\|_{L^\infty(G)} \\
&\leq C(j+1)! K^{j+1} \delta^{-(j+1)} \|u_0\|_{L^\infty(G)} \frac{C_1}{K(j+1)(1-k)^j k} \\
&\stackrel{k=\frac{1}{(j+1)}}{\leq} C(j+1)! K^{j+1} \delta^{-(j+1)} \|u_0\|_{L^\infty(G)} \underbrace{\frac{C_1(j+1)^j}{K(j)^j}}_{\leq 1, \text{ for suitable } K} \\
&\leq C(j+1)! K^{j+1} \delta^{-(j+1)} \|u_0\|_{L^\infty(G)}
\end{aligned}$$

Therefore, we proved the claim and by Cauchy's integral theorem for derivatives also the lemma.  $\square$

With these auxiliary lemmas we are capable of proving the following regularity properties of the analytic solution as in [7]:

**Theorem 2.4.** (*Regularity*). *Let  $u_\epsilon$  be the solution of (2.2). Then there exist constants  $C, K > 0$  depending on the constants on  $C_a, C_b, C_f, \gamma_a, \gamma_b$  and  $\gamma_f$  from (2.7) - (2.9) and  $a_0$  and  $b_0$  from (2.3) and (2.4) such that:*

$$\|u_\epsilon^{(n)}\|_{L^\infty(\Omega)} \leq CK^n \max(n, \epsilon^{-1})^n, \quad \forall n \in \mathbb{N}_0 \quad (2.26)$$

$$|u_\epsilon^{+(n)}(x)| \leq CK^n \max(n, \epsilon^{-1})^n e^{-a_0(1-x)/(2\epsilon)}, \quad \forall n \in \mathbb{N}_0, \forall x \in \bar{\Omega} \quad (2.27)$$

Furthermore, under the assumption  $0 < \epsilon m K \leq 1$ , the terms in the decomposition (1.10) satisfy:

$$\|w_m^{(n)}\|_{L^\infty(\Omega)} \leq C K^n n!, \quad \forall n \in \mathbb{N}_0 \quad (2.28)$$

$$\|r_m^{(n)}\|_{L^\infty(\Omega)} \leq C \epsilon^{1-n} (\epsilon m K)^m, \quad n = 0, 1, 2 \quad (2.29)$$

$$|C_m| \leq C. \quad (2.30)$$

*Proof.* We proof (2.26). We choose  $K > \max\{1, \gamma_f, \gamma_a, \gamma_b\}$  such that:

$$\frac{C_f}{K^2} + \frac{C_a}{K} \frac{1}{1 - \gamma_a/K} + \frac{C_b}{K^2} \frac{1}{1 - \gamma_b/K} \leq 1$$

By Lemma 2.1, we can choose  $C$  such that (2.26) holds for  $n = 0, 1$ .

Now, we proof by induction over  $n$  and suppose that (2.26) holds for all  $k \leq n+1$ .

Therefore, we differentiate the differential equation (2.2)  $n$  times:

$$-\epsilon u_\epsilon^{(n+2)} = f^{(n)} - (a u_\epsilon')^{(n)} - (b u_\epsilon)^{(n)} = f^{(n)} - \sum_{k=0}^n \binom{n}{k} (a^{(k)} u_\epsilon^{(n+1-k)} + b^{(k)} u_\epsilon^{(n-k)})$$

Using the induction hypothesis and the estimations (2.7) and (2.8), we get:

$$\begin{aligned} \epsilon \|u_\epsilon^{(n+2)}\|_{L^\infty(\Omega)} &\leq \|f^{(n)}\|_{L^\infty(\Omega)} + C \sum_{k=0}^n \binom{n}{k} (C_b \gamma_b^k k! K^{n-k} \max\{n-k, \frac{1}{\epsilon}\}^{n-k} \\ &\quad + C_a \gamma_a^k k! K^{n+1-k} \max\{n+1-k, \frac{1}{\epsilon}\}^{n+1-k}). \end{aligned}$$

We can estimate:

$$\begin{aligned} \underbrace{\binom{n}{k} k!}_{= \frac{n!}{(n-k)!} \leq n^k} \max\{n-k, \frac{1}{\epsilon}\}^{n-k} &\leq n^k \max\{n, \frac{1}{\epsilon}\}^{n-k} \leq \max\{n+1, \frac{1}{\epsilon}\}^{n+1} \\ \binom{n}{k} k! \max\{n+1-k, \frac{1}{\epsilon}\}^{n+1-k} &\leq n^k \max\{n+1, \frac{1}{\epsilon}\}^{n+1-k} \leq \max\{n+1, \frac{1}{\epsilon}\}^{n+1} \\ \|f^{(n)}\|_{L^\infty(\Omega)} &\stackrel{(2.9)}{\leq} C_f \gamma_f^n n! \leq C_f K^n \max\{n+1, \frac{1}{\epsilon}\}^{n+1} \end{aligned}$$

and simplify:

$$\begin{aligned} \epsilon \|u_\epsilon^{(n+2)}\|_{L^\infty(\Omega)} &\leq \|f^{(n)}\|_{L^\infty(\Omega)} + C K^{n+2} \max\{n+1, \frac{1}{\epsilon}\}^{n+1} \sum_{k=0}^n \left( \frac{C_a \gamma_a^k}{K^{k+1}} + \frac{C_b \gamma_b^k}{K^{k+2}} \right) \\ &\stackrel{\text{geometric series}}{\leq} C K^{n+2} \max\{n+1, \frac{1}{\epsilon}\}^{n+1} \underbrace{\left( \frac{C_f}{K^2} + \frac{C_a}{K} \frac{1}{1 - \gamma_a/K} + \frac{C_b}{K^2} \frac{1}{1 - \gamma_b/K} \right)}_{\leq 1, \text{ by the choice of } K} \\ &\leq C K^{n+2} \max\{n+1, \frac{1}{\epsilon}\}^{n+1}. \end{aligned}$$



Dividing by  $\epsilon$  concludes the induction argument and proves (2.26):

$$\|u_\epsilon^{(n+2)}\|_{L^\infty(\Omega)} \leq CK^{n+2} \max\{n+2, \frac{1}{\epsilon}\}^{n+2}$$

The proof of (2.27) proceeds analogue but with the induction argument stated in Lemma 2.2.

We proof (2.28). The assumptions of Lemma 2.3 are satisfied because  $\|u_0\|_{L^\infty(G)}$ , the size of  $G$  and the constants  $C$ ,  $K_1$  and  $K_2$  can be controlled by the constants of (2.3)-(2.9).

Therefore, we can estimate by Lemma 2.3:

$$\|u_j^{(n)}\|_{L^\infty(\Omega)} \leq Cj!n!K_1^jK_2^n, \quad \forall j, n \in \mathbb{N}_0.$$

Hence, we can also estimate  $w_m^{(n)}$  given by (2.11) for all  $n \in \mathbb{N}_0$ :

$$\|w_m^{(n)}\|_{L^\infty(\Omega)} \leq Cn!K_2^n \sum_{j=0}^m \epsilon^j K_1^j j! \leq Cn!K_2^n \underbrace{\sum_{j=0}^m (\epsilon K_1 m)^j}_{\substack{\epsilon K m \leq 1 \\ \leq m+1, \text{ for } K > K_1}} \leq CK^n n!$$

such that the inequality (2.28) yields.

Since we defined  $C_m$  in (2.17) as  $C_m = -w_m(1)$  we can immediately conclude (2.30) by (2.28) with  $n = 0$ :

$$|C_m| \leq C.$$

Finally, (2.29) for  $n = 0, 1$  follows by a modification of Lemma 2.1 and by (2.28) and the differential equation satisfied by  $r_m$  gives the claim for  $n = 2$  as in the proof of (2.26).  $\square$

# The hp discontinuous Galerkin Finite Element Method

---

In this chapter we introduce the hp-DG-FEM for convection-diffusion equations in one space dimension, explicit for our model problem (2.2), following the detailed concepts and proofs of [2, 3].

Therefore, we define the Finite Element spaces to derive a variational formulation and its Galerkin discretization. Further, we prove the well-posedness of this Galerkin discretization.

In a second step, we prove the existence and uniqueness of the Finite Element solution by using relevant tools from functional analysis as stated in [1].

## 3.1 Finite Element Spaces for the DG-FEM

In this Section, we introduce the hp-FE-spaces which we use to derive the variational formulation.

Therefore, we present first the Hilbertian Sobolev spaces  $H^k(\Omega)$  which are defined as:

$$H^0(\Omega) := L^2(\Omega)$$

$$H^1(\Omega) := \{u \in L^2(\Omega) : \exists g \in L^2(\Omega) \text{ s.t. } \int_{\Omega} u \phi' dx = - \int_{\Omega} g \phi dx, \forall \phi \in C_{\text{comp}}^1(\Omega)\}$$

$$H^k(\Omega) := \{u \in L^2(\Omega) : u' \in H^{k-1}(\Omega)\}.$$

$$L^2(\Omega) \supset H^1(\Omega) \supset \dots \supset H^k(\Omega).$$

In [1] is proven that functions in  $H^1(\Omega)$  are continuous.

Further, we introduce the FE meshes and the traces following the definitions in [3] (Section 2.1):

Firstly, we define the partition  $\mathcal{T} = \{K_1, K_2, \dots, K_N\} = \{K_i\}_{i=1}^N$  of  $\Omega$  into open intervals  $K_i = (x_{i-1}, x_i)$  with element-width  $h_i := x_i - x_{i-1}$  for  $i = 1, \dots, N$  and

$$-1 = x_0 < x_1 < \dots < x_{N-1} < x_N = 1.$$

We consider the  $K_i$ 's as images of a reference element  $I = (-1, 1)$  under the affine maps:

$$F_{K_i} : I \longrightarrow K_i$$

$$\zeta \mapsto \frac{\zeta h + x_i + x_{i-1}}{2}$$

and define the following subsets of  $\partial K_i$ :

$$\begin{aligned} \partial_- K_i &:= \{x \in \partial K_i : n_i(x) = -1\} \\ \partial_+ K_i &:= \{x \in \partial K_i : n_i(x) = +1\}, \end{aligned}$$

where  $n_i(x)$  is the unit outward normal vector to  $K_i$ .

The set of the inner nodal points is given by:

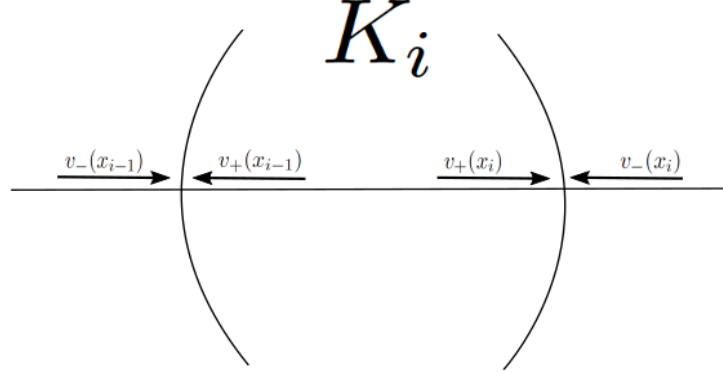
$$\Gamma_{int} := \{x_1, \dots, x_{N-1}\}.$$

In addition, we define for  $K_i \in \mathcal{T}$  and  $v \in H^1(\mathcal{T}, \Omega) := \{u \mid u \in H^1(K_i), \forall K_i \in \mathcal{T}\}$  the inner traces of  $v$  on  $\partial K_i$  for  $i = 1, \dots, N$ :

$$\begin{aligned} v_+(x_{i-1}) &:= \lim_{x \rightarrow x_{i-1}^+} v(x) \\ v_+(x_i) &:= \lim_{x \rightarrow x_i^-} v(x), \end{aligned}$$

the outer traces of  $v$  on  $\partial K_i$  for  $i = 1, \dots, N$ :

$$\begin{aligned} v_-(x_{i-1}) &:= \lim_{x \rightarrow x_{i-1}^-} v(x) \\ v_-(x_i) &:= \lim_{x \rightarrow x_i^+} v(x) \end{aligned}$$

Figure 3.1: inner and outer traces of  $v$  on  $\partial K_i$ 

and hence, for  $x_i \in \Gamma_{int}$  the jump operator:

$$[v](x_i) := \lim_{x \rightarrow x_i^+} v(x) - \lim_{x \rightarrow x_i^-} v(x)$$

and the average operator:

$$\langle v \rangle(x_i) := \frac{1}{2} \left( \lim_{x \rightarrow x_i^+} v(x) + \lim_{x \rightarrow x_i^-} v(x) \right).$$

The space of the polynomials of degree  $p \geq 0$  on  $I$  is given as:

$$\mathcal{P}_p(I) := \text{span}\{x^j : 0 \leq j \leq p, \text{ with } j, p \in \mathbb{N}_0\},$$

such that we can define the FE-spaces for the hp-FEM:

$$\mathcal{S}_0^{\mathbf{p},l}(\Omega, F_{\mathcal{T}}) := \{u \in H^l(\Omega) : u|_{K_i} \circ F_{K_i} \in \mathcal{P}_{p_i}, K_i \in \mathcal{T}, u_+(\pm 1) = 0\}, \quad (3.1)$$

where  $F_{\mathcal{T}} := \{F_{K_i} : K_i \in \mathcal{T}\}$  and  $\mathbf{p} := (p_i)_{i=1}^N$  with  $p_i$  the polynomial degree of the  $i$ -th element of the mesh  $\mathcal{T}$ .

The homogeneous Dirichlet boundary conditions are strongly enforced in the definition of  $\mathcal{S}_0^{\mathbf{p},l}(\Omega, F_{\mathcal{T}})$  by presuming  $u_+(\pm 1) = 0$ .

We notice, that in the case  $l = 0$  discontinuous functions are allowed and in the case  $l = 1$  the continuity is enforced such that we can define the FE-spaces for the hp-DG-FEM as:

$$\mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}}) := \{u \in H^0(\Omega) : u|_{K_i} \circ F_{K_i} \in \mathcal{P}_{p_i}, K_i \in \mathcal{T}, u_+(\pm 1) = 0\},$$

This space has dimension:

$$M = \dim(\mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})) = p_1 + \sum_{i=2}^{N-1} (p_i + 1) + p_N = N - 2 + \sum_{i=1}^N p_i,$$

since  $\dim(\mathcal{P}_{p_i}(I)) = p_i + 1$ ,  $u_+(\pm 1) = 0$  imposes two constraints and  $N = \#(\mathcal{T})$  is the number of elements.

### 3.2 Variational Formulation

The Variational formulation also known as the weak formulation forms the starting step of the Finite Element Method. It is an important tool for the analysis of differential equations, which permits the transfer of concepts of linear algebra. In the sense that the differential equation is written in a discretized form, which results in simpler equations and has instead weak solutions only with respect to certain test functions by using linear algebra methods over a vector space.

Firstly, we multiply the differential equation of our model problem (2.2) with a test function  $v \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$  and integrate over each interval  $K_i$  in  $\mathcal{T}$ :

$$\int_{x_{i-1}}^{x_i} -\epsilon u'' v \, dx + \int_{x_{i-1}}^{x_i} a(x) u' v \, dx + \int_{x_{i-1}}^{x_i} b(x) u v \, dx = \int_{x_{i-1}}^{x_i} f(x) v \, dx.$$

By integrating the first term by parts, we get:

$$\int_{x_{i-1}}^{x_i} \epsilon u' v' \, dx - \left[ \epsilon u' v \right]_{x_{i-1}}^{x_i} + \int_{x_{i-1}}^{x_i} a(x) u' v \, dx + \int_{x_{i-1}}^{x_i} b(x) u v \, dx = \int_{x_{i-1}}^{x_i} f(x) v \, dx.$$

By adding all  $N$  equations above, we obtain:

$$\begin{aligned} \sum_{i=1}^N \int_{x_{i-1}}^{x_i} \epsilon u' v' \, dx - \underbrace{\sum_{i=1}^N \left[ \epsilon u' v \right]_{x_{i-1}}^{x_i}}_{=-\sum_{i=1}^{N-1} \langle u' \rangle [v](x_i)} + \sum_{i=1}^N \int_{x_{i-1}}^{x_i} a(x) u' v \, dx + \sum_{i=1}^N \int_{x_{i-1}}^{x_i} b(x) u v \, dx \\ = \sum_{i=1}^N \int_{x_{i-1}}^{x_i} f(x) v \, dx. \end{aligned} \tag{3.2}$$

#### 3.2.1 Galerkin Discretization

We can state now the abstract hp discontinuous Galerkin Finite Element discretization of our model problem (2.2):

Find  $u_{DG} \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$  such that:

$$\begin{aligned}
B_{DG}(u_{DG}, v) &:= \epsilon \sum_{i=1}^N \int_{K_i} u'_{DG} v' dx + \epsilon \sum_{i=1}^{N-1} \{ \langle u'_{DG} \rangle [v] - [u_{DG}] \langle v' \rangle \} (x_i) \\
&\quad + \sum_{i=1}^N \int_{K_i} a u'_{DG} v dx + \sum_{i=1}^{N-1} a [u_{DG}] v_+(x_i) \\
&\quad + \sum_{i=1}^N \int_{K_i} b u_{DG} v dx \\
&= \sum_{i=1}^N \int_{K_i} f v dx =: l_{DG}(v), \quad \forall v \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}}),
\end{aligned} \tag{3.3}$$

where  $B_{DG}(\cdot, \cdot)$  is a bilinear form,  $l_{DG}(\cdot)$  a linear form and the terms  $[u_{DG}] \langle v \rangle (x_i)$  and  $a[u_{DG}] v_+(x_i)$  penalizes the jumps (discontinuities) of  $u_{DG}$ ,  $v$  and their derivatives in  $x_i$ .

We now prove that the Galerkin discretization (3.3) is consistent. Therefore, we show that the analytic solution  $u_\epsilon$  of our model problem (2.2) satisfies (3.3) by proving:

$$B_{DG}(u_\epsilon, v) = l_{DG}(v), \quad \forall v \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}}).$$

*Proof.* Since  $u_\epsilon$  is analytic we get for the jump and average operator for all inner nodes  $x_i$ :

$$\begin{aligned}
[u_\epsilon](x_i) &= 0 \\
\langle u'_\epsilon \rangle (x_i) &= u'_\epsilon(x_i)
\end{aligned}$$

and:

$$\begin{aligned}
\sum_{i=1}^{N-1} u'_\epsilon[v](x_i) &= \sum_{i=1}^{N-1} u'_\epsilon(v_- - v_+)(x_i) = \sum_{i=1}^{N-1} u'_\epsilon v_-(x_i) - \sum_{i=1}^{N-1} u'_\epsilon v_+(x_i) \\
&= \sum_{i=2}^N u'_\epsilon v_+(x_{i-1}) - \sum_{i=1}^{N-1} u'_\epsilon v_+(x_i) \\
&= - \sum_{i=2}^N n_i u'_\epsilon v_+(x_{i-1}) - \sum_{i=1}^{N-1} n_i u'_\epsilon v_+(x_i) \\
&= - \sum_{i=1}^N \sum_{x_j \in \partial K_i} n_i u'_\epsilon v_+(x_j)
\end{aligned}$$

Therefore, we can write  $B_{DG}(u_\epsilon, v)$  as:

$$\begin{aligned}
B_{DG}(u_\epsilon, v) &= \epsilon \sum_{i=1}^N \int_{K_i} u'_\epsilon v' dx + \epsilon \sum_{i=1}^{N-1} \{ \langle u'_\epsilon \rangle [v] - \underbrace{[u_\epsilon] \langle v' \rangle}_{=0} \}(x_i) \\
&\quad + \sum_{i=1}^N \int_{K_i} a u'_\epsilon v dx + \underbrace{\sum_{i=1}^{N-1} a [u_\epsilon] v_+(x_i)}_{=0} + \sum_{i=1}^N \int_{K_i} b u_\epsilon v dx \\
&= \epsilon \sum_{i=1}^N \int_{K_i} u'_\epsilon v' dx + \epsilon \underbrace{\sum_{i=1}^{N-1} u'_\epsilon [v](x_i)}_{=-\epsilon \sum_{i=1}^N \sum_{x_j \in \partial K_i} n_i u'_\epsilon v_+(x_j)} \\
&\quad + \sum_{i=1}^N \int_{K_i} a u'_\epsilon v dx + \sum_{i=1}^N \int_{K_i} b u_\epsilon v dx
\end{aligned}$$

and by integrating the first term by parts:

$$\begin{aligned}
B_{DG}(u_\epsilon, v) &= -\epsilon \sum_{i=1}^N \int_{K_i} u''_\epsilon v dx + \epsilon \sum_{i=1}^N \underbrace{\left[ u'_\epsilon v \right]_{\partial K_i}}_{=\sum_{x_j \in \partial K_i} n_i u'_\epsilon v_+(x_j)} - \epsilon \sum_{i=1}^N \sum_{x_j \in \partial K_i} n_i u'_\epsilon v_+(x_j) \\
&\quad + \sum_{i=1}^N \int_{K_i} a u'_\epsilon v dx + \sum_{i=1}^N \int_{K_i} b u_\epsilon v dx \\
&= \sum_{i=1}^N \int_{K_i} v \underbrace{\mathcal{L}_\epsilon u_\epsilon}_{=f} dx = l_{DG}(v)
\end{aligned}$$

□

Since  $B_{DG}$  is bilinear and the consistency of (3.3), we have the Galerkin orthogonality of the DG-error:

$$B_{DG}(u_\epsilon - u_{DG}, v) = 0, \quad \forall v \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}}).$$

### 3.3 Existence and Uniqueness of the DG-FE Solution

In this section, we prove the existence and uniqueness of the solution  $u_{DG} \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$  of the problem (3.3) for an arbitrary mesh  $\mathcal{T}$  and an arbitrary function  $f \in L^2(\Omega)$ .

In order to do so, we have to apply the generalisation of the Lax-Milgram Lemma (Peter Lax & Arthur Milgram, 1954) stated and proven in [1]:

**Lemma 3.1** (Lax-Milgram Lemma). *Let  $H$  be a Hilbert space and  $a(.,.) : H \times H \rightarrow \mathbb{R}$  a continuous, coercive bilinear form and  $V \subset H$  a closed subspace.*

*Then, for every  $l \in H^* := L(H; \mathbb{R})$  exists a unique solution  $u \in V$  of the variational problem:*

$$\text{Find } u \in V \text{ such that: } a(u, v) = l(v), \quad \forall v \in V,$$

In our explicit case, we have  $H := L^2(\Omega)$ ,  $V := \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$ ,  $a(.,.) := a_{DG}(.,.)$  and  $l(.) := l_{DG}(.)$ . We note that  $L^2(\Omega)$  is Hilbertian, proven in [9] (Chapter 2.4), and  $\mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$  is obviously a closed subspace of  $L^2(\Omega)$ .

Therefore, we only have to show that  $a_{DG}(.,.)$  is continuous and coercive and that  $l_{DG}(.)$  is continuous.

For this purpose, we define the norm  $\|\cdot\|_{DG}$  on  $\mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$  as in [2]:

$$\|u\|_{DG}^2 := \epsilon \sum_{i=1}^N \|u'\|_{L^2(K_i)}^2 + \sum_{i=1}^N \|\sqrt{c}u\|_{L^2(K_i)}^2 + \frac{1}{2} \sum_{i=1}^{N-1} a[u]^2(x_i),$$

where  $c(x) := b(x) - \frac{1}{2}a'(x) \geq c_0 \stackrel{(2.5)}{>} 0$  and prove that  $a_{DG}(.,.)$  and  $l_{DG}(.)$  are bounded following the proof in [2], Chapter 2.5 and [3], Chapter 2.3 (The concept of the relation between continuity and boundedness of linear maps between normed vector spaces is explained detailed in [9], Chapter 2.2):

**Lemma 3.2** (Continuity). *There exists a constant  $C > 0$  such that for all  $u, v \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$ :*

$$\begin{aligned} \sup_{\|u\|_{DG} \leq 1, \|v\|_{DG} \leq 1} |B_{DG}(u, v)| &\leq C \\ \sup_{\|v\|_{DG} \leq 1} |l_{DG}(v)| &\leq C \end{aligned}$$

For the proof of Lemma 3.2, we state two estimates of the traces defined in Section 3.1:

**Lemma 3.3** (Trace Estimate). *Let  $K_i \in \mathcal{T}$  and  $h_i$  be as in Section 3.1 and  $u \in H^1(K_i)$ .*

*Then, there exists a constant  $C > 0$  independent of  $u$  and  $K_i$  such that:*

$$|u_+(x)|^2 \leq C(h_i^{-1}\|u\|_{L^2(K_i)}^2 + h_i\|u'\|_{L^2(K_i)}^2), \quad x \in \partial K_i$$

*and for a polynomial  $v \in \mathcal{P}_{p_i}(K_i)$ :*

$$|v_+(x)|^2 \leq \frac{(p_i + 1)^2}{h_i} \|v\|_{L^2(K_i)}^2, \quad x \in \partial K_i$$



*Proof.* For the proof of the first estimate, we use the embedding theorem for Sobolev-spaces to estimate the trace  $u_+$ :

**Sobolev embedding:** Let  $\Omega \in \mathbb{R}$  be an open and bounded domain. Then, the embedding:

$$H^1 \subset C^0(\overline{\Omega})$$

is continuous and therefore, there exists a constant  $C > 0$  depending on  $\Omega$  such that:

$$\sup_{x \in \Omega} |u(x)| \leq C \|u\|_{H^1(\Omega)}, \quad u \in H^1(\Omega). \quad (3.4)$$

We define the linear mapping:

$$\begin{aligned} m_i : [0, 1] &\longrightarrow K_i \\ x &\longmapsto h_i(x - 1) + x_i. \end{aligned}$$

We apply the Sobolev embedding to estimate for  $y \in \overline{K}_i$ ,  $x \in [0, 1]$  with  $m_i(x) = y$  and  $C > 0$ :

$$\begin{aligned} |u_+(y)|^2 &= |(u_+ \circ m_i)(x)|^2 \leq \sup_{x \in [0, 1]} |(u \circ m_i)(x)|^2 \stackrel{(3.4)}{\leq} C \|u \circ m_i\|_{H^1(\Omega)}^2 \\ &= C \left( \int_0^1 ((u \circ m_i)(x))^2 dx + \int_0^1 ((u \circ m_i)'(x))^2 dx \right) \\ &= C \left( \int_{x_{i-1}}^{x_i} \underbrace{\frac{d(m_i^{-1}(y))}{dy}}_{=h_i^{-1}} u(y)^2 dy + \int_0^1 (u'(m_i(x))) \underbrace{m_i'(x)}_{=h_i} dx \right) \\ &= C \left( \frac{1}{h_i} \int_{x_{i-1}}^{x_i} u(y)^2 dy + \frac{h_i^2}{h_i} \int_{x_{i-1}}^{x_i} u'(y)^2 dy \right) \\ &= C (h_i^{-1} \|u\|_{L^2(K_i)}^2 + h_i \|u'\|_{L^2(K_i)}^2) \end{aligned}$$

The first estimate is achieved by choosing  $y \in \partial K_i \subset \overline{K}_i$ .

We prove now the second estimate of Lemma 3.3.

Since every polynomial can be written as a linear combination of the Legendre polynomials  $L_n$  we can describe  $v$  as the series:

$$v(x) = \sum_{n=0}^{p_i} a_n L_n(x), \quad a_n \in \mathbb{R}$$

and get for the  $L^2((-1, 1))$ -norm:

$$\begin{aligned}
\|v\|_{L^2((-1,1))}^2 &= \left\| \sum_{n=0}^{p_i} a_n L_n(x) \right\|_{L^2((-1,1))}^2 = \int_{-1}^1 \left( \sum_{n=0}^{p_i} a_n L_n(x) \right)^2 dx \\
&= \int_{-1}^1 \left( \sum_{n=0}^{p_i} a_n L_n(x) \right) \left( \sum_{l=0}^{p_i} a_l L_l(x) \right) dx = \int_{-1}^1 \sum_{n=0}^{p_i} \sum_{l=0}^{p_i} a_n L_n(x) a_l L_l(x) dx \\
&= \sum_{n=0}^{p_i} \sum_{l=0}^{p_i} a_n a_l \int_{-1}^1 L_n(x) L_l(x) dx.
\end{aligned}$$

By the orthogonality of the Legendre polynomials it holds:

$$\int_{-1}^1 L_n(x) L_l(x) dx = \begin{cases} \frac{2}{2n+1}, & n = l \\ 0, & \text{else} \end{cases}$$

and therefore:

$$\begin{aligned}
\|v\|_{L^2((-1,1))}^2 &= \sum_{n=0}^{p_i} \sum_{l=0}^{p_i} a_n a_l \int_{-1}^1 L_n(x) L_l(x) dx \\
&= \sum_{n=0}^{p_i} \frac{2a_n^2}{2n+1}.
\end{aligned}$$

Hence, we can estimate for  $x_{\pm} \in \partial K_i$  by using the property of the Legendre polynomials that  $|L_n(x)| \leq 1$  for all  $n \in \mathbb{N}$  and  $x \in [-1, 1]$  and  $y \in [0, 1]$  with  $m_i(y) = x_{\pm}$ :

$$\begin{aligned}
|v_+(x_{\pm})|^2 &= |v_+(m_i(y))|^2 \\
&\leq \left| \sum_{n=0}^{p_i} a_n L_n(m_i(y)) \right|^2 \leq \left| \sum_{n=0}^{p_i} a_n \right|^2, \quad m_i(y) \in \partial K_i \subset [-1, 1] \\
&\leq \left( \sum_{n=0}^{p_i} \frac{2a_n^2}{2n+1} \right) \underbrace{\left( \sum_{n=0}^{p_i} \frac{2n+1}{2} \right)}_{= \frac{(2p_i+1)(p_i+1)}{2}} \\
&\leq \|v\|_{L^2((-1,1))}^2 (p_i+1)^2 \\
&= (p_i+1)^2 \int_{-1}^1 v(x)^2 dx \\
&= (p_i+1)^2 \int_{x_{i-1}}^{x_i} v(z)^2 \frac{1}{h_i} dz, \quad z \in \overline{K_i} \\
&= \frac{(p_i+1)^2}{h_i} \|v\|_{L^2(K_i)}^2.
\end{aligned}$$

And hence, we can conclude that also the second trace estimate holds.  $\square$

With the aid of these estimates, we are able to prove Lemma 3.2.

*Proof.* Firstly, we integrate the integral  $\int au'v dx$  of  $B_{DG}$  defined in (3.3) by parts:

$$\begin{aligned}
\sum_{i=1}^N \int_{K_i} au'v dx &= \sum_{i=1}^N \left[ auv \right]_{x_{i-1}}^{x_i} - \sum_{i=1}^N \int_{K_i} (av)'u dx \\
&= \sum_{i=1}^N \left[ auv \right]_{x_{i-1}}^{x_i} - \sum_{i=1}^N \int_{K_i} (a'v + av')u dx \\
&= \sum_{i=1}^N \left[ auv \right]_{x_{i-1}}^{x_i} - \sum_{i=1}^N \int_{K_i} a'uv dx - \sum_{i=1}^N \int_{K_i} auv' dx \\
&= \sum_{i=1}^N \left[ auv \right]_{x_{i-1}}^{x_i} - \sum_{i=1}^N \int_{K_i} a'uv dx - \frac{1}{2} \sum_{i=1}^N \int_{K_i} auv' dx \\
&\quad + \frac{1}{2} \sum_{i=1}^N \int_{K_i} (au)'v dx - \frac{1}{2} \sum_{i=1}^N \left[ auv \right]_{x_{i-1}}^{x_i} \\
&= \frac{1}{2} \sum_{i=1}^N \left[ auv \right]_{x_{i-1}}^{x_i} - \frac{1}{2} \sum_{i=1}^N \int_{K_i} a'uv dx - \frac{1}{2} \sum_{i=1}^N \int_{K_i} auv' dx \\
&\quad + \frac{1}{2} \sum_{i=1}^N \int_{K_i} au'v dx.
\end{aligned}$$

We note that:

$$\begin{aligned}
&\frac{1}{2} \sum_{i=1}^N \left[ auv \right]_{x_{i-1}}^{x_i} + \sum_{i=1}^{N-1} a[u]v_+(x_i) = \frac{1}{2} \left( \sum_{i=1}^N auv(x_i) - auv(x_{i-1}) \right) + \sum_{i=1}^{N-1} a[u]v_+(x_i) \\
&= \underbrace{\frac{1}{2} auv(x_N) - \frac{1}{2} auv(x_0)}_{=0, \text{ since } u(-1)=0 \text{ and } u(1)=0} + \frac{1}{2} \sum_{i=1}^{N-1} \left( au_+v_+(x_i) - au_-v_-(x_i) \right) + \sum_{i=1}^{N-1} a[u]v_+(x_i) \\
&= \frac{1}{2} \sum_{i=1}^{N-1} \left( a(u_+v_+ - u_-v_-)(x_i) \right) + \frac{1}{2} \sum_{i=1}^{N-1} a(u_-v_- - u_+v_+ + [u][v] + ([u]\langle v \rangle - \langle u \rangle[v]))(x_i) \\
&= \frac{1}{2} \sum_{i=1}^{N-1} a([u]\langle v \rangle - \langle u \rangle[v] + [u][v])(x_i)
\end{aligned}$$

and therefore:

$$\begin{aligned}
|B_{DG}(u, v)| &= \left| \epsilon \sum_{i=1}^N \int_{K_i} u' v' dx + \epsilon \sum_{i=1}^{N-1} (\langle u' \rangle [v] - [u] \langle v' \rangle)(x_i) + \sum_{i=1}^N \int_{K_i} a u' v dx \right. \\
&\quad \left. + \sum_{i=1}^N \int_{K_i} b u v dx + \sum_{i=1}^{N-1} a [u] v_+(x_i) \right| \\
&= \left| \epsilon \sum_{i=1}^N \int_{K_i} u' v' dx + \epsilon \sum_{i=1}^{N-1} (\langle u' \rangle [v] - [u] \langle v' \rangle)(x_i) + \frac{1}{2} \sum_{i=1}^N \left[ a u v \right]_{x_{i-1}}^{x_i} \right. \\
&\quad \left. - \frac{1}{2} \sum_{i=1}^N \int_{K_i} a' u v dx - \frac{1}{2} \sum_{i=1}^N \int_{K_i} a u v' dx + \frac{1}{2} \sum_{i=1}^N \int_{K_i} a u' v dx \right. \\
&\quad \left. + \sum_{i=1}^N \int_{K_i} b u v dx + \sum_{i=1}^{N-1} a [u] v_+(x_i) \right| \\
&= \left| \underbrace{\epsilon \sum_{i=1}^N \int_{K_i} u' v' dx}_{\leq \|u'\|_{L^2(K_i)} \|v'\|_{L^2(K_i)}, \text{ by Hölder's Inequality}} + \epsilon \sum_{i=1}^{N-1} (\langle u' \rangle [v] - [u] \langle v' \rangle)(x_i) + \frac{1}{2} \sum_{i=1}^N \int_{K_i} a (u' v - u v') dx \right. \\
&\quad \left. + \sum_{i=1}^N \int_{K_i} \underbrace{\left(b - \frac{1}{2} a'\right)}_{=c(x)} u v dx + \frac{1}{2} \sum_{i=1}^{N-1} a (([u] \langle v \rangle - \langle u \rangle [v]) + [u] [v])(x_i) \right| \\
&\leq \epsilon \sum_{i=1}^N \|u'\|_{L^2(K_i)} \|v'\|_{L^2(K_i)} + \frac{\epsilon}{\sqrt{a_0}} \sum_{i=1}^{N-1} \sqrt{a} (|\langle u' \rangle| |[v]| + |[u]| |\langle v' \rangle|)(x_i) \\
&\quad + \frac{C_a}{\sqrt{2c_0}} \sum_{i=1}^N (\|u'\|_{L^2(K_i)} \|\sqrt{c} v\|_{L^2(K_i)} - \|\sqrt{c} u\|_{L^2(K_i)} \|v'\|_{L^2(K_i)}) \\
&\quad + \sum_{i=1}^N \|\sqrt{c} u\|_{L^2(K_i)} \|\sqrt{c} v\|_{L^2(K_i)} + \frac{1}{2} \sum_{i=1}^{N-1} a |[u]| |[v]|(x_i) \\
&\quad + \frac{\sqrt{C_a}}{2} \sum_{i=1}^{N-1} \sqrt{a} (|[u]| |\langle v \rangle| + |\langle u \rangle| |[v]|)(x_i)
\end{aligned}$$

$$\begin{aligned}
&\leq \left( \left( \epsilon + \frac{C_a}{2\sqrt{c_0}} \right) \sum_{i=1}^N \|u'\|_{L^2(K_i)}^2 + \left( 1 + \frac{C_a}{2\sqrt{c_0}} \right) \sum_{i=1}^N \|\sqrt{c}u\|_{L^2(K_i)}^2 \right. \\
&\quad + \left( \frac{1}{2} + \frac{\sqrt{C_a}}{2} + \frac{\epsilon}{\sqrt{a_0}} \right) \sum_{i=1}^{N-1} a[u]^2(x_i) + \frac{\sqrt{C_a}}{2} \sum_{i=1}^{N-1} |\langle u \rangle(x_i)|^2 \\
&\quad + \frac{\epsilon}{\sqrt{a_0}} \sum_{i=1}^{N-1} |\langle u' \rangle(x_i)|^2 \Big)^{\frac{1}{2}} \left( \left( \epsilon + \frac{C_a}{2\sqrt{c_0}} \right) \sum_{i=1}^N \|v'\|_{L^2(K_i)}^2 \right. \\
&\quad + \left( 1 + \frac{C_a}{2\sqrt{c_0}} \right) \sum_{i=1}^N \|\sqrt{c}v\|_{L^2(K_i)}^2 + \left( \frac{1}{2} + \frac{\sqrt{C_a}}{2} + \frac{\epsilon}{\sqrt{a_0}} \right) \sum_{i=1}^{N-1} a[v]^2(x_i) \\
&\quad + \frac{\sqrt{C_a}}{2} \sum_{i=1}^{N-1} |\langle v \rangle(x_i)|^2 + \frac{\epsilon}{\sqrt{a_0}} \sum_{i=1}^{N-1} |\langle v' \rangle(x_i)|^2 \Big)^{\frac{1}{2}} \\
&\leq C_0 \left( \|u\|_{DG}^2 + \sum_{i=1}^{N-1} |\langle u \rangle(x_i)|^2 + \sum_{i=1}^{N-1} |\langle u' \rangle(x_i)|^2 \right)^{\frac{1}{2}} \\
&\quad \cdot \left( \|v\|_{DG}^2 + \sum_{i=1}^{N-1} |\langle v \rangle(x_i)|^2 + \sum_{i=1}^{N-1} |\langle v' \rangle(x_i)|^2 \right)^{\frac{1}{2}}.
\end{aligned}$$

We estimate the term  $\sum_{i=1}^{N-1} |\langle u \rangle(x_i)|^2$  by the first trace estimate of Lemma 3.3:

$$\begin{aligned}
\sum_{i=1}^{N-1} |\langle u \rangle(x_i)|^2 &\leq \sum_{i=1}^N \sum_{x_j \in \partial_- K_i} |\langle u \rangle(x_j)|^2 = \frac{1}{4} \sum_{i=1}^N \sum_{x_j \in \partial_- K_i} |(u_+ + u_-)(x_j)|^2 \\
&= \frac{1}{4} \sum_{i=1}^N \sum_{x_j \in \partial_- K_i} |(u_+^2 + u_-^2 + \underbrace{2u_+u_-}_{\leq u_-^2 + u_+^2})(x_j)| \leq \frac{1}{2} \sum_{i=1}^N \sum_{x_j \in \partial_- K_i} (u_+^2 + u_-^2)(x_j) \\
&= \frac{1}{2} \sum_{i=1}^N \sum_{x_j \in \partial K_i} |u_+(x_j)|^2 \leq C_1 \sum_{i=1}^N (h_i^{-1} \|u\|_{L^2(K_i)}^2 + h_i \|u'\|_{L^2(K_i)}^2) \\
&\leq C \|u\|_{DG}^2,
\end{aligned}$$

where  $C$  depends on  $h_i$ .

Further, we can estimate the term  $\sum_{i=1}^{N-1} |\langle u' \rangle(x_i)|^2$  with the second trace estimate of Lemma 3.3:

$$\begin{aligned}
\sum_{i=1}^{N-1} |\langle u' \rangle(x_i)|^2 &\leq \sum_{i=1}^N \sum_{x_j \in \partial K_i} |u'_+(x_j)|^2 \leq 2 \sum_{i=1}^N \frac{(p_i + 1)^2}{h_i} \underbrace{\|u'\|_{L^2(K_i)}^2}_{\leq \|u\|_{DG}^2} \\
&\leq C \|u\|_{DG}^2,
\end{aligned}$$

where  $C$  also depends on  $h_i$ .

Therefore, we can estimate  $|B_{DG}(u, v)|$  by:

$$\begin{aligned} |B_{DG}(u, v)| &\leq C_0 \left( \|u\|_{DG}^2 + \sum_{i=1}^{N-1} |\langle u \rangle(x_i)|^2 + \sum_{i=1}^{N-1} |\langle u' \rangle(x_i)|^2 \right)^{\frac{1}{2}} \\ &\quad \cdot \left( \|v\|_{DG}^2 + \sum_{i=1}^{N-1} |\langle v \rangle(x_i)|^2 + \sum_{i=1}^{N-1} |\langle v' \rangle(x_i)|^2 \right)^{\frac{1}{2}} \\ &\leq C \|u\|_{DG} \|v\|_{DG} \end{aligned}$$

and conclude that there exists a constant  $C > 0$  such that:

$$\sup_{\|u\|_{DG} \leq 1, \|v\|_{DG} \leq 1} |B_{DG}(u, v)| \leq C.$$

Finally, we prove the boundedness of  $\sup_{\|v\|_{DG} \leq 1} |l_{DG}(v)|$  by estimating:

$$|l_{DG}(v)| \leq \sum_{i=1}^N \underbrace{\int_{K_i} |fv| dx}_{\|f\|_{L^2(K_i)} \|v\|_{L^2(K_i)}, \text{ by Hölder}} \leq \frac{1}{\sqrt{c_0}} \sum_{i=1}^N \|f\|_{L^2(K_i)} \underbrace{\|\sqrt{c}v\|_{L^2(K_i)}}_{\leq \|v\|_{DG}} \leq C \|v\|_{DG}$$

such that:

$$\sup_{\|v\|_{DG} \leq 1} |l_{DG}(v)| \leq C.$$

□

Consequently, we proved that  $B_{DG}(\cdot, \cdot)$  and  $l_{DG}(\cdot)$  are continuous. Now, we only have to show that  $B_{DG}(\cdot, \cdot)$  is coercive to apply Lemma 3.1:

**Lemma 3.4.** *The bilinear form  $B_{DG}(\cdot, \cdot)$  is coercive, i.e.  $B_{DG}(\cdot, \cdot)$  admits also a lower bound:*

$$\exists c > 0 : B_{DG}(u, u) \geq c \|u\|_{DG}^2, \quad u \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_T)$$

*Proof.*

$$\begin{aligned} B_{DG}(u, u) &= \epsilon \sum_{i=1}^N \int_{K_i} (u')^2 dx + \epsilon \underbrace{\sum_{i=1}^{N-1} (\langle u' \rangle[u] - [u]\langle u' \rangle)(x_i)}_{=0} + \frac{1}{2} \underbrace{\sum_{i=1}^N \int_{K_i} a(u'u - uu') dx}_{=0} \\ &\quad + \sum_{i=1}^N \int_{K_i} cu^2 dx + \frac{1}{2} \sum_{i=1}^{N-1} a(\underbrace{([u]\langle u \rangle - \langle u \rangle[u])}_{=0} + [u][u])(x_i) \\ &= \epsilon \sum_{i=1}^N \|u'\|_{L^2(K_i)}^2 + \sum_{i=1}^N \|\sqrt{c}u\|_{L^2(K_i)}^2 + \frac{1}{2} \sum_{i=1}^{N-1} a[u]^2(x_i) \\ &= \|u\|_{DG}^2 \end{aligned}$$

Therefore, we can choose  $c \leq 1$  to conclude the estimate. □

Hence, we can apply our generalised Lax-Milgram Lemma (Lemma 3.1) and conclude that the solution  $u_{DG} \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$  of the problem (3.3) for an arbitrary mesh  $\mathcal{T}$  and an arbitrary function  $f \in L^2(\Omega)$  exists and is unique.

# Stability and Convergence of the hp-DG-FEM

---

In this chapter, we analyse in a first step the stability of the hp-DG-FEM solution  $u_{DG}$  by proving the continuous dependence on initial data and some apriori estimates.

In a second step, we present the main statement of [2, 3], that the hp-DG-FEM for the convection-diffusion problem in one space dimension converges at a robust, independent of  $\epsilon$ , exponential rate.

## 4.1 Stability

We present two stability statements about  $u_{DG}$  by following the stability propositions in [2, 3].

Firstly, we prove that we can estimate  $\|u_{DG}\|_{DG}$  by the input data independently of the chosen mesh  $\mathcal{T}$ .

**Lemma 4.1** (Stability-Lemma 1). *The hp-DG-FEM solution  $u_{DG}$  of (3.3) can be estimated by the input functions  $a, b, f \in C^\infty(\overline{\Omega})$ :*

$$\|u_{DG}\|_{DG} \leq \frac{1}{\sqrt{c_0}} \|f\|_{L^2(\Omega)}$$

We note that Lemma 4.1 would also hold with weaker assumptions on the initial data, that is  $f \in L^2(\overline{\Omega})$ ,  $b \in L^\infty(\overline{\Omega})$  and  $a \in C^1(\overline{\Omega})$ .



*Proof.* We can estimate by Lemma 3.4 and by the Hölder's Inequality:

$$\begin{aligned}
\|u_{DG}\|_{DG}^2 &\leq B_{DG}(u_{DG}, u_{DG}) \stackrel{(3.3)}{=} l_{DG}(u_{DG}) = \sum_{i=1}^N \int_{K_i} f u_{DG} dx \\
&\stackrel{\text{Hölder}}{\leq} \frac{1}{\sqrt{c_0}} \sum_{i=1}^N \|f\|_{L^2(K_i)} \|\sqrt{c} u_{DG}\|_{L^2(K_i)} \\
&\leq \frac{1}{\sqrt{c_0}} \left( \underbrace{\sum_{i=1}^N \|f\|_{L^2(K_i)}^2}_{=\sum_i \int_{K_i} |f|^2 dx = \int_{\Omega} |f|^2 dx} \right)^{\frac{1}{2}} \left( \underbrace{\sum_{i=1}^N \|\sqrt{c} u_{DG}\|_{L^2(K_i)}^2}_{\leq \|u_{DG}\|_{DG}^2} \right)^{\frac{1}{2}} \\
&\leq \frac{1}{\sqrt{c_0}} \|f\|_{L^2(\Omega)} \|u_{DG}\|_{DG}.
\end{aligned}$$

We can end the proof by dividing by  $\|u_{DG}\|_{DG}$ .  $\square$

For the second stability statement, we have to do some preparatory work because we consider the case where  $\epsilon$  is very small such that the solution  $u_\epsilon$  exhibits a boundary layer at the outflow boundary  $x = 1$ , explained in section 2.2.

In [10] Chapter 5, it is explained and proven in detail that the mesh  $\mathcal{T}$  of the hp-DG-FEM must contain an extra element of size  $\mathcal{O}(\epsilon p)$  in the boundary layer to solve the problem (2.2) with this hindered conditions.

Therefore, we define a simple boundary resolving 2-element-mesh:

**Definition 4.2** (2-element-meshes). For  $\epsilon > 0$ ,  $\kappa > 0$  and  $p \in \mathbb{N}$ , we define a 2-element-mesh  $\mathcal{T} = \mathcal{T}_{\epsilon, \kappa, p}$  as:

$$\mathcal{T}_{\epsilon, \kappa, p} = \begin{cases} \{(-1, 1 - \epsilon \kappa p), (1 - \epsilon \kappa p, 1)\} & \text{if } \epsilon \kappa p < 1 \\ \{\Omega\} & \text{if } \epsilon \kappa p \geq 1. \end{cases}$$

We now present some approximation results on the defined 2-element-mesh  $\mathcal{T}_{\epsilon, \kappa, p}$  as stated in [11], Section 2.2, Lemma 2.4:

**Lemma 4.3.** *There are  $C$ ,  $\sigma$ ,  $\kappa_0 > 0$  depending only on  $a$ ,  $b$  and  $f$  such that the following holds. Assume that for every  $p \in \mathbb{N}$  a mesh  $\mathcal{T}$  is given such that for some  $\kappa \in (0, \kappa_0)$  we have  $\mathcal{S}_0^{\mathbf{p}, 1}(\mathcal{T}_{\epsilon, \kappa, p}) \subset \mathcal{S}_0^{\mathbf{p}, 1}(\mathcal{T})$ . Then, for each  $p \in \mathbb{N}$  the solution  $u_\epsilon$  of (2.2) admits a splitting:*

$$u_\epsilon = u_{reg} + u_{BL}$$

and there is  $u_p \in \mathcal{S}_0^{\mathbf{p}, 1}(\mathcal{T})$  with a corresponding splitting:

$$u_p = u_{reg, p} + u_{BL, p} \quad \text{with } u_{reg, p}, u_{BL, p} \in \mathcal{S}_0^{\mathbf{p}, 1}(\mathcal{T})$$

such that the errors

$$\eta_{reg} := u_{reg} - u_{reg,p} \quad \eta_{BL} := u_{BL} - u_{BL,p} \quad \eta := \eta_{reg} + \eta_{BL}$$

satisfy  $\eta_{reg}, \eta_{BL} \in H_0^1(\Omega) := \{u \in H^1(\Omega) \mid u(-1) = u(1) = 0\}$  and

$$\begin{aligned} \|\eta'_{reg}\|_{L^\infty(K_i)} + \|\eta_{reg}\|_{L^\infty(K_i)} &\leq Ch_i e^{-\sigma p}, & i = 1, \dots, N \\ \|\eta_{BL}^l\|_{L^\infty(\Omega)} &\leq C(\epsilon \kappa p)^{-l} e^{-\sigma \kappa p}, & l = 0, 1 \\ \epsilon \sqrt{\kappa p} \|\eta'_{BL}\|_{L^2(\Omega)} + \|\eta_{BL}\|_{L^2(\Omega)} &\leq C \sqrt{\epsilon} e^{-\sigma \kappa p} \end{aligned}$$

and

$$\sum_{i=1}^N \min\left\{1, \frac{h_i}{\epsilon}\right\} \left( (\epsilon \kappa p)^2 \|\eta'\|_{L^\infty(K_i)}^2 + \|\eta\|_{L^\infty(K_i)}^2 \right) \leq C e^{-\sigma \kappa p}.$$

*Proof.* A sketch of the proof is presented in [11]. □

By Lemma 4.3, we can rewrite the hp-DG-FEM-error as:

$$u_\epsilon - u_{DG} = \eta + \zeta$$

with

$$\begin{aligned} \eta &= \eta_{reg} + \eta_{BL} = u_\epsilon - (u_{reg,p} + u_{BL,p}) \\ \zeta &:= u_{reg,p} + u_{BL,p} - u_{DG}. \end{aligned}$$

In the second stability statement, we show that we can estimate the hp-DG-FEM-error only by  $\eta_{reg}$  and  $\eta_{BL}$ :

**Lemma 4.4** (Stability-Lemma 2). *There exists a constant  $C > 0$  depending only on the constants in (2.3)-(2.9) such that for the DG-FEM (3.3) on the 2-element-mesh in Definition 4.2 there holds the apriori estimate:*

$$\begin{aligned} \|\zeta\|_{DG}^2 &\leq C \left( \sum_{i=1}^2 \|\eta'_{reg}\|_{L^2(K_i)}^2 + \sum_{i=1}^2 \|\eta_{reg}\|_{L^2(K_i)}^2 + \epsilon^2 \sum_{i=1}^2 \|\eta'_{reg}\|_{L^\infty(K_i)}^2 \right. \\ &\quad \left. + \epsilon \|\eta'_{BL}\|_{L^2(\Omega)}^2 + \epsilon^2 \|\eta'_{BL}\|_{L^\infty(\Omega)}^2 + \frac{1}{\epsilon} \|\eta_{BL}\|_{L^2(\Omega)}^2 + \|\eta_{BL}\|_{L^\infty(\Omega)}^2 \right) \end{aligned}$$

*Proof.* From our rewritten hp-DG-FEM-error it follows:

$$|B_{DG}(\eta, \zeta)| = |B_{DG}(u_\epsilon - u_{DG} - \zeta, \zeta)| \stackrel{\text{bilinearity}}{=} |B_{DG}(u_\epsilon - u_{DG}, \zeta) - B_{DG}(\zeta, \zeta)|$$

and by the Galerkin orthogonality of the DG-error which we showed in Section 3.2.1 and by the proof of Lemma 3.4 we get:

$$\begin{aligned} |B_{DG}(\eta, \zeta)| &= \left| \underbrace{B_{DG}(u_\epsilon - u_{DG}, \zeta)}_{=0} - B_{DG}(\zeta, \zeta) \right| = | - B_{DG}(\zeta, \zeta) | = |B_{DG}(\zeta, \zeta)| \\ &\stackrel{\text{Lemma 3.4}}{=} \|\zeta\|_{DG}^2 \end{aligned}$$

We know that the interpolant  $\eta$  in Lemma 4.3 is continuous on  $\Omega$  and therefore the jump operator vanishes on the inner node  $[\eta](x_1) = 0$  and we can write (3.3) as:

$$B_{DG}(\eta, \zeta) = \epsilon \sum_{i=1}^2 \int_{K_i} \eta' \zeta' dx + \epsilon \langle \eta' \rangle [\zeta](x_1) + \sum_{i=1}^2 \int_{K_i} a \eta' \zeta + b \eta \zeta dx.$$

We insert the regular and singular terms of  $\eta$  to transform the last integral term:

$$\begin{aligned} \sum_{i=1}^2 \int_{K_i} a \eta' \zeta + b \eta \zeta dx &= \sum_{i=1}^2 \int_{K_i} a \eta'_{BL} \zeta + a \eta'_{reg} \zeta + b \eta \zeta dx \\ &= \sum_{i=1}^2 \left( \underbrace{\int_{K_i} a \eta'_{BL} \zeta dx}_{= [a \eta_{BL} \zeta]_{x_{i-1}}^{x_i} - \int_{K_i} \eta_{BL} (a \zeta)' dx \text{ by integration by parts}} + \int_{K_i} a \eta'_{reg} \zeta dx + \int_{K_i} b \eta \zeta dx \right) \\ &= \sum_{i=1}^2 \left( \int_{K_i} -\eta_{BL} (a \zeta)' + a \eta'_{reg} \zeta + b \eta \zeta dx + [a \eta_{BL} \zeta]_{x_{i-1}}^{x_i} \right) \end{aligned}$$

Since  $a$  and  $\eta_{BL}$  are continuous and  $\zeta$  vanishes on  $x_0$  and  $x_N$  by the boundary conditions we get:

$$\sum_{i=1}^2 [a \eta_{BL} \zeta]_{x_{i-1}}^{x_i} = a \eta_{BL} \zeta_+(x_1) - a \eta_{BL} \zeta_-(x_1) = -a \eta_{BL} [\zeta](x_1)$$

and therefore:

$$\sum_{i=1}^2 \int_{K_i} a \eta' \zeta + b \eta \zeta dx = \sum_{i=1}^2 \left( \int_{K_i} -\eta_{BL} (a \zeta)' + a \eta'_{reg} \zeta + b \eta \zeta dx \right) - a \eta_{BL} [\zeta](x_1).$$

It follows:

$$\begin{aligned}
|B_{DG}(\eta, \zeta)| &= \left| \epsilon \sum_{i=1}^2 \int_{K_i} \eta' \zeta' dx + \epsilon \langle \eta' \rangle [\zeta](x_1) + \sum_{i=1}^2 \int_{K_i} a \eta' \zeta + b \eta \zeta dx \right| \\
&= \left| \epsilon \sum_{i=1}^2 \int_{K_i} \eta' \zeta' dx + \epsilon \langle \eta' \rangle [\zeta](x_1) \right. \\
&\quad \left. + \sum_{i=1}^2 \left( \int_{K_i} -\eta_{BL}(a\zeta)' + a \eta'_{reg} \zeta + b \eta \zeta dx \right) - a \eta_{BL} [\zeta](x_1) \right| \\
&= \left| \sqrt{\epsilon} \sum_{i=1}^2 \int_{K_i} \zeta' (\sqrt{\epsilon} \eta' - \frac{a}{\sqrt{\epsilon}} \eta_{BL}) dx \right. \\
&\quad \left. + \sum_{i=1}^2 \int_{K_i} \zeta (a \eta'_{reg} - a' \eta_{BL} + b \eta) dx + \sqrt{a} [\zeta] \left( \frac{\epsilon}{\sqrt{a}} \langle \eta' \rangle - \sqrt{a} \eta_{BL} \right) (x_1) \right| \\
&\stackrel{\text{Hölder}}{\leq} \sum_{i=1}^2 \|\sqrt{\epsilon} \zeta'\|_{L^2(K_i)} \|\sqrt{\epsilon} \eta' - \frac{a}{\sqrt{\epsilon}} \eta_{BL}\|_{L^2(K_i)} \\
&\quad + \frac{1}{\sqrt{c_0}} \sum_{i=1}^2 \|\sqrt{c} \zeta\|_{L^2(K_i)} \|a \eta'_{reg} - a' \eta_{BL} + b \eta\|_{L^2(K_i)} \\
&\quad + \sqrt{a} [\zeta] \left\| \frac{\epsilon}{\sqrt{a}} \langle \eta' \rangle - \sqrt{a} \eta_{BL} \right\| (x_1)
\end{aligned}$$

and since the inequality  $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$  holds for all  $a, b \in \mathbb{R}_+$  and by the defined norm  $\|\cdot\|_{DG}$  in section 3.3 we can estimate further:

$$\begin{aligned}
|B_{DG}(\eta, \zeta)| &\leq \|\zeta\|_{DG} \left( \sum_{i=1}^2 \|\sqrt{\epsilon} \eta' - \frac{a}{\sqrt{\epsilon}} \eta_{BL}\|_{L^2(K_i)}^2 \right. \\
&\quad \left. + \frac{1}{c_0} \sum_{i=1}^2 \|a \eta'_{reg} - a' \eta_{BL} + b \eta\|_{L^2(K_i)}^2 \right. \\
&\quad \left. + 2 \left\| \frac{\epsilon}{\sqrt{a}} \langle \eta' \rangle - \sqrt{a} \eta_{BL} \right\| (x_1) \right)^{\frac{1}{2}} \\
&\leq \|\zeta\|_{DG} \left( \left( 3\epsilon + \frac{4C_a^2}{c_0} \right) \sum_{i=1}^2 \|\eta'_{reg}\|_{L^2(K_i)}^2 + 3\epsilon \sum_{i=1}^2 \|\eta'_{BL}\|_{L^2(K_i)}^2 \right. \\
&\quad \left. + \frac{4C_b^2}{c_0} \sum_{i=1}^2 \|\eta_{reg}\|_{L^2(K_i)}^2 + \frac{4}{c_0} (C_a^2 \gamma_a^2 + C_b^2 + \frac{3C_a^2 c_0}{4\epsilon}) \sum_{i=1}^2 \|\eta_{BL}\|_{L^2(K_i)}^2 \right. \\
&\quad \left. + 4 \left( \frac{\epsilon^2}{a_0} |\langle \eta' \rangle (x_1)|^2 + C_a |\eta_{BL}(x_1)|^2 \right) \right)^{\frac{1}{2}}.
\end{aligned}$$

For the last step we use the inequality  $2ab \leq a^2 + b^2$  for all  $a, b \in \mathbb{R}_+$  and the following estimate of the average operator of  $\eta'$  on  $\Omega$ , where we use its continuity:

$$\begin{aligned} |\langle \eta' \rangle(x_1)|^2 &= |\eta'(x_1)|^2 = |\eta'_{reg}(x_1) + \eta'_{BL}(x_1)|^2 \\ &\leq |\eta'_{reg}(x_1)|^2 + |\eta'_{BL}(x_1)|^2 + 2|\eta'_{reg}(x_1)||\eta'_{BL}(x_1)| \\ &\leq 2|\eta'_{reg}(x_1)|^2 + 2|\eta'_{BL}(x_1)|^2 \\ &\leq \sum_{i=1}^2 \|\eta'_{reg}\|_{L^\infty(K_i)}^2 + 2\|\eta'_{BL}\|_{L^\infty(\Omega)}^2 \end{aligned}$$

Hence, we can conclude:

$$\begin{aligned} |B_{DG}(\eta, \zeta)| &\leq \|\zeta\|_{DG} \left( \left( 3\epsilon + \frac{4C_a^2}{c_0} \right) \sum_{i=1}^2 \|\eta'_{reg}\|_{L^2(K_i)}^2 + 3\epsilon \|\eta'_{BL}\|_{L^2(\Omega)}^2 \right. \\ &\quad + \frac{4C_b^2}{c_0} \sum_{i=1}^2 \|\eta_{reg}\|_{L^2(K_i)}^2 + \frac{4}{c_0} (C_a^2 \gamma_a^2 + C_b^2 + \frac{3C_a^2 c_0}{4\epsilon}) \|\eta_{BL}\|_{L^2(\Omega)}^2 \\ &\quad \left. + \frac{4\epsilon^2}{a_0} \sum_{i=1}^2 \|\eta'_{reg}\|_{L^\infty(K_i)}^2 + \frac{8\epsilon^2}{a_0} \|\eta'_{BL}\|_{L^\infty(\Omega)}^2 + 4C_a \|\eta_{BL}\|_{L^\infty(\Omega)}^2 \right)^{\frac{1}{2}}. \end{aligned}$$

By dividing both sides by  $\|\zeta\|_{DG}$ , squaring and choosing  $C > 0$  big enough we get:

$$\begin{aligned} |\zeta|_{DG}^2 &\leq C \left( \sum_{i=1}^2 \|\eta'_{reg}\|_{L^2(K_i)}^2 + \epsilon \|\eta'_{BL}\|_{L^2(\Omega)}^2 + \sum_{i=1}^2 \|\eta_{reg}\|_{L^2(K_i)}^2 + \frac{1}{\epsilon} \|\eta_{BL}\|_{L^2(\Omega)}^2 \right. \\ &\quad \left. + \epsilon^2 \sum_{i=1}^2 \|\eta'_{reg}\|_{L^\infty(K_i)}^2 + \epsilon^2 \|\eta'_{BL}\|_{L^\infty(\Omega)}^2 + \|\eta_{BL}\|_{L^\infty(\Omega)}^2 \right). \end{aligned}$$

□

## 4.2 Convergence

In this section, we present the main theorem of [2, 3], the robust exponential convergence of the hp discontinuous Galerkin FEM.

Therefore, we assume that the conditions (2.3) - (2.9) hold and the assumptions in Lemma 4.3 are satisfied.

**Theorem 4.5.** *There exists two constants  $C, \lambda > 0$ , both independent of  $p$  and  $\epsilon$  such that the hp-DG-FEM (3.3) with the 2-element-mesh  $\mathcal{T}_{\epsilon, \kappa, p}$  as in Definition 4.2 converges at an exponential rate, independent of  $\epsilon$ :*

$$\|u_\epsilon - u_{DG}\|_{DG}^2 \leq C e^{-\lambda p} \quad (4.1)$$

*Proof.* By Lemma 4.1, we can rewrite the hp-DG-FEM-error as:

$$u_\epsilon - u_{DG} = \eta + \zeta$$

and therefore, we can estimate by the triangular inequality and  $2ab \leq a^2 + b^2$  for all  $a, b \in \mathbb{R}_+$ :

$$\|u_\epsilon - u_{DG}\|_{DG}^2 = \|\eta + \zeta\|_{DG}^2 \leq 2\|\eta\|_{DG}^2 + 2\|\zeta\|_{DG}^2.$$

We estimate the term of the interpolant  $\eta$  with aid of the definition of the norm  $\|\cdot\|_{DG}$ :

$$\begin{aligned} \|\eta\|_{DG}^2 &\stackrel{\text{Def.}}{=} \epsilon \sum_{i=1}^2 \|\eta'\|_{L^2(K_i)}^2 + \sum_{i=1}^2 \|\sqrt{c}\eta\|_{L^2(K_i)}^2 + \frac{1}{2}a \underbrace{[\eta]^2}_{=0 \text{ since } \eta \text{ is continuous on } \Omega}(x_1) \\ &= \epsilon \sum_{i=1}^2 \|\eta'_{reg} + \eta'_{BL}\|_{L^2(K_i)}^2 + \sum_{i=1}^2 \|\sqrt{c}(\eta_{reg} + \eta_{BL})\|_{L^2(K_i)}^2 \\ &\leq 2\epsilon \sum_{i=1}^2 \left( \|\eta'_{reg}\|_{L^2(K_i)}^2 + \|\eta'_{BL}\|_{L^2(K_i)}^2 \right) + 2\bar{c} \sum_{i=1}^2 \left( \|\eta_{reg}\|_{L^2(K_i)}^2 + \|\eta_{BL}\|_{L^2(K_i)}^2 \right) \end{aligned}$$

By Lemma 4.4, it follows:

$$\begin{aligned} \|u_\epsilon - u_{DG}\|_{DG}^2 &\leq C_0 \left( \sum_{i=1}^2 \|\eta'_{reg}\|_{L^2(K_i)}^2 + \epsilon \|\eta'_{BL}\|_{L^2(\Omega)}^2 + \sum_{i=1}^2 \|\eta_{reg}\|_{L^2(K_i)}^2 + \frac{1}{\epsilon} \|\eta_{BL}\|_{L^2(\Omega)}^2 \right. \\ &\quad \left. + \epsilon^2 \sum_{i=1}^2 \|\eta'_{reg}\|_{L^\infty(K_i)}^2 + \epsilon^2 \|\eta'_{BL}\|_{L^\infty(\Omega)}^2 + \|\eta_{BL}\|_{L^\infty(\Omega)}^2 \right) \end{aligned}$$

for a constant  $C_0 > 0$  depending only on the constants in (2.3) - (2.9).

Now, we make the following auxiliary estimate for all  $u \in L^\infty(K_i)$ , where  $i = 1, 2$ :

$$\|u\|_{L^2(K_i)}^2 = \int_{K_i} |u|^2 dx \leq \int_{K_i} \max_{x \in K_i} |u(x)|^2 dx = h_i (\max_{x \in K_i} |u(x)|)^2 = h_i \|u\|_{L^\infty(K_i)}^2$$

and therefore we get:

$$\begin{aligned} \|u_\epsilon - u_{DG}\|_{DG}^2 &\leq C_0 \left( \sum_{i=1}^2 (h_i + \epsilon^2) \|\eta'_{reg}\|_{L^\infty(K_i)}^2 + \epsilon \|\eta'_{BL}\|_{L^2(\Omega)}^2 + \sum_{i=1}^2 h_i \|\eta_{reg}\|_{L^\infty(K_i)}^2 \right. \\ &\quad \left. + \frac{1}{\epsilon} \|\eta_{BL}\|_{L^2(\Omega)}^2 + \epsilon^2 \|\eta'_{BL}\|_{L^\infty(\Omega)}^2 + \|\eta_{BL}\|_{L^\infty(\Omega)}^2 \right). \end{aligned}$$

We can conclude by squaring and adjusting the estimates in Lemma 4.3 that there exist constants  $\sigma, C_1 > 0$  not depending on  $\epsilon$  such that:

$$\begin{aligned} \epsilon \|\eta'_{BL}\|_{L^2(\Omega)}^2 + \frac{1}{\epsilon} \|\eta_{BL}\|_{L^2(\Omega)}^2 &\leq \frac{C_1}{\kappa p} e^{-\sigma \kappa p} \\ \|\eta_{BL}\|_{L^\infty(\Omega)}^2 &\leq C_1 e^{-\sigma \kappa p} \\ \epsilon^2 \|\eta'_{BL}\|_{L^\infty(\Omega)}^2 &\leq \frac{C_1}{(\kappa p)^2} e^{-\sigma \kappa p} \\ (h_i + \epsilon^2) \|\eta'_{reg}\|_{L^\infty(K_i)}^2 + h_i \|\eta_{reg}\|_{L^\infty(K_i)}^2 &\leq C_1 (h_i + \epsilon^2) h_i^2 e^{-\sigma p}. \end{aligned}$$

Therefore, we can estimate:

$$\begin{aligned} \|u_\epsilon - u_{DG}\|_{DG}^2 &\leq C_0 \left( C_1 e^{-\sigma p} \sum_{i=1}^2 (h_i + \epsilon^2) h_i^2 + C_1 \left(1 + \frac{1}{\kappa p} + \frac{1}{(\kappa p)^2}\right) e^{-\sigma \kappa p} \right) \\ &\leq C \left( e^{-\sigma p} \sum_{i=1}^2 (h_i + \epsilon^2) h_i^2 + \left(1 + \frac{1}{\kappa p} + \frac{1}{(\kappa p)^2}\right) e^{-\sigma \kappa p} \right) \\ &\leq C e^{-\lambda p} \end{aligned}$$

for constants  $\lambda, C > 0$  big enough and independent of  $\epsilon$ . □

# Numerical Scheme and Implementations

---

In this chapter, we firstly introduce a basic algorithmic pattern of the hp-DG-FEM by illustrating this numerical scheme for our model problem (2.2). For this purpose, we follow some of the algorithmic concepts of the FEM stated in [1, 4, 12].

Secondly, we present the implementations corresponding to the hp-DG-FEM algorithm and a numerical experiment in order to show the theoretical finding of the robust exponential convergence as in [2].

## 5.1 Numerical Scheme

The Galerkin discretization (3.3) defined in Section 3.2.1 corresponds to a linear system of equations with  $M = \dim(\mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}}))$  unknowns:

$$\begin{aligned} u_{DG}(x) &= \sum_{i=1}^M \alpha_i b_i(x) = \underline{\alpha}^\top \underline{b} \\ v &= \sum_{j=1}^M v_j b_j(x) = \underline{v}^\top \underline{b}, \end{aligned} \tag{5.1}$$

where  $\underline{b}$  is the vector of the FE-basis functions of  $\mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$ ,  $\underline{v}$  indicates the vector of coefficients for the test function  $v$  and  $\underline{\alpha}$  is the vector of unknowns.

In order to present the hp-DG-FEM algorithm, we firstly do not consider the homogeneous Dirichlet boundary conditions. Therefore, we set  $M := \sum_{k=1}^N (p_k + 1)$  such that  $\{b_i\}_{i=1}^M$  is a basis of  $\mathcal{S}^{\mathbf{p},0}(\Omega, F_{\mathcal{T}}) = \{u \in H^0(\Omega) : u|_{K_i} \circ F_{K_i} \in \mathcal{P}_{p_i}, K_i \in \mathcal{T}\}$ .



Substituting (5.1) into the Galerkin discretization (3.3) and utilising the bilinearity and linearity of  $B_{DG}(\cdot, \cdot)$  and  $l_{DG}(\cdot)$ , we can simplify and obtain the matrix formulation which is equivalent to (3.3) disregarding the boundary conditions:

$$\mathbf{A}\underline{\alpha} = \underline{l}, \quad (5.2)$$

where  $\mathbf{A} \in \mathbb{R}^{M \times M}$  is the global stiffness matrix with  $A_{ij} = B_{DG}(b_j, b_i) = (B_{DG}(b_i, b_j))^\top$  and  $\underline{l} \in \mathbb{R}^M$  is the global load vector with  $l_i = l(b_i)$ .

The bilinear form  $B_{DG}(\cdot, \cdot)$  is not symmetric and therefore we have  $\mathbf{A} \neq \mathbf{A}^\top$ .

### 5.1.1 Computing the elementwise Matrices and Vectors

Because of the non-symmetry of the stiffness matrix we first compute the transpose  $\mathbf{A}^\top$  of  $\mathbf{A}$  to not get an imbroglio with the indices and then transpose it back before solving the linear system (5.2).

Due to the local support of the global basis functions, the entries of the global block matrix  $\mathbf{A}^\top$  can be obtained by computing and assembling local matrices corresponding to elements or contact nodes between two elements. In what follows, we first describe how to compute the local matrices. We will regroup the terms defining  $A_{ij}^\top = B_{DG}(b_i, b_j)$  into two groups:

The terms involving integrals:

$$B_{DG}^e(b_i, b_j) := \epsilon \sum_{k=1}^N \int_{K_k} b'_i b'_j dx + \sum_{k=1}^N \int_{K_k} a(x) b'_i b_j dx + \sum_{k=1}^N \int_{K_k} b(x) b_i b_j dx$$

and the terms involving the interior nodes  $x_k$ :

$$B_{DG}^{int}(b_i, b_j) := \epsilon \sum_{k=1}^{N-1} \{ \langle b'_i \rangle [b_j] - [b_i] \langle b'_j \rangle \}(x_k) + \sum_{k=1}^{N-1} a[b_i](b_j)_+(x_k),$$

such that  $B_{DG}(b_i, b_j) = B_{DG}^e(b_i, b_j) + B_{DG}^{int}(b_i, b_j)$ .

First, we consider the terms corresponding to the integrals by computing the blocks  $\mathbf{A}_k^e \in \mathbb{R}^{(p_k+1) \times (p_k+1)}$  in  $\mathbf{A}^\top$  associated with each element  $k$ . Therefore, we rewrite the basis  $\{b_i\}_{i=1}^M$  of  $\mathcal{S}^{\mathbf{p},0}(\Omega, F\mathcal{T})$  as  $\{\underline{b}_k\}_{k=1}^N$  such that  $\underline{b}_k = \{b_i\}_{i=0}^{p_k}$  is the vector of basis functions corresponding to the  $k$ -th element and:

$$\begin{aligned} \mathbf{A}_k^e = & \epsilon \int_{K_k} (\underline{b}'_k(x)) (\underline{b}'_k(x))^\top dx + \int_{K_k} a(x) (\underline{b}'_k(x)) (\underline{b}_k(x))^\top dx \\ & + \int_{K_k} b(x) (\underline{b}_k(x)) (\underline{b}_k(x))^\top dx. \end{aligned}$$

This matrices can be evaluated independently for each element by transforming each element of the mesh  $\mathcal{T}$  to a reference element  $I = (-1, 1)$  via an element mapping:

$$K_k \ni x = F_{K_k}(\omega) := m_k + h_k \frac{\omega}{2}, \quad \omega \in I \text{ and } k = 0, \dots, N,$$

where  $K_k = (x_{k-1}, x_k)$  is the  $k$ -th element,  $h_k = x_k - x_{k-1}$  the element-width and  $m_k = \frac{x_{k-1} + x_k}{2}$  the element-mid-point.

Therefor, we introduce the so-called element shape-functions  $\underline{N}_{K_k}(x)$  for the  $k$ -th element  $K_k$  of the mesh  $\mathcal{T}$ :

$$\begin{aligned} \underline{b}_k(x) &= \underline{N}_{K_k}(x), & x \in K_k \\ \underline{N}_{K_k}(F_{K_k}(\omega)) &= \hat{N}_k(\omega), & \omega \in I \end{aligned}$$

where  $\hat{N}_k = \{\hat{N}_j\}_{j=0}^{p_k}$  are the reference element shape-functions on  $I$  which build a basis of  $\mathcal{P}_{p_k}(I)$  with  $p_k$  the polynomial degree of the  $k$ -th element of  $\mathcal{T}$ .

For  $\hat{N}_j$  we choose the 1D hierarchical reference element shape-functions which are explained detailed in [1].

We can build these functions out of the Legendre polynomials  $\{L_p\}_{p=0}^{\infty}$ .

The Legendre polynomials have the advantages that they can be easily evaluated at the bounds of the reference element  $I$ :

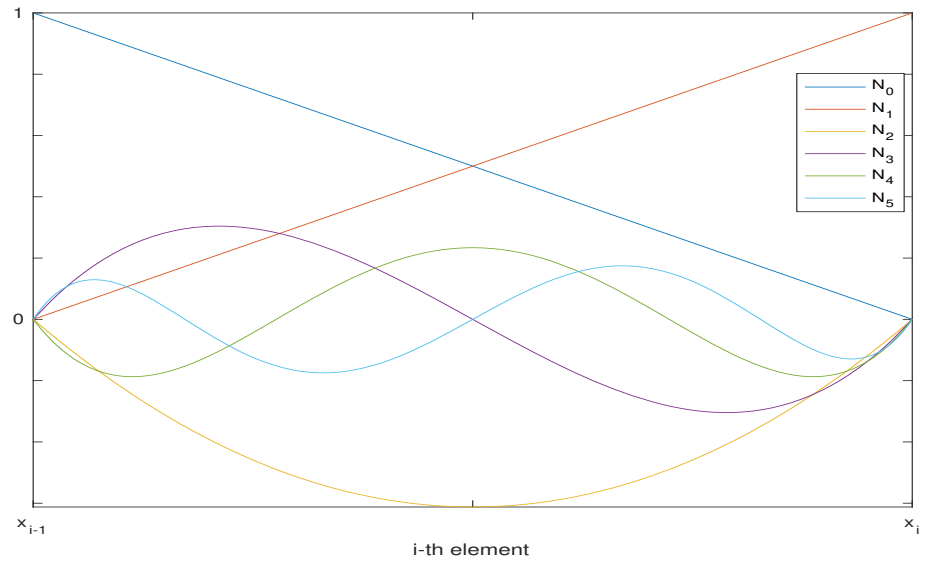
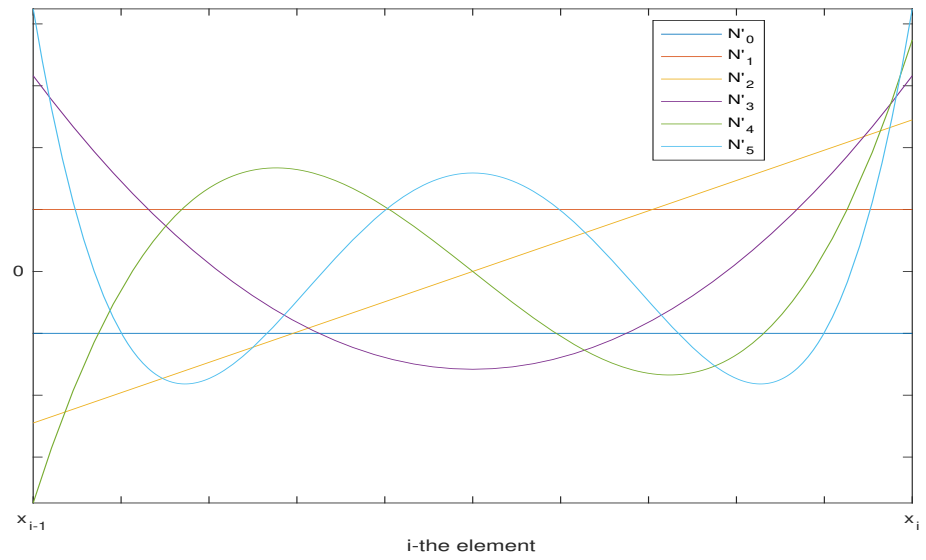
$$\begin{aligned} L_p(-1) &= (-1)^p \\ L_p(1) &= 1 \end{aligned}$$

and that they satisfy the orthogonality:

$$\int_{-1}^1 L_n(\omega) L_m(\omega) d\omega = \begin{cases} \frac{2}{2n+1} & \text{if } n = m \\ 0 & \text{else} \end{cases}.$$

A hierarchic sequence of reference element shape-functions  $\hat{N}_k$  can then be defined as:

$$\begin{aligned} \hat{N}_0(\omega) &= \frac{1 - \omega}{2} \\ \hat{N}_1(\omega) &= \frac{1 + \omega}{2} \\ \hat{N}_j(\omega) &= \sqrt{\frac{2j-1}{2}} \int_{-1}^{\omega} L_{j-1}(x) dx \\ &= \frac{1}{\sqrt{2(2j-1)}} (L_j - L_{j-2})(\omega), \quad j \geq 2 \end{aligned}$$

Figure 5.1: The first six shape-functions on the  $i$ -th element.Figure 5.2: The derivatives of the first six shape-functions on the  $i$ -th element.

Therefore, we can determine the elementwise local matrices  $\mathbf{A}_k^e$  as follows:

$$\begin{aligned} \mathbf{A}_k^e &= \mathbf{B}_{DG}^e(\underline{N}_k, \underline{N}_k) \\ &= \epsilon \int_{K_k} (\underline{N}'_k)(\underline{N}'_k)^\top dx + \int_{K_k} a(x)(\underline{N}'_k)(\underline{N}_k)^\top dx + \int_{K_k} b(x)(\underline{N}_k)(\underline{N}_k)^\top dx \\ &= \epsilon \int_I (\hat{N}'_k)(\hat{N}'_k)^\top \left( \frac{d\omega}{dx} \right) d\omega + \int_I \hat{a}_k(\omega)(\hat{N}'_k)(\hat{N}_k)^\top d\omega \\ &\quad + \int_I \hat{b}_k(\omega)(\hat{N}_k)(\hat{N}_k)^\top \left( \frac{dx}{d\omega} \right) d\omega \end{aligned}$$

where  $\hat{a}_k(\omega) = a(F_{K_k}(\omega))$  and  $\hat{b}_k(\omega) = b(F_{K_k}(\omega))$  and since  $\frac{d\omega}{dx} = \frac{2}{h_k}$  we obtain:

$$\begin{aligned} \mathbf{A}_k^e &= \underbrace{\epsilon \left( \frac{2}{h_k} \right) \int_I (\hat{N}'_k)(\hat{N}'_k)^\top d\omega}_{=:\mathbf{A}_k^\epsilon} + \underbrace{\int_I \hat{a}_k(\omega)(\hat{N}'_k)(\hat{N}_k)^\top d\omega}_{=:\mathbf{A}_k^a} \\ &\quad + \underbrace{\left( \frac{h_k}{2} \right) \int_I \hat{b}_k(\omega)(\hat{N}_k)(\hat{N}_k)^\top d\omega}_{=:\mathbf{A}_k^b} \end{aligned}$$

such that each entry can be computed as:

$$\begin{aligned} (A_k^\epsilon)_{ij} &= \epsilon \left( \frac{2}{h_k} \right) \int_I \hat{N}'_{i-1} \hat{N}'_{j-1} d\omega \\ (A_k^a)_{ij} &= \int_I \hat{a}_k(\omega) \hat{N}'_{i-1} \hat{N}_{j-1} d\omega \\ (A_k^b)_{ij} &= \left( \frac{h_k}{2} \right) \int_I \hat{b}_k(\omega) \hat{N}_{i-1} \hat{N}_{j-1} d\omega \\ (A_k^e)_{ij} &= (A_k^\epsilon)_{ij} + (A_k^a)_{ij} + (A_k^b)_{ij}. \end{aligned}$$

Now we consider the terms involving the interior nodes. Therefore, we have to compute the expanded matrices  $\mathbf{A}_k^{int} \in \mathbb{R}^{(p_k+p_{k+1}+2) \times (p_k+p_{k+1}+2)}$  for each interior node  $x_k$  which depend only on the two elements adjacent to the node:

$$\begin{aligned} \mathbf{A}_k^{int} &:= \epsilon \left\{ \left\langle \begin{pmatrix} \underline{N}'_k \\ \underline{N}'_{k+1} \end{pmatrix} \right\rangle \left[ \begin{pmatrix} \underline{N}_k \\ \underline{N}_{k+1} \end{pmatrix} \right]^\top - \left[ \begin{pmatrix} \underline{N}_k \\ \underline{N}_{k+1} \end{pmatrix} \right] \left\langle \begin{pmatrix} \underline{N}'_k \\ \underline{N}'_{k+1} \end{pmatrix} \right\rangle^\top \right\} (x_k) \\ &\quad + a \left[ \begin{pmatrix} \underline{N}_k \\ \underline{N}_{k+1} \end{pmatrix} \right] \begin{pmatrix} (\underline{N}_k)_- \\ (\underline{N}_{k+1})_+ \end{pmatrix}^\top (x_k). \end{aligned}$$

By writing out the jump and average operator, we can decompose  $\mathbf{A}_k^{int}$  as explained detailed in [12] into four matrices  $\mathbf{A}_k^{++} \in \mathbb{R}^{(p_k+1) \times (p_k+1)}$ ,  $\mathbf{A}_k^{+-} \in \mathbb{R}^{(p_k+1) \times (p_{k+1}+1)}$ ,

$\mathbf{A}_k^{-+} \in \mathbb{R}^{(p_{k+1}+1) \times (p_k+1)}$  and  $\mathbf{A}_k^{--} \in \mathbb{R}^{(p_{k+1}+1) \times (p_{k+1}+1)}$ :

$$\begin{aligned} \mathbf{A}_k^{int} &:= \epsilon \frac{1}{2} \left\{ \begin{pmatrix} (\underline{N}'_k)_- + (\underline{N}'_k)_+ \\ (\underline{N}'_{k+1})_+ + (\underline{N}'_{k+1})_- \end{pmatrix} \begin{pmatrix} (\underline{N}_k)_- - (\underline{N}_k)_+ \\ (\underline{N}_{k+1})_+ - (\underline{N}_{k+1})_- \end{pmatrix}^\top \right. \\ &\quad \left. - \begin{pmatrix} (\underline{N}_k)_- - (\underline{N}_k)_+ \\ (\underline{N}_{k+1})_+ - (\underline{N}_{k+1})_- \end{pmatrix} \begin{pmatrix} (\underline{N}'_k)_- + (\underline{N}'_k)_+ \\ (\underline{N}'_{k+1})_+ + (\underline{N}'_{k+1})_- \end{pmatrix}^\top \right\} (x_k) \\ &\quad + a \begin{pmatrix} (\underline{N}_k)_- - (\underline{N}_k)_+ \\ (\underline{N}_{k+1})_+ - (\underline{N}_{k+1})_- \end{pmatrix} \begin{pmatrix} (\underline{N}_k)_- \\ (\underline{N}_{k+1})_+ \end{pmatrix}^\top (x_k) \\ &= \begin{pmatrix} \mathbf{A}_k^{++} & \mathbf{A}_k^{+-} \\ \mathbf{A}_k^{-+} & \mathbf{A}_k^{--} \end{pmatrix}, \end{aligned}$$

where:

$$\begin{aligned} \mathbf{A}_k^{++} &= \left( \frac{1}{2} \epsilon ((\underline{N}_k)_+ (\underline{N}'_k)_+^\top - (\underline{N}'_k)_+ (\underline{N}_k)_+^\top) \right) (x_k) \\ \mathbf{A}_k^{+-} &= \left( \frac{1}{2} \epsilon ((\underline{N}'_k)_+ (\underline{N}_{k+1})_+^\top + (\underline{N}_k)_+ (\underline{N}'_{k+1})_+^\top) - a (\underline{N}_k) (\underline{N}_{k+1})_+^\top \right) (x_k) \\ \mathbf{A}_k^{-+} &= \left( -\frac{1}{2} \epsilon ((\underline{N}_{k+1})_+ (\underline{N}'_k)_+^\top + (\underline{N}'_{k+1})_+ (\underline{N}_k)_+^\top) \right) (x_k) \\ \mathbf{A}_k^{--} &= \left( \frac{1}{2} \epsilon ((\underline{N}'_{k+1})_+ (\underline{N}_{k+1})_+^\top - (\underline{N}_{k+1})_+ (\underline{N}'_{k+1})_+^\top) + a (\underline{N}_{k+1})_+ (\underline{N}_{k+1})_+^\top \right) (x_k), \end{aligned}$$

since  $(\underline{N}_k)_-(x_k) = (\underline{N}'_k)_-(x_k) = (\underline{N}_{k+1})_-(x_k) = (\underline{N}'_{k+1})_-(x_k) = 0$  because the element shape-functions vanish outside of their element.

We should notice that we can also transform the shape-functions to reference element shape-functions for the sake of convenience and that only the first two shape-functions and the derivatives of the shape-functions do not vanish at the boundaries (Figure 5.1 & 5.2).

Similarly to the global matrix  $\mathbf{A}^\top$ , we can obtain the global vector  $\underline{l}$  by first computing the local vectors  $\underline{l}^k$  corresponding to the  $k$ -th element:

$$\begin{aligned} \underline{l}^k &= \frac{h_k}{2} \int_I \hat{f}_k(\omega) \hat{N}(\omega) d\omega, \\ l_i^k &= \frac{h_k}{2} \int_I \hat{f}_k(\omega) \hat{N}_{i-1}(\omega) d\omega, \end{aligned}$$

where  $\hat{f}_k(\omega) = f(F_{K_k}(\omega))$ .

### 5.1.2 Assembly of the global Stiffness Matrix and Load Vector

The next step is to assemble the global matrix  $\mathbf{A}^\top$  and the global load vector  $\underline{l}$  in the following way:

$$\begin{aligned} \mathbf{A}^\top = & \begin{pmatrix} \mathbf{A}_1^e + \mathbf{A}_1^{++} & \mathbf{A}_1^{+-} & & & \\ \mathbf{A}_1^{-+} & \mathbf{A}_2^e + \mathbf{A}_1^{--} & & & \\ & & 0 & & \\ & & & \ddots & \\ & & & & 0 \end{pmatrix} + \begin{pmatrix} \mathbf{0}^{(p_1+1)^2} & & & & \\ & \mathbf{A}_2^{++} & \mathbf{A}_2^{+-} & & \\ & \mathbf{A}_2^{-+} & \mathbf{A}_3^e + \mathbf{A}_2^{--} & & \\ & & & 0 & \\ & & & & \ddots \\ & & & & & 0 \end{pmatrix} \\ & + \cdots + \begin{pmatrix} 0 & & & & \\ & \ddots & & & \\ & & 0 & & \\ & & & \mathbf{A}_{N-1}^{++} & \mathbf{A}_{N-1}^{+-} \\ & & & \mathbf{A}_{N-1}^{-+} & \mathbf{A}_N^e + \mathbf{A}_{N-1}^{--} \end{pmatrix} \\ \underline{l} = & \begin{pmatrix} \underline{l}^1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \begin{pmatrix} \underline{0}^{p_1+1} \\ \underline{l}^2 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \cdots + \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \underline{l}^N \end{pmatrix} \end{aligned}$$

By transposing  $\mathbf{A}^\top$  back to  $\mathbf{A}$  we obtain the desired global stiffness matrix  $(\mathbf{A}^\top)^\top = \mathbf{A}$ .

### 5.1.3 Imposing the Boundary Conditions

We can now modify the linear system in order to enforce the homogeneous Dirichlet boundary conditions. Since only the first two shape-functions  $N_0, N_1$  do not vanish at the boundary of the elements (Figure 5.1), we can conclude that the first and the  $(2 + \sum_{i=1}^{N-1} (p_i + 1))$ -th unknown  $\alpha_1, \alpha_d$  for  $d = 2 + \sum_{i=1}^{N-1} (p_i + 1)$  are equal to zero.

Therefore, we can delete the corresponding rows and columns marked below with red and define a new matrix  $\tilde{\mathbf{A}} \in \mathbb{R}^{(M-2) \times (M-2)}$  and vector  $\tilde{\underline{l}} \in \mathbb{R}^{(M-2)}$ :

$$\mathbf{A} = \begin{pmatrix} \textcolor{red}{A}_{11} & \textcolor{red}{A}_{12} & \cdots & \textcolor{red}{A}_{1d} & \cdots & \textcolor{red}{A}_{1 \text{ end}} \\ \textcolor{red}{A}_{21} & A_{22} & \cdots & \textcolor{red}{A}_{2d} & \cdots & A_{2 \text{ end}} \\ \vdots & & \ddots & \vdots & & \vdots \\ \textcolor{red}{A}_{d1} & \cdots & \cdots & \textcolor{red}{A}_{dd} & \cdots & \textcolor{red}{A}_{d \text{ end}} \\ \vdots & & & \vdots & \ddots & \vdots \\ \textcolor{red}{A}_{\text{end } 1} & \cdots & \cdots & \textcolor{red}{A}_{\text{end } d} & \cdots & A_{\text{end end}} \end{pmatrix} \xrightarrow{\text{Delete } \textcolor{red}{\text{red}} \text{ rows \& columns}} \tilde{\mathbf{A}}$$

$$\underline{l} = \begin{pmatrix} \textcolor{red}{l}_1 \\ l_2 \\ \vdots \\ l_{d-1} \\ \textcolor{red}{l}_d \\ l_{d+1} \\ \vdots \\ l_{\text{end}} \end{pmatrix} \xrightarrow{\text{Delete } \textcolor{red}{\text{red}} \text{ entries}} \tilde{\underline{l}}$$

We can now compute the vector of unknowns  $\underline{\alpha}$  by solving the linear system:

$$\tilde{\mathbf{A}}\underline{\alpha} = \tilde{\underline{l}}$$

and determine the approximated solution:

$$u_{DG}(x) = \sum_{i=1}^M \alpha_i b_i(x).$$

## 5.2 Implementations

In this section, we firstly present the implementation of the hp-DG-FEM for the model problem (2.2) in MATLAB, i.e. with double precision (16 decimal) accuracy, by following the numerical scheme presented in Section 5.1, and the simplified implementations for the case with constant coefficients such that  $a = b = f = 1$  as in [2] which we will use in a numerical experiment.

In a second step, we present the implementations for the  $L^2$ - and the  $L^\infty$ -error analysis.

### 5.2.1 hp-DG-FEM Solution

Below, the implemented algorithms to solve the Galerkin problem (3.3) are listed.

We start with the functions which generate the 1D hierarchic reference shape-functions and their first derivatives:

```

1  function shape_function = shape_func(i)
2  %%% The function determines the i-th shape-function
3
4
5  %% Input:
6
7  % i is the index
8
9
10 %% Output:
11
12 % The i-th reference shape-function (function-handle)
13
14
15 %% Definitions:
16
17 if i == 0
18     shape_function = @(x) 1/2*(1-x);
19
20 elseif i == 1
21     shape_function = @(x) 1/2*(1+x);
22
23 else
24     syms x;
25     LegP1 = legendreP(i,x);
26     LegP2 = legendreP(i-2,x);
27     shape_function = 1/(sqrt(2*(2*i-1)))*(LegP1-LegP2);
28     shape_function = matlabFunction(shape_function);
29 end
30
31
32 end

```



```

1 function shape_function_diff = shape_funct_diff(i)
2 %%% This function determines the derivative of the i-th shape-
3 function
4 %% Input:
5
6 % i is the index
7
8
9 %% Output:
10
11 % The derivative of the i-th reference shape-function (function-
12 handle)
13
14 %% Initialization:
15
16 syms x;
17 f = shape_funct(i);
18
19
20 %% Derivative:
21
22 shape_function_diff = matlabFunction(diff(f(x)));
23
24
25 end

```

The next three functions compute the elementwise matrices  $\mathbf{A}_k^\epsilon$ ,  $\mathbf{A}_k^a$  and  $\mathbf{A}_k^b$ , where the build-in function "integral" is used which bases upon a quadrature formula. Quadrature formulas require that  $a, b \in C^1(\overline{\Omega})$ .

```

1 function [M_epsilon] = matrix_epsilon(p,epsilon,x1,x2)
2 %%% Computation of the local matrix A^\epsilon.
3
4 %% Input:
5
6 % p          degree of local polynomial (integer)
7 % epsilon    diffusivity (real number between 0 and 1)
8 % x1         left bound of the element (real number)
9 % x2         right bound of the element (real number)
10
11
12 %% Output:
13
14 % Matrix describing the term of \int(\epsilon u'(x)v'(x))dx
15
16
17 %% Initialize:
18
19 M_epsilon = eye(p+1);

```

```

20
21 % Element-width
22 h = x2-x1;
23
24
25 %% Computation of the matrix:
26
27 M_epsilon(1,1) = 0.5;
28 M_epsilon(1,2) = -0.5;
29 M_epsilon(2,1) = -0.5;
30 M_epsilon(2,2) = 0.5;
31
32 M_epsilon = M_epsilon*epsilon*(2/h);
33
34
35 end

```

```

1 function [M_a] = matrix_a(p,a,x1,x2)
2 %%% Computation of the local matrix A^a.
3
4 %% Input:
5
6 % p          degree of local polynomial (integer)
7 % a          velocity of the transporting medium (function handle
8 )
9 % x1         left bound of the element (real number)
10 % x2         right bound of the element (real number)
11
12 %% Output:
13
14 % Matrix describing the term of \int(a(x)u'(x)v(x))dx
15
16
17 %% Initialization:
18
19 M_a = zeros(p+1);
20 % Element-width
21 h = x2-x1;
22 % Element-mid-point
23 m = (x1+x2)/2;
24 % Element mapping
25 phi = @(x) m + h*x/2;
26
27 % a through phi (Substitution) to work on reference element
28 syms x;
29 A = a(phi(x));
30 A = @(x) A;
31
32 %% Computation of the matrix:
33
34 for j = 1:p+1
35     u1 = -0.5;
36     u2 = 0.5;

```

```

37     v = shape_func(j-1);
38     F1 = A(x)*u1*v(x);
39     F2 = A(x)*u2*v(x);
40     F1 = matlabFunction(F1);
41     F2 = matlabFunction(F2);
42     M_a(1,j) = integral(F1,-1,1);
43     M_a(2,j) = integral(F2,-1,1);
44 end
45
46 for i = 3:p+1
47     for j = 1:p+1
48         s1 = shape_func_diff(i-1);
49         s2 = shape_func(j-1);
50         F1 = A(x)*s1(x)*s2(x);
51         F = matlabFunction(F1);
52         M_a(i,j) = integral(F,-1,1);
53     end
54 end
55
56 end

1 function [M_b] = matrix_b(p,b,x1,x2)
2 %%% Computation of the local matrix A~b.
3
4 %% Input:
5
6 % p          degree of local polynomial (integer)
7 % b          losses/sources of the substance (function handle)
8 % x1         left bound of the element (real number)
9 % x2         right bound of the element (real number)
10
11
12 %% Output:
13
14 % Matrix describing the term of \int(b(x)u(x)v(x))dx
15
16
17 %% Initialization:
18
19 M_b = zeros(p+1);
20 % Element-width
21 h = x2-x1;
22 % Element-mid-point
23 m = (x1+x2)/2;
24 % Element mapping
25 phi = @(x) m + h*x/2;
26
27 % b through phi (Substitution) to work on reference element
28 syms x;
29 B = b(phi(x));
30 B = @(x) B;
31
32 %% Computation of the matrix:
33

```

```

34 for i = 1:p+1
35     for j = i:p+1
36         s1 = shape_funct(i-1);
37         s2 = shape_funct(j-1);
38         F = B(x)*s1(x)*s2(x);
39         F = matlabFunction(F);
40         M_b(i,j) = integral(F,-1,1);
41         M_b(j,i) = M_b(i,j);
42     end
43 end
44
45 M_b = M_b*(h/2);
46
47 end

```

The four functions below compute the terms involving the interior nodes  $\mathbf{A}_k^{++}$ ,  $\mathbf{A}_k^{+-}$ ,  $\mathbf{A}_k^{-+}$  and  $\mathbf{A}_k^{--}$ :

```

1 function [M_a] = matrix_A_11(p,epsilon,x1,x2)
2 %% Computation of the local the matrix A^{++}
3
4 %% Input:
5
6 % p          degree of local polynomial (integer)
7 % epsilon    diffusivity (real number between 0 and 1)
8 % x1         left bound of the element (real number)
9 % x2         right bound of the element (real number)
10
11 %% Output:
12
13 % Matrix describing the term of the end-node of the
corresponding element
14
15
16 %% Initialization:
17
18 M_a = zeros(p+1);
19 % Element-width
20 h = x2-x1;
21
22
23 %% Computation of the matrix:
24
25 M_a(1,2) = 0.5*epsilon*0.5*2/h;
26 M_a(2,1) = -M_a(1,2);
27
28 for i = 3:p+1
29     M_a(2,i) = sqrt((2*(i-1)-1)/2)*0.5*epsilon*legendreP(i-2,1)
        *2/h;
30     M_a(i,2) = -M_a(2,i);
31 end
32
33 end

```

```

1 function [M_c] = matrix_A_12(p1,p2,a,epsilon,x1,x2,x3)
2 %%% Computation of the matrix  $A^{+-}$ 
3
4 %% Input:
5
6 % p1          degree of local polynomial of i-th element (integer)
7 % p2          degree of local polynomial of (i+1)-th element (
   integer)
8 % a          velocity of the transporting medium (function handle
   )
9 % epsilon    diffusivity (real number between 0 and 1)
10 % x1         left bound of the first element (real number)
11 % x2         interior node between the two elements (real number)
12 % x3         right bound of the second element (real number)
13
14 %% Output:
15
16 % Matrix describing a term of the node between two elements
17
18
19 %% Initialization:
20
21 M_c = zeros(p1+1,p2+1);
22 % Element-width of first element
23 h1 = x2-x1;
24 % Element-width of second element
25 h2 = x3-x2;
26 % a evaluated at the interior node x2 between the two elements
27 a_x2 = a(x2);
28
29 %% Computation of the matrix:
30
31 M_c(1,1) = -0.5*epsilon*0.5*2/h1;
32 M_c(2,2) = -M_c(1,1)*h1/h2;
33 M_c(2,1) = -a_x2+0.5*epsilon*(0.5*2/h1-0.5*2/h2);
34
35 for i = 3:p2+1
36     M_c(2,i) = sqrt((2*(i-1)-1)/2)*0.5*epsilon*legendreP(i-2,-1)
   *2/h2;
37 end
38
39 for i = 3:p1+1
40     M_c(i,1) = sqrt((2*(i-1)-1)/2)*0.5*epsilon*legendreP(i-2,1)
   *2/h1;
41 end
42
43
44 end

```

```

1 function [M_d] = matrix_A_21(p1,p2,epsilon,x1,x2,x3)
2 %%% Computation of the matrix  $A^{-+}$ 
3
4 %% Input:

```

```

5
6 % p1          degree of local polynomial of i-th element (integer)
7 % p2          degree of local polynomial of (i+1)-th element (
      integer)
8 % epsilon     diffusivity (real number between 0 and 1)
9 % x1          left bound of the first element (real number)
10 % x2         interior node between the two elements (real number)
11 % x3         right bound of the second element (real number)
12
13 %% Output:
14
15 % Matrix describing a term of the node between two elements
16
17
18 %% Initialization:
19
20 M_d = zeros(p2+1,p1+1);
21 % Element-width of first element
22 h1 = x2-x1;
23 % Element-width of second element
24 h2 = x3-x2;
25
26
27 %% Computation of the matrix:
28
29 M_d(1,1) = 0.5*epsilon*0.5*2/h1;
30 M_d(2,2) = -M_d(1,1)*h1/h2;
31 M_d(1,2) = 0.5*epsilon*(0.5*2/h2-0.5*2/h1);
32
33 for i = 3:p1+1
34     M_d(1,i) = -sqrt((2*(i-1)-1)/2)*0.5*epsilon*legendreP(i-2,1)
        *2/h1;
35 end
36
37 for i = 3:p2+1
38     M_d(i,2) = -sqrt((2*(i-1)-1)/2)*0.5*epsilon*legendreP(i-2,-1)
        *2/h2;
39 end
40
41
42 end

1 function [M_a] = matrix_A_22(p,a,epsilon,x1,x2)
2 %%% Computation of the matrix A^--
3
4 %% Input:
5
6 % p          degree of local polynomial (integer)
7 % a          velocity of the transporting medium (function handle
      )
8 % epsilon     diffusivity (real number between 0 and 1)
9 % x1         left bound of the element (real number)
10 % x2        right bound of the element (real number)
11

```

```

12 %% Output:
13
14 % Matrix describing the term of the start-node of the
    corresponding element
15
16
17 %% Initialization:
18
19 M_a = zeros(p+1);
20 % Element-width
21 h = x2-x1;
22 % a evaluated at the interior node x1
23 a_x1 = a(x1);
24
25 %% Computation of the matrix:
26
27 M_a(1,2) = -0.5*epsilon*0.5*2/h;
28 M_a(2,1) = -M_a(1,2);
29 M_a(1,1) = a(x1);
30
31 for i = 3:p+1
32     M_a(1,i) = -sqrt((2*(i-1)-1)/2)*0.5*epsilon*legendreP(i-2,-1)
        *2/h;
33     M_a(i,1) = -M_a(1,i);
34 end
35
36
37 end

```

The next function computes the elementwise load vector  $\underline{l}^k$  again using the build-function "integral" requiring  $f \in C^1(\bar{\Omega})$ :

```

1 function [load_v] = load_vector(p,f,x1,x2)
2 %%% Computation of the local load vector  $\underline{l}^k$ 
3
4
5 %% Input:
6
7 % p          degree of local polynomial (integer)
8 % f          external source term (function handle)
9 % x1         left bound of the element (real number)
10 % x2        right bound of the element (real number)
11
12
13 %% Initialization:
14
15 load_v = zeros(p+1,1);
16 % Element-width
17 h = x2-x1;
18 % Element-mid-point
19 m = (x1+x2)/2;
20 % Element mapping
21 g = @(x) m + h*x/2;
22

```

```

23 % f through g (Substitution) to work on reference element
24 syms x;
25 G = f(g(x));
26 G = @(x) G;
27
28 %% Computation of the element load vector:
29
30 for i = 1:p+1
31     u1 = shape_funct(i-1);
32     F = G(x)*u1(x);
33     F = matlabFunction(F);
34     load_v(i) = integral(F,-1,1);
35 end
36
37 load_v = load_v*(h/2);
38
39 end

```

The next function solves the model problem (2.2) for an arbitrary mesh and arbitrary polynomial degrees by using all the functions above and by following the numerical scheme explained in Section 5.1:

```

1 function sol = solution(p,n,epsilon,a,b,f)
2 %%% Solving the Differential equation:
3 %%% -\epsilon u'' + au' + bu = f with homogeneous Dirichlet
    boundary
4 %%% conditions (u(-1) = 0 = u(+1)) by using the hp-DG-FEM.
5
6 %% Input:
7
8 % p          the vector of the polynomial degrees of each element
    (natural
9 % numbers > 0)
10 % n          the vector of all nodes starting with -1 and ending
    with 1
11 % (increasing series of real numbers between -1 and 1)
12 % epsilon    diffusivity (real number between 0 and 1)
13 % a          velocity of the transporting medium (function handle
    )
14 % b          losses/sources of the substance (function handle)
15 % f          external source term (function handle)
16
17
18 %% Output:
19
20 % f is the piecewise continuous function which approximates the
    exact
21 % solution of the Differential equation
22
23
24 %% Generating element mesh:
25
26 % Number of elements

```



```

27 num = length(n)-1;
28
29 % Mesh
30 % Initialization
31 m = zeros(num,2);
32
33 for i = 1:num
34     m(i,1) = n(i);
35     m(i,2) = n(i+1);
36 end
37
38
39 %% Computations of the element matrices and load vectors:
40
41 % Computation of first element matrix  $A_1^e + A_1^{++}$ 
42 A1 = matrix_a(p(1),a,m(1,1),m(1,2)) + matrix_b(p(1),b,m(1,1),m(
    1,2)) + matrix_epsilon(p(1),epsilon,m(1,1),m(1,2)) +
    matrix_A_11(p(1),epsilon,m(1,1),m(1,2));
43
44 % Computation of last element matrix  $A_N^e + A^{--}$ 
45 A2 = matrix_a(p(end),a,m(end,1),m(end,2)) + matrix_b(p(end),b,m(
    end,1),m(end,2)) + matrix_epsilon(p(end),epsilon,m(end,1),m(
    end,2)) + matrix_A_22(p(end),a,epsilon,m(end,1),m(end,2));
46
47 % Maximum polynomial degree in vector p
48 maxp = max(p);
49
50 % Computation of i-th element matrix ( $1 < i < n$ )  $A_i^e + A_{i-1}^{--} + A_i^{++}$ 
51 % Initialization
52 A_store = zeros(maxp+1,maxp+1,num);
53 % Storing first and last element matrix
54 A_store(1:p(1)+1,1:p(1)+1,1) = A1;
55 A_store(1:p(end)+1,1:p(end)+1,end) = A2;
56 % Storing all other element matrices
57 for i = 2:num-1
58     A_store(1:p(i)+1,1:p(i)+1,i) = matrix_a(p(i),a,m(i,1),m(i,2))
        + matrix_b(p(i),b,m(i,1),m(i,2)) + matrix_epsilon(p(i),
        epsilon,m(i,1),m(i,2)) + matrix_A_22(p(i),a,epsilon,m(i
        ,1),m(i,2)) + matrix_A_11(p(i),epsilon,m(i,1),m(i,2));
59 end
60
61 % Computation of the matrices  $A_i^{+-}$  and  $A_i^{-+}$  describing the
    boundary
62 % part at the inner node of two neighbored elements
63 % Initialization
64 A_12_store = zeros(maxp+1,maxp+1,num-1);
65 A_21_store = zeros(maxp+1,maxp+1,num-1);
66 % Storing all matrices
67 for i = 1:num-1
68     A_12_store(1:p(i)+1,1:p(i+1)+1,i) = matrix_A_12(p(i),p(i+1),
        a,epsilon,m(i,1),m(i,2),m(i+1,2));
69     A_21_store(1:p(i+1)+1,1:p(i)+1,i) = matrix_A_21(p(i),p(i+1),
        epsilon,m(i,1),m(i,2),m(i+1,2));

```

```

70 end
71
72 % Computaion of the load vectors
73 % Initialization
74 l_store = zeros(maxp+1,1,num);
75 % Storing all local load vectors
76 for i = 1:num
77     l_store(1:p(i)+1,:,i) = load_vector(p(i),f,m(i,1),m(i,2));
78 end
79
80
81 %% Assembly of the global matrix and load vector:
82
83 % Initialization
84 A = zeros(sum(p)+num,sum(p)+num);
85 l = zeros(sum(p)+num,1);
86
87 % Help-Indices
88 a_help = 1;
89 b_help = 1;
90
91 % Assembly
92 % Assembly of the element matrices
93 for i = 1:num
94     A(a_help:a_help + p(i),a_help:a_help + p(i)) = A_store(1:p(i)
95         +1,1:p(i)+1,i);
96     l(a_help:a_help + p(i)) = l_store(1:p(i)+1,:,i);
97     a_help = a_help + p(i) + 1;
98 end
99
100 % Assembly of the interior node matrices
101 for i = 1:num-1
102     A(b_help:b_help + p(i),b_help + p(i) + 1:b_help + p(i) + p(i)
103         +1) = A_12_store(1:p(i)+1,1:p(i+1)+1,i);
104     A(b_help + p(i) + 1:b_help + p(i) + p(i+1) + 1,b_help:b_help
105         + p(i)) = A_21_store(1:p(i+1)+1,1:p(i)+1,i);
106     b_help = b_help + p(i) + 1;
107 end
108
109 % Transpose A
110 A = A';
111
112 %% Consideration of the homogeneous Dirichlet boundary condition
113 :
114
115 % Adjusting the elementwise load vectors
116 l(end-(p(end)-1)) = [];
117 l(1) = [];
118
119 % Adjusting the global stiffness Matrix
120 A(:,end-(p(end)-1)) = [];
121 A(end-(p(end)-1),:) = [];
122 A(:,1) = [];
123 A(1,:) = [];

```

```

120
121 A = sparse(A);
122
123
124 %% Solving the linear system:
125
126 % Computation of the coefficients
127 coeff = A\1;
128
129
130 %% Building the solution by adding up the basis elements
    multiplied by the corresponding coefficient:
131
132 % Initialization
133 sol = @(x) 0;
134 % First two shape-functions
135 s1 = shape_func(0);
136 s2 = shape_func(1);
137 % Index for coefficient vector:
138 a_help = 1;
139
140 % Adding up:
141
142     % First element:
143     % Width, midpoint and inverse reference map of the first
        element:
144     h1 = m(1,2)-m(1,1);
145     m1 = (m(1,1)+m(1,2))/2;
146     f1 = @(x) (x - m1)*2/h1;
147
148     syms x;
149     sol = sol + piecewise(x<m(1,1),0,m(1,1)<x<m(1,2),coeff(
        a_help)*s2(f1(x)),m(1,2)<x,0);
150     a_help = a_help+1;
151     for i = 2:p(1)
152         s3 = shape_func(i);
153         syms x;
154         sol = sol + piecewise(x<m(1,1),0,m(1,1)<x<m(1,2),coeff(
            a_help)*s3(f1(x)),m(1,2)<x,0);
155         a_help = a_help+1;
156     end
157
158     % i-th element:
159     for i = 2:num-1
160         % Width, midpoint and inverse reference map of the i-th
            element:
161         h3 = m(i,2)-m(i,1);
162         m3 = (m(i,1)+m(i,2))/2;
163         f3 = @(x) (x - m3)*2/h3;
164
165         syms x;
166         sol = sol + piecewise(x<m(i,1),0,m(i,1)<x<m(i,2),coeff(
            a_help)*s1(f3(x)),m(i,2)<x,0);
167         a_help = a_help+1;

```

```

168     syms x;
169     sol = sol + piecewise(x<m(i,1),0,m(i,1)<x<m(i,2),coeff(
170         a_help)*s2(f3(x)),m(i,2)<x,0);
171     a_help = a_help+1;
172
173     for j = 2:p(i)
174         s3 = shape_funct(j);
175         syms x;
176         sol = sol + piecewise(x<m(i,1),0,m(i,1)<x<m(i,2),coeff(
177             a_help)*s3(f3(x)),m(i,2)<x,0);
178         a_help = a_help+1;
179     end
180
181     % Last element
182     % Width, midpoint and inverse reference map of the last
183     % element:
184     h2 = m(end,2) - m(end,1);
185     m2 = (m(end,1) + m(end,2))/2;
186     f2 = @(x) (x - m2)*2/h2;
187
188     syms x;
189     sol = sol + piecewise(x<m(end,1),0,m(end,1)<x<m(end,2),coeff(
190         a_help)*s1(f2(x)),m(end,2)<x,0);
191     a_help = a_help+1;
192     for i = 2:p(end)
193         s3 = shape_funct(i);
194         syms x;
195         sol = sol + piecewise(x<m(end,1),0,m(end,1)<x<m(end,2),
196             coeff(a_help)*s3(f2(x)),m(end,2)<x,0);
197         a_help = a_help+1;
198     end
199 end

```

### 5.2.2 hp-DG-FEM with constant Coefficients and 2-element Mesh

For the numerical experiment we assume constant coefficients  $a = b = f = 1$  as in [2]. Therefore, we can simplify the implementation of the matrix  $\mathbf{A}_k^a$ ,  $\mathbf{A}_k^b$  and the vector  $\underline{l}^k$  by utilising the properties of the shape-functions and the Legendre polynomials explained in Subsection 5.1.1. This leads to faster and even more accurate computations by avoiding numerical integration.

```

1 function [M_a] = matrix_a(p)
2     %%% Computation of the local matrix A^a
3
4
5     %% Assumptions:
6
7     % a=1 is constant
8

```

```

9
10 %% Input:
11
12 % p          degree of local polynomial (integer)
13
14 %% Output:
15
16 % Matrix describing the term of \int(u'(x)v(x))dx
17
18
19 %% Initialization:
20
21 M_a = zeros(p+1);
22
23
24 %% Computation of the matrix:
25
26 M_a(1:2,1:2) = [-0.5, -0.5; 0.5, 0.5];
27
28 if p>1
29     M_a(1,3) = sqrt(1/6);
30     M_a(2,3) = -sqrt(1/6);
31     M_a(3,1) = -sqrt(1/6);
32     M_a(3,2) = sqrt(1/6);
33 end
34
35 for i = 3:p
36     M_a(i,i+1) = -1/sqrt(4*(i-1)^2-1);
37     M_a(i+1,i) = -M_a(i,i+1);
38 end
39
40
41 end

```

```

1 function [M_b] = matrix_b(p,x1,x2)
2 %%% Computation of the local matrix A^b
3
4
5 %% Assumptions:
6
7 % b=1 is constant
8
9
10 %% Input:
11
12 % p          degree of local polynomial (integer)
13 % x1         left bound of the element (real number)
14 % x2         right bound of the element (real number)
15
16
17 %% Output:
18
19 % Matrix describing the term of \int(u(x)v(x))dx
20

```

```

21
22 %% Initialization:
23
24 M_b = zeros(p+1);
25
26 % Element-width
27 h = x2-x1;
28
29
30 %% Computation of the matrix:
31
32 M_b (1:2 ,1:2)= [2/3 ,1/3;1/3 ,2/3];
33
34 if p > 1
35   M_b(1,3) = -1/sqrt(6);
36   M_b(2,3) = M_b(1,3);
37   M_b(3,1) = M_b(1,3);
38   M_b(3,2) = M_b(1,3);
39 end
40
41 if p > 2
42   M_b(1,4)= 1/(3*sqrt(10));
43   M_b(2,4)= -M_b(1,4);
44   M_b(4,1)= M_b(1,4);
45   M_b(4,2)= M_b (2,4);
46 end
47
48 for i = 3:p+1
49   M_b(i,i) = 1/(2*(i-1)-1)*(1/(2*(i-3)+1)+1/(2*(i-1)+1));
50
51   if(p+1 > i+1)
52     M_b(i,i+2) = -1/((2*i-1)*sqrt((2*i-3)*(2*i+1)));
53     M_b(i+2,i) = M_b(i,i+2);
54   end
55
56
57 end
58   M_b = M_b*(h/2);
59
60
61 end

```

```

1 function [load_v] = load_vector(p,x1,x2)
2 %%% Computation of the local load vector l^k
3
4
5 %% Assumptions:
6
7 % f=1 is constant.
8
9
10 %% Input:
11
12 % p          degree of local polynomial (integer)

```

```

13 % x1          left bound of the element (real number)
14 % x2          right bound of the element (real number)
15
16
17 %% Initialization:
18
19 load_v = zeros(p+1,1);
20
21 % Element-width
22 h = x2-x1;
23
24
25 %% Computation of the element load vector:
26
27 load_v(1:2) = [1;1];
28     if p > 1
29         load_v(3) = -2*sqrt(1/6);
30     end
31 load_v = load_v*(h/2);
32
33
34 end

```

The next function solves the model problem (2.2) with the assumption that  $a = b = f = 1$  and with the 2-element-mesh defined in Definition 4.2 such that both elements have the same polynomial degree  $p$ :

```

1 function sol = solution_const(p,eps,k)
2 %%% Solving the Differential equation:
3 %%%  $-\epsilon u'' + u' + u = 1$  with homogeneous Dirichlet
    boundary
4 %%% conditions ( $u(-1) = 0 = u(+1)$ ) for certain assumptions by
    using the
5 %%% hp-DG-FEM and the 2-element-mesh
6
7
8 %% Input:
9
10 % p          polynomial degree
11 % epsilon    diffusivity (real number between 0 and 1)
12 % k          mesh-parameter
13
14
15 %% Output:
16
17 % f is the piecewise continuous function which approximates the
    exact
18 % solution of the Differential equation with constant
    coefficients
19
20
21 %% Assumptions:
22

```

```

23 % The polynomial degree p holds for both elements.
24
25 % Constant parameters
26     a = @(x) 1;
27     % b = @(x) 1;
28     % f = @(x) 1;
29
30 % 2-element-mesh
31 x1 = -1;
32 x2 = 1-k*eps*p;
33 x3 = 1;
34
35
36 %% Computations of the element matrices and load vectors:
37
38 % Computation of the first element matrix
39 A1 = matrix_a(p) + matrix_b(p,x1,x2) + matrix_epsilon(p,eps,x1,
    x2) + matrix_A_11(p,eps,x1,x2);
40
41 % Computation of the second element matrix
42 A4 = matrix_a(p) + matrix_b(p,x2,x3) + matrix_epsilon(p,eps,x2,
    x3) + matrix_A_22(p,a,eps,x2,x3);
43
44 % Computation of the matrices describing the interior node part
    at x_1 of
45 % the two elements
46 A2 = matrix_A_12(p,p,a,eps,x1,x2,x3);
47 A3 = matrix_A_21(p,p,eps,x1,x2,x3);
48
49 % Computaion of the load vector corresponding to the first
    element
50 l1 = load_vector(p,x1,x2);
51
52 % Computaion of the load vector corresponding to the second
    element
53 l2 = load_vector(p,x2,x3);
54
55
56 %% Assembly of the global matrix and load vector:
57
58 % Assembly of the global stiffness Matrix
59 A = [A1,A2;A3,A4];
60 A = A';
61
62 % Assembly of the global load vector
63 l = [l1;l2];
64
65 %% Consideration of the homogeneous Dirichlet boundary
    conditions:
66
67 % Adjusting the elementwise load vectors
68 l(p+3) = [];
69 l(1) = [];
70

```



```

71 % Adjusting the global stiffness Matrix
72 A(p+3,:)=[];
73 A(:,p+3)=[];
74 A(1,:)=[];
75 A(:,1)=[];
76
77 A = sparse(A);
78
79
80 %% Solving the linear system:
81
82 % Computing the coefficients
83 coeff = A\1;
84
85
86 %% Building the solution by adding up the basis elements
   multiplied by the corresponding coefficient:
87
88 % Initialization
89 sol = @(x) 0;
90 % Shape-Functions
91 s1 =shape_funct(0);
92 s2 =shape_funct(1);
93 % Index for coefficient vector:
94 a_help = 1;
95
96 % Adding up:
97
98     % First element
99     % Width, midpoint and inverse reference map of the first
       element:
100     h1 = 1+x2;
101     m1 = (-1+x2)/2;
102     f1 = @(x) (x - m1)*2/h1;
103
104     syms x;
105     sol = sol + piecewise(x<-1,0,-1<x<x2,coeff(a_help)*s2(f1(x))
       ,x2<x,0);
106     a_help = a_help+1;
107     for i = 2:p
108         s3 = shape_funct(i);
109         syms x;
110         sol = sol + piecewise(x < -1 ,0,-1 < x < x2,coeff(a_help
       )*s3(f1(x)),x2<x,0);
111         a_help = a_help + 1;
112     end
113
114     % Second element
115     % Width, midpoint and inverse reference map of the
       second element:
116     h2 = 1-x2;
117     m2 = (1+x2)/2;
118     f2 = @(x) (x - m2)*2/h2;
119

```

```

120     sol = sol + piecewise(x < x2,0,x2<x<1,coeff(a_help)*s1(f2(x)
121     ),1<x,0);
121     a_help = a_help + 1;
122     for i = 2:p
123         s3 = shape_funct(i);
124         syms x;
125         sol = sol + piecewise(x < x2,0,x2<x<1,coeff(a_help)*s3(
126         f2(x)),1<x,0);
126         a_help = a_help + 1;
127     end
128
129
130 end

```

### Approximation Examples

In this subsection, we visualise some approximation results of our model problem with constant coefficients and  $\epsilon = 0.1$  by varying the element width  $h$  and the polynomial degree  $p$  in the hp-DG-FEM with a uniform mesh.

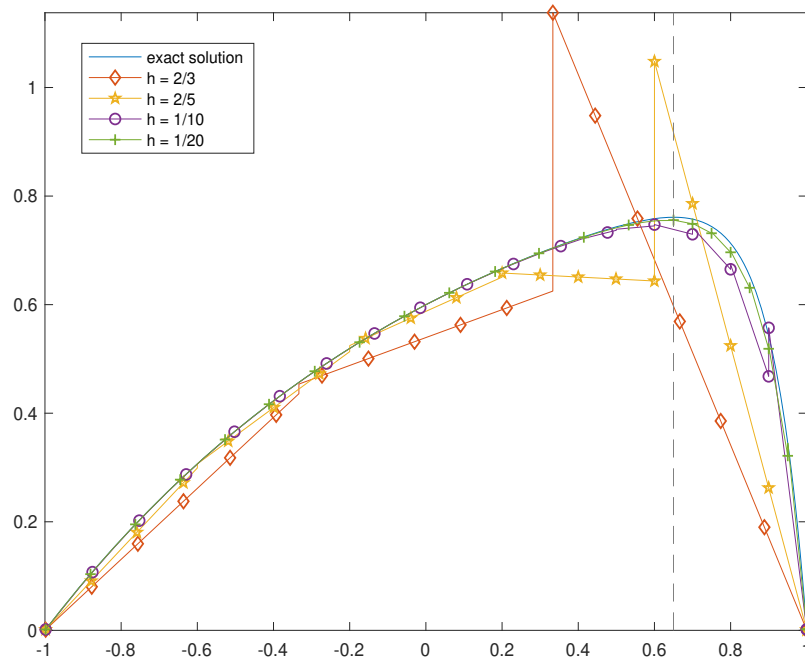


Figure 5.3: Approximation of the exact solution by decreasing the element width  $h$  with polynomial degree  $p = 1$ .

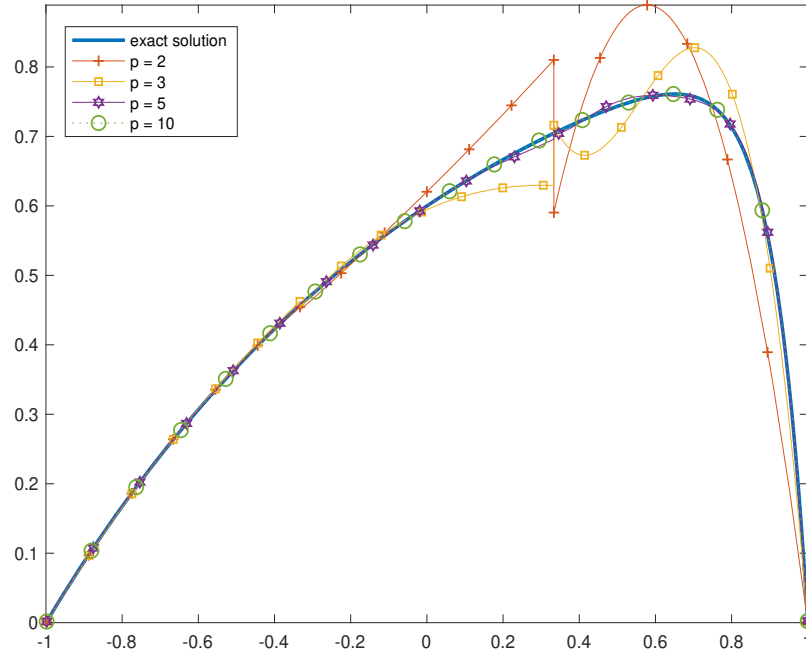


Figure 5.4: Approximation of the exact solution by increasing the polynomial degree  $p$  with element width  $h = \frac{2}{3}$ .

In this figures, we can clearly see that the refinement of the element width with small polynomial degrees as well as increasing polynomial degrees in a not strongly refined domain approximate the exact solution quite accurately.

### 5.2.3 Error Estimators

For the numerical experiment we analyse the behaviour of the DG-FEM in the  $L^2$ - and  $L^\infty$ -norm.

Therefore we use the exact solution of the model problem with constant coefficients which we generate with the function:

```

1 function f = exact_solution(epsilon)
2 %%% Generating the exact solution of the Differential equation:
3 %%% -\epsilon u'' + u' + u = 1 with homogeneous Dirichlet
   boundary
4 %%% conditions (u(-1) = 0 = u(+1))
5
6
7 %% Input:
8
```

```

9  % epsilon    diffusivity (real number between 0 and 1)
10
11
12  %% Output:
13
14  % f is the the exact solution of the Differential equation with
    constant
15  % coefficients
16
17
18  %% Initialization:
19
20  epsilon = vpa(epsilon);
21  lambda1 = 1/(2*epsilon)*(1+sqrt(1+4*epsilon));
22  lambda2 = 1/(2*epsilon)*(1-sqrt(1+4*epsilon));
23  C1 = exp(-lambda1)*(exp(-2*lambda2)-1)/(exp(-2*lambda1)-exp(-2*
    lambda2));
24  C2 = exp(-lambda2)*(-exp(-2*lambda1)+1)/(exp(-2*lambda1)-exp(-2*
    lambda2));
25
26
27  %% Solution:
28
29  f = @(x) C1*exp(lambda1*x)+C2*exp(lambda2*x)+1;
30
31
32  end

```

The next functions compute the  $L^2$ - and the  $L^\infty$ -error between the DG-FEM solution computed with the function "solution\_const" and the exact solution.

```

1  function L2E = L2_error(p,epsilon,k)
2  %%% Computation of the L2-error of the numerical solution
3
4
5  %% Input:
6
7  % p          the polynomial degree
8  % epsilon    diffusivity (real number between 0 and 1)
9  % k          mesh-parameter
10
11
12  %% Initialization:
13
14  % Node x1:
15  node_1 = 1-k*epsilon*p;
16  % Numerical solution:
17  num_sol = solution_const(p,epsilon,k);
18  % Exact solution:
19  exact_sol = exact_solution(epsilon);
20
21
22  %% Computation of the L2-error:
23

```

```

24 % Absolute difference between exact and numerical solution
25 diff = abs(num_sol-exact_sol);
26
27 % Square of the absolute difference
28 f = diff*diff;
29
30 % Integral of the square of the absolute difference
31 syms x;
32 int = vpaintegral(f,x,-1,1);
33
34 % square-root of the total integral:
35 L2E = sqrt(int);
36
37
38 end

1 function LinfE = L_inf_error(p,epsilon,k)
2 %%% Computation of the L\inf-error of the numerical solution
3
4
5 %% Input:
6
7 % p          the polynomial degree
8 % epsilon    diffusivity (real number between 0 and 1)
9 % k          mesh-parameter
10
11
12 %% Initialization:
13
14 % Node x1:
15 node_1 = 1-k*epsilon*p;
16 % Numerical solution:
17 num_sol = solution_const(p,epsilon,k);
18 % Exact solution:
19 exact_sol = exact_solution(epsilon);
20
21
22 %% Decomposition of numerical solution in elements:
23
24 % Elementwise decomposition of numerical solution:
25 pieces = children(num_sol);
26 % Numerical solution on first element:
27 num_sol_1 = pieces(2);
28 % Numerical solution on second element:
29 num_sol_2 = pieces(3);
30
31
32 %% Computation of the L\inf-error:
33
34 % Absolute difference between exact and numerical solution on
   first
35 % element:
36 diff_1 = abs(num_sol_1-exact_sol);
37 diff_1 = matlabFunction(diff_1);

```

```
38
39 % Absolute difference between exact and numerical solution on
   second
40 % element:
41 diff_2 = abs(num_sol_2-exact_sol);
42 diff_2 = matlabFunction(diff_2);
43
44
45 % Maximum of the difference between exact and numerical solution
   on first
46 % element:
47 loc1 = -fminbnd(diff_1,-1,node_1);
48 max1 = diff_1(loc1);
49
50 % Maximum of the difference between exact and numerical solution
   on first
51 % element:
52 loc2 = -fminbnd(diff_2,node_1,1);
53 max2 = diff_1(loc2);
54
55 % Maximum of the two elementwise maximums
56 LinfE = max([max1,max2]);
57
58 end
```

# Numerical Experiments

---

In this chapter, we copy the numerical experiments of [2]. The aim of these experiments is to illustrate the robust exponential convergence of the hp-DG-FEM, that is to demonstrate that the method is numerical stable independent of a very small  $\epsilon$ .

## 6.1 The Model Problem

We already mentioned in the section "Implementations" that we consider a model with constant coefficients of the form:

$$\begin{aligned} -\epsilon u'' + u' + u &= 1 & \text{in } \Omega = [-1, 1] \\ u(\pm 1) &= 0. \end{aligned} \tag{6.1}$$

We can generate the exact solution  $u_\epsilon$  of this problem with the function "exact\_solution":

$$u_\epsilon(x) = C_1 e^{\lambda_1 x} + C_2 e^{\lambda_2 x} + 1, \tag{6.2}$$

where:

$$\begin{aligned} \lambda_1 &= \frac{1}{2\epsilon}(1 + \sqrt{1 + 4\epsilon}) \\ \lambda_2 &= \frac{1}{2\epsilon}(1 - \sqrt{1 + 4\epsilon}) \\ C_1 &= e^{-\lambda_1} \frac{e^{-2\lambda_2} - 1}{e^{-2\lambda_1} - e^{-2\lambda_2}} \\ C_2 &= e^{-\lambda_2} \frac{1 - e^{-2\lambda_1}}{e^{-2\lambda_1} - e^{-2\lambda_2}}. \end{aligned}$$

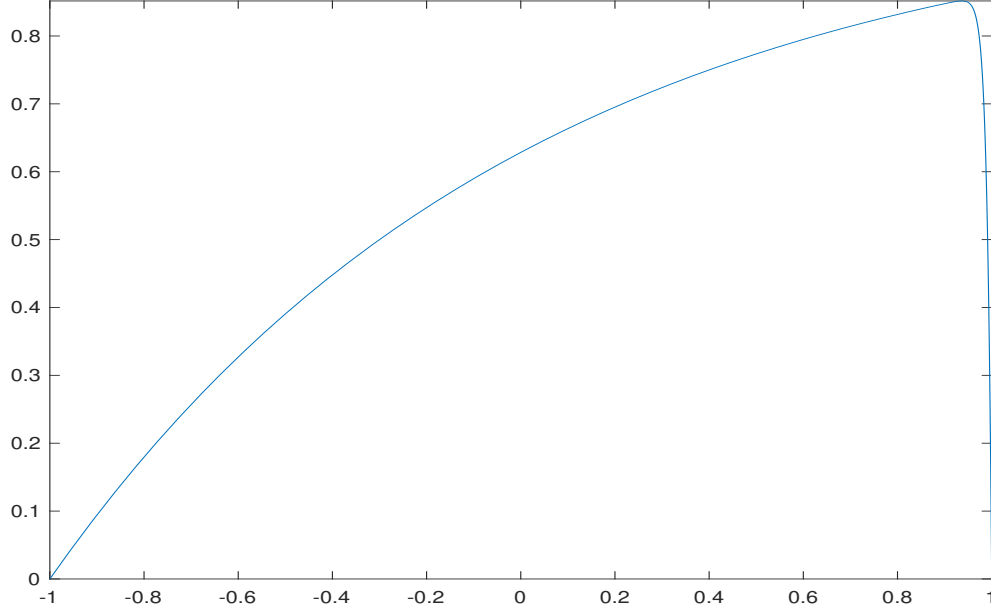


Figure 6.1: The exact solution  $u_\epsilon$  of the model problem (6.1) with  $\epsilon = 0.01$ .

## 6.2 Numerical Results

In Figure 6.1 we see already that the exact solution  $u_\epsilon$  has a boundary layer on the right boundary. As explained in Section 4.1 we use the 2-element-mesh defined in Definition 4.2 to resolve this boundary layer as in the function "solution\_const". We use the mesh-parameter  $\kappa = 1$ .

The results in Section 4.2 state the robust exponential convergence of the hp discontinuous Galerkin FEM in the  $\|\cdot\|_{DG}$ -norm on a 2-element-mesh. We illustrate the convergence in the  $L^2$ -norm computed by the function "L2\_error" in Figure 6.2 since a lemma stated in [2] declares:

**Lemma 6.1.** *For any FE-mesh  $\mathcal{T}$  and any function  $u \in \mathcal{S}_0^{\mathbf{p},0}(\mathcal{T}, \Omega)$  there holds the norm estimate:*

$$\|u\|_{L^2(\Omega)}^2 \leq \frac{1}{c_0} \|u\|_{DG}^2$$

*Proof.*  $\|u\|_{L^2(\Omega)}^2 \leq \sum_{i=1}^N \|u\|_{L^2(K_i)}^2 \leq \frac{1}{c_0} \sum_{i=1}^N \|\sqrt{c}u\|_{L^2(K_i)}^2 \leq \frac{1}{c_0} \|u\|_{DG}^2 \quad \square$

Furthermore, we illustrate the convergence also in the  $L^\infty$ -norm computed by the function "L\_inf\_error" in Figure 6.3.



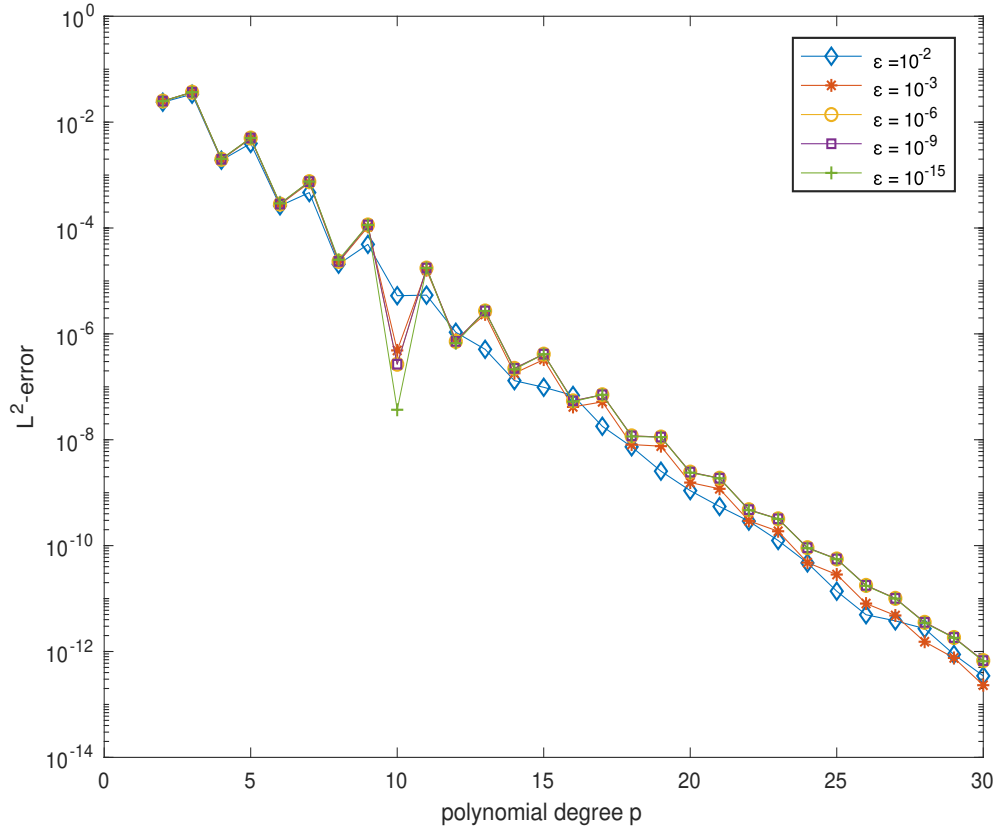


Figure 6.2:  $L^2$  performance  $\|u_{DG} - u_\epsilon\|_{L^2(\Omega)}$  of 2-element mesh  $\mathcal{T}_{\epsilon,1,p}$ .

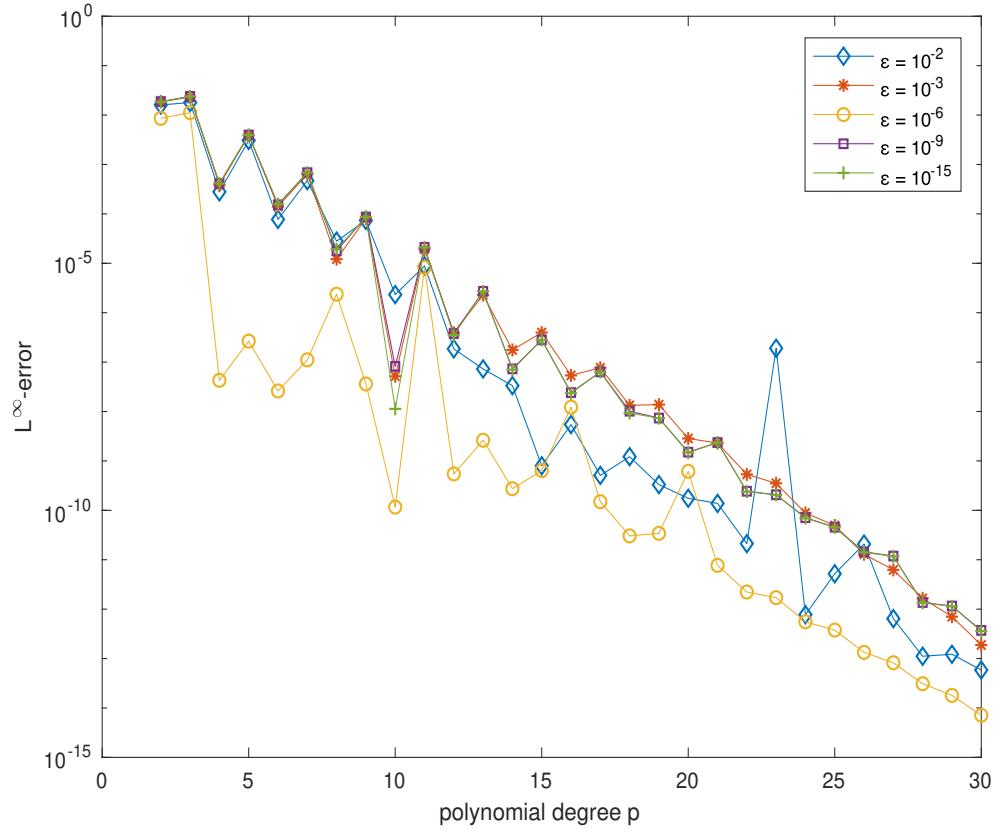


Figure 6.3:  $L^\infty$  performance  $\|u_{DG} - u_\epsilon\|_{L^\infty(\Omega)}$  of 2-element mesh  $\mathcal{T}_{\epsilon,1,p}$ .

# Stochastic Differential Equations and the hp-DG-FEM

---

In most simulations all input data are assumed to be perfectly known. In reality, this is only very rarely the case because all data are related with a certain uncertainty. By uncertainty we mean either intrinsic variability of physical quantities as random vibration or simply lack of knowledge about some physical behavior like experimentally obtained material properties. We can interpret these uncertainties often as randomness. Therefore, we can apply probability theory which requires considerable empirical information about the random quantities like for example probability distributions or their statistical moments.

To include such randomness in our problem we can describe the corresponding input coefficients of the differential equation by a stochastic probabilistic character. Consequently, the solution of such a stochastic differential equation appears also to be stochastic. The computation of these solutions is rather difficult. Hence, numerical methods as for example the Monte Carlo method have been developed in order to approximately compute quantities which describe the probabilistic behavior of the solution. Such quantities are the statistical moments as the "ensemble average" or the variance function.

In this chapter, we extend the convection-diffusion model problem in one space dimension to include one random coefficient. We compute statistical moments of the solution to this extended model equation, such as the "ensemble average", by using the Monte Carlo method. Furthermore, we state a convergence property of the Monte Carlo method and visualise it in a numerical experiment. Therefore, we follow the detailed explanations in [13, 15].

## 7.1 The Stochastic Problem Formulation

We extend our model problem (2.2) to include one coefficient which is depending on a random variable. Therefore, we firstly have to do some preparatory work and introduce some mathematical formulations.

Let  $(U, \mathcal{F}, P)$  denote a probability space, where  $U$  is the sample space, the set of all possible outcomes,  $\mathcal{F} \subset 2^U$  is the  $\sigma$ -algebra, a collection of subsets of  $U$ , and  $P$  is a probability measure on  $\mathcal{F}$ :

$$P : \mathcal{F} \longrightarrow [0, 1]$$

For  $\tilde{X} \in L^1(U)$  a real random variable in  $(U, \mathcal{F}, P)$  we denote its expected value as:

$$E[\tilde{X}] = \int_U X(\omega) dP(\omega) = \int_{\mathbb{R}} x d\mu(x),$$

where  $\mu$  is the distribution probability measure for  $\tilde{X}$  given by:

$$\mu(B) = P(X^{-1}(B))$$

for  $B$  a Borel set in  $\mathbb{R}$ .

We can define now a stochastic function as a jointly measurable function of the form:

$$\tilde{\phi}(x) = \phi(x, \omega) : \Omega \times U \longrightarrow \mathbb{R},$$

where  $\Omega = (-1, 1)$  is the domain of our model problem (2.2). Hence, we can state a new stochastic model problem derived from our initial model problem (2.2):

Find a stochastic function

$$\tilde{u}(x) : \Omega \times U \longrightarrow \mathbb{R} \tag{7.1}$$

such that almost surely (a.s.):

$$\begin{aligned} \mathcal{L}_\epsilon \tilde{u} &\equiv -\epsilon \tilde{u}'' + \tilde{a}(x) \tilde{u}' + b(x) \tilde{u} = f(x) & \text{in } \Omega \\ \tilde{u}(\pm 1) &= 0, \end{aligned} \tag{7.2}$$

where  $\tilde{a}(x) = a(x, \omega)$  and therefore also the solution  $\tilde{u}(x) = u(x, \omega)$  are stochastic functions. The expression almost surely means that it happens with probability 1.

We can take over nearly all assumptions made in (2.3) - (2.9) for the model problem (2.2) except of some adaptations for the stochastic coefficient  $\tilde{a}$ .

We assume that  $\tilde{a}$  is strongly measurable (i.e. a function that equals almost everywhere the limit of a sequence of measurable countably-valued functions) with the  $\sigma$ -algebra  $(B(\Omega) \otimes \mathcal{F})$  and bounded similar to the assumption (2.3):

$$\exists a_0 \in (0, +\infty) : P(\omega \in U : a(x, \omega) > a_0, \quad \forall x \in \overline{\Omega}) = 1, \quad (7.3)$$

where  $a_0$  is deterministic.

Further, we can reformulate the analytic regularity estimate in (2.7) to:

$\exists C_a, \gamma_a \in (0, +\infty)$  such that  $\forall n \in \mathbb{N}_0$  holds:

$$P(\omega \in U : a(\cdot, \omega) \in C^\infty(\overline{\Omega}) \wedge \|a(\cdot, \omega)^{(n)}\|_{L^\infty(\Omega)} \leq C_a \gamma_a n!) = 1, \quad (7.4)$$

where the constants  $C_a$  and  $\gamma_a$  are deterministic.

With these assumptions our model problem (7.2) is well-defined and has almost surely a unique solution.

Furthermore, we can state a variational discretization formulation similar to (3.3):

Find the stochastic function  $u_{DG}(\cdot, \omega) \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$  such that for all  $\omega \in U$ :

$$\begin{aligned} B_{DG}(\tilde{u}_{DG}, v) &:= \epsilon \sum_{i=1}^N \int_{K_i} \tilde{u}'_{DG} v' dx + \epsilon \sum_{i=1}^{N-1} \{ \langle \tilde{u}'_{DG} \rangle [v] - [\tilde{u}_{DG}] \langle v' \rangle \} (x_i) \\ &\quad + \sum_{i=1}^N \int_{K_i} \tilde{a} \tilde{u}'_{DG} v dx + \sum_{i=1}^{N-1} \tilde{a} [\tilde{u}_{DG}] v_+(x_i) \\ &\quad + \sum_{i=1}^N \int_{K_i} b \tilde{u}_{DG} v dx \\ &= \sum_{i=1}^N \int_{K_i} f v dx =: l_{DG}(v), \quad \forall v \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}}). \end{aligned} \quad (7.5)$$

By the same argumentation as in Chapter 3, we can say that this Galerkin discretization (7.5) is almost surely consistent and that by the generalisation of the Lax-Milgram Lemma (Lemma 3.1) there exists a unique solution  $u_{DG}(\cdot, \omega) \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$  of the formulation (7.5) for every  $\omega \in U$ .

For computing statistical quantities of  $\tilde{u}_{DG}$ , we have to prove that the solution  $\tilde{u}_{DG}$  is measurable with the  $\sigma$ -algebra  $(B(\Omega) \otimes \mathcal{F})$ .

Therefore, we show that there exists a Lipschitz continuous function from the stochastic coefficient  $\tilde{a}$  to the solution  $\tilde{u}_{DG}$ :

**Lemma 7.1.** *If  $u_1$  and  $u_2$  are hp-DG-FE solution to the variational problem (3.3) with the same coefficients  $b$  and  $f$  satisfying (2.4) - (2.9) but with coefficients  $a_1(x) = a(x, \omega_1) \in C^\infty(\bar{\Omega})$  and  $a_2(x) = a_2(x, \omega_2) \in C^\infty(\bar{\Omega})$ , respectively, where  $\omega_1, \omega_2 \in U$ , and if  $\tilde{a}(x) = a(x, \omega)$  satisfies the assumptions (7.3) and (7.4), then there exists a constant  $C > 0$  such that the following inequality holds:*

$$\|u_1 - u_2\|_{DG} \leq C \|a_1 - a_2\|_{C^\infty(\bar{\Omega})}$$

*Proof.* We can prove this inequality by following some concepts in the proof of the Strang Lemma stated in [17] (Theorem 8.3). First of all, we subtract the variational formulations (3.3) for  $u_1$  and  $u_2$  and we obtain for all  $v \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$ :

$$\begin{aligned} & \epsilon \sum_{i=1}^N \int_{K_i} (u'_1 - u'_2) v' dx + \epsilon \sum_{i=1}^{N-1} \{ \langle u'_1 - u'_2 \rangle [v] - [u_1 - u_2] \langle v' \rangle \} (x_i) \\ & + \sum_{i=1}^N \int_{K_i} a_1 u'_1 v dx - \sum_{i=1}^N \int_{K_i} a_2 u'_2 v dx + \sum_{i=1}^{N-1} a_1 [u_1] v_+(x_i) - \sum_{i=1}^{N-1} a_2 [u_2] v_+(x_i) \\ & + \sum_{i=1}^N \int_{K_i} b(u_1 - u_2) v dx \\ & = 0 \end{aligned}$$

We note, that since  $u_1$  and  $u_2$  are both in  $\mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$  also their difference  $u_1 - u_2$  is in  $\mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$ . The same holds for  $a_1$  and  $a_2$  and their corresponding space  $C^\infty(\bar{\Omega})$ .

Furthermore, we can expand the terms depending on the functions  $a_1$  and  $a_2$  in the following way:

$$\begin{aligned} & \epsilon \sum_{i=1}^N \int_{K_i} (u'_1 - u'_2) v' dx + \epsilon \sum_{i=1}^{N-1} \{ \langle u'_1 - u'_2 \rangle [v] - [u_1 - u_2] \langle v' \rangle \} (x_i) \\ & + \sum_{i=1}^N \int_{K_i} a_1 (u'_1 - u'_2) v dx + \sum_{i=1}^N \int_{K_i} (a_1 - a_2) u'_2 v dx + \sum_{i=1}^{N-1} a_1 [u_1 - u_2] v_+(x_i) \\ & + \sum_{i=1}^{N-1} (a_1 - a_2) [u_2] v_+(x_i) + \sum_{i=1}^N \int_{K_i} b(u_1 - u_2) v dx \\ & = 0 \end{aligned}$$

Then, we can rearrange the equality and define  $u_\Delta = u_1 - u_2$  to obtain the familiar bilinear form  $B_{DG}$ :

$$\begin{aligned}
B_{DG}(u_\Delta, v) &:= \epsilon \sum_{i=1}^N \int_{K_i} u'_\Delta v' dx + \epsilon \sum_{i=1}^{N-1} \{ \langle u'_\Delta \rangle [v] - [u_\Delta] \langle v' \rangle \} (x_i) \\
&+ \sum_{i=1}^N \int_{K_i} a_1 u'_\Delta v dx + \sum_{i=1}^{N-1} a_1 [u_\Delta] v_+(x_i) + \sum_{i=1}^N \int_{K_i} b u_\Delta v dx \\
&= - \sum_{i=1}^N \int_{K_i} (a_1 - a_2) u'_2 v dx - \sum_{i=1}^{N-1} (a_1 - a_2) [u_2] v_+(x_i)
\end{aligned}$$

By the coercivity of the bilinear form  $B_{DG}$  stated in Lemma 3.4, we can identify the error  $u_\Delta$  with the equality above and estimate the result by using the Hölder's Inequality:

$$\begin{aligned}
\|u_1 - u_2\|_{DG}^2 &= \|u_\Delta\|_{DG}^2 \stackrel{\text{Lemma 3.4}}{=} B_{DG}(u_\Delta, u_\Delta) \\
&= - \sum_{i=1}^N \int_{K_i} (a_1 - a_2) u'_2 u_\Delta dx - \sum_{i=1}^{N-1} (a_1 - a_2) [u_2] u_{\Delta+}(x_i) \\
&\leq \|a_1 - a_2\|_{C^\infty(\bar{\Omega})} \sum_{i=1}^N \int_{K_i} u'_2 u_\Delta dx + \|a_1 - a_2\|_{C^\infty(\bar{\Omega})} \cdot \underbrace{\sum_{i=1}^{N-1} [u_2] u_{\Delta+}(x_i)}_{\leq C_1 \|u_\Delta\|_{DG}} \\
&\stackrel{\text{Hölder}}{\leq} \|a_1 - a_2\|_{C^\infty(\bar{\Omega})} \left( \frac{1}{\sqrt{c_0}} \sum_{i=1}^N \|u'_2\|_{L^2(K_i)} \|\sqrt{c} u_\Delta\|_{L^2(K_i)} + C_1 \|u_\Delta\|_{DG} \right) \\
&\leq \|a_1 - a_2\|_{C^\infty(\bar{\Omega})} \left( \underbrace{\frac{1}{\sqrt{c_0}} \left( \sum_{i=1}^N \|u'_2\|_{L^2(K_i)}^2 \right)^{\frac{1}{2}}}_{\leq C_2} \underbrace{\left( \sum_{i=1}^N \|\sqrt{c} u_\Delta\|_{L^2(K_i)}^2 \right)^{\frac{1}{2}}}_{\leq \|u_\Delta\|_{DG}^2} + C_1 \|u_\Delta\|_{DG} \right) \\
&\leq C \|a_1 - a_2\|_{C^\infty(\bar{\Omega})} \|u_\Delta\|_{DG}
\end{aligned}$$

By dividing both side by  $\|u_\Delta\|_{DG}$ , we can conclude that the inequality in Lemma 7.1 holds.  $\square$

Therefore, we have proven that the function from the stochastic coefficient  $\tilde{a}$  to the solution  $\tilde{u}_{DG}$  is Lipschitz continuous.

Further, we assumed that  $\tilde{a}$  is measurable with the  $\sigma$ -algebra  $(B(\Omega) \otimes \mathcal{F})$ . Therefore, we can deduce that the hp-DG-FE solution  $\tilde{u}_{DG}$  is also measurable, as a composition of a measurable function and a Lipschitz continuous function.

## 7.2 Monte Carlo Method

The Monte Carlo method deals with generating random variates from probability density functions in order to estimate unknown parameters or general functions of unknown quantities such as the mean, variance and covariance. This technique relies on repeated random sampling to obtain numerical results.

Monte Carlo methods are an important tool for physical and mathematical problems as optimization, numerical integration or generating draws from a probability distribution.

In this work, we use the Monte Carlo method to compute quantities which describe the probabilistic behavior of the solution  $\tilde{u}_{DG}$  of the problem formulation (7.5).

These quantities are for example the statistical moments of  $\tilde{u}_{DG}$  such as the "ensemble average" (mean):

$$\bar{u}_{DG} := \mathbb{E}[\tilde{u}_{DG}] = \int_U u_{DG}(\cdot, \omega) dP(\omega) \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$$

and the variance function:

$$C := \mathbb{E}[(\tilde{u}_{DG} - \bar{u}_{DG})^2] \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}}) \otimes \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}}),$$

which measures how far a random quantity is spread out from its mean.

To compute these quantities, we firstly have to draw  $N \in \mathbb{N}$  independent functions  $\{a_1, \dots, a_N\}$  of the stochastic coefficient  $\tilde{a}$ . These draws are the base of the Monte Carlo Method.

Further, we can solve for each of these  $a_i$ 's the corresponding differential equation as in our model problem (2.2):

$$\begin{aligned} \mathcal{L}_\epsilon u_i &\equiv -\epsilon u_i'' + a_i(x)u_i' + b(x)u_i = f(x) \quad \text{in } \Omega \\ u_i(\pm 1) &= 0. \end{aligned}$$

to obtain associated solutions  $u_i$ .

In our case we solve these differential equations numerically by the hp-DG-FEM and obtain therefore  $u_i \in \mathcal{S}_0^{\mathbf{p},0}(\Omega, F_{\mathcal{T}})$ .

Out of the resulting sample solutions  $\{u_1, \dots, u_N\}$  we can estimate the "ensemble average" of  $\tilde{u}_{DG}$ :

$$\bar{u}_{DG}^N := \frac{1}{N} \sum_{i=1}^N u_i$$



and the variance function:

$$C_M^N := \frac{1}{N} \sum_{i=1}^N (u_i - \bar{u}_{DG}^M)^2.$$

### 7.2.1 Convergence

In this subsection, we analyse the convergence of the Monte Carlo estimate of the expected solution  $\bar{u}_{DG}$  to the variational problem (7.5).

The following result stated in [13, 15] is a bound on the expected statistical error  $\mathbb{E}[|\bar{u}_{DG} - \bar{u}_{DG}^N|_{DG}]$ :

**Lemma 7.2.** *For any  $N \in \mathbb{N}$  and for the solution  $\tilde{u}_{DG}$  of (7.5) holds:*

$$\mathbb{E}[|\bar{u}_{DG} - \bar{u}_{DG}^N|_{DG}] \leq (\mathbb{E}[|\tilde{u}_{DG}|_{DG}^2])^{\frac{1}{2}} N^{-\frac{1}{2}}$$

*Proof.* By using the Cauchy-Schwarz inequality, it holds:

$$\mathbb{E}[|\bar{u}_{DG} - \bar{u}_{DG}^N|_{DG}]^2 = \mathbb{E}[|\bar{u}_{DG} - \bar{u}_{DG}^N|_{DG} \cdot 1]^2 \leq \mathbb{E}[|\bar{u}_{DG} - \bar{u}_{DG}^N|_{DG}^2] \cdot \underbrace{\mathbb{E}[1^2]}_{=1}$$

and therefore, it is enough to show:

$$\mathbb{E}[|\bar{u}_{DG} - \bar{u}_{DG}^N|_{DG}^2]^{\frac{1}{2}} \leq \frac{1}{N^2} \mathbb{E}[|\tilde{u}_{DG}|_{DG}^2]^{\frac{1}{2}}.$$

With the independence of the identically distributed samples  $u_i$ , the fact that their laws are identical to the law of  $u_{DG}$  and the linearity of the expectation it follows:

$$\begin{aligned} \mathbb{E}[|\bar{u}_{DG} - \bar{u}_{DG}^N|_{DG}^2] &= \mathbb{E}\left[\left|\bar{u}_{DG} - \frac{1}{N} \sum_{i=1}^N u_i\right|_{DG}^2\right] \leq \frac{1}{N^2} \sum_{i=1}^N \mathbb{E}[|\bar{u}_{DG} - u_i|_{DG}^2] \\ &\stackrel{\text{i.i.d.}}{=} \frac{1}{N^2} \sum_{i=1}^N \mathbb{E}[|\bar{u}_{DG} - u_1|_{DG}^2] = \frac{1}{N} \mathbb{E}[|\bar{u}_{DG} - \tilde{u}_{DG}|_{DG}^2] \\ &= \frac{1}{N} \left( \underbrace{\mathbb{E}[|\bar{u}_{DG}|_{DG}^2]}_{=|\bar{u}_{DG}|_{DG}^2} - 2 \underbrace{\mathbb{E}[|\tilde{u}_{DG}|_{DG}]}_{=|\bar{u}_{DG}|_{DG}} \underbrace{\mathbb{E}[|\bar{u}_{DG}|_{DG}]}_{=|\bar{u}_{DG}|_{DG}} + \mathbb{E}[|\tilde{u}_{DG}|_{DG}^2] \right) \\ &= \frac{1}{N} \left( \mathbb{E}[|\tilde{u}_{DG}|_{DG}^2] - |\bar{u}_{DG}|_{DG}^2 \right) \leq \frac{1}{N} \mathbb{E}[|\tilde{u}_{DG}|_{DG}^2] \end{aligned}$$

By taking the root on both sides we have proved the lemma.  $\square$

Lemma 7.2 shows that Monte Carlo approximations have a convergence rate of  $\frac{1}{2}$  provided that the solution  $\tilde{u}_{DG}$  as a random function has finite second moments. That means the statistical error converges sublinearly.

Since  $\bar{u}_{DG}^N$  is computed by the hp-DG-FEM, we should choose appropriate element-width and polynomial degrees such that the corresponding discretization error does not affect the Monte Carlo convergence rate.

By Lemma 6.1 we can conclude further:

$$\mathbb{E}[\|\bar{u}_{DG} - \bar{u}_{DG}^N\|_{L^2(\Omega)}^2] \leq \frac{1}{c_0} \mathbb{E}[\|\bar{u}_{DG} - \bar{u}_{DG}^N\|_{DG}^2], \quad (7.6)$$

what is important for the numerical experiment in section 7.5.

### 7.3 Random Number Generator

At the kernel of the Monte Carlo method is random number generation. Therefore, we explain the principle and importance of this tool.

A random number generator produces a sequence of numbers which is random and therefore can not be predicted better than by a random chance. On the one hand, there are true random number generators (TRNG) which generate genuinely random numbers like rolling a die for certain times. On the other hand, there are pseudorandom number generators (PRNG) which create not truly random numbers because they are determined by a certain algorithm with a particular initial value like picking numbers from a precalculated table.

TRNG's are suitable for applications where it is important that the numbers are really unpredictable, such as data encryption and gambling. On the contrary, PRNG's are due to their speed and reproducibility suitable for applications where many numbers are required and where it is useful that the same sequence can be replayed easily such as simulation and modeling applications like the Monte Carlo simulations. Hence, we introduce the pseudorandom number generator more detailed in this section by following some explanations and examples as in [14].

#### 7.3.1 Mathematical Formulation

Before introducing the PRNG in a mathematical way, we have to make some definitions:

$P$  is the probability distribution on  $(\mathbb{R}, \mathcal{B})$ , where  $\mathcal{B}$  is the Borel algebra on the real line.

$A \subset \mathbb{R}$  is the non-empty subset contained in the support of  $P$  and containing its interior.

With these definitions, we call a function  $f : \mathbb{N} \rightarrow \mathbb{R}$  a pseudorandom number generator for  $P$  taking values in  $A$  if and only if:

- $f(\mathbb{N}) \subset A$
- $\forall E \in \mathcal{B} \forall \epsilon \in \mathbb{R}_+ \exists N \in \mathbb{N} \forall n \geq N : \left| \frac{\#\{i \in \{1, 2, \dots, n\} : f(i) \in E\}}{n} - P(E) \right| < \epsilon.$

That means all generated numbers have positive measure with respect to the probability distribution  $P$  and the generated pseudorandom numbers follow approximately the distribution  $P$ .

### 7.3.2 Basic Example

The so-called Linear Congruential Generator is the most common and oldest algorithm for generating pseudorandom numbers. The generator is defined by the following recurrence relation:

$$x_{n+1} = (a \cdot x_n + b) \bmod c,$$

where  $c \in \mathbb{R}_+$ ,  $0 < a < c$  and  $0 \leq b < c$ . Moreover, a initial value  $x_0$  is used which is often called seed. These data affects the statistical properties and the cycle width (the smallest number of steps until the value of the seed is reached again).

The Linear Congruential Generator is a very basic PRNG but it shows the idea of pseudorandomness. There exists a lot of different and more complicated methods for creating pseudorandom numbers supporting all possible distributions.

For our implementation in MATLAB, we use the build-in generator "rand" which produces uniformly distributed pseudorandom numbers.

## 7.4 Implementation

In this section, we present the implementation of the Monte Carlo method to approximate the "ensemble average"  $\bar{u}_{DG}^N$  and the variance function  $C_N^N$  of the numerical solution  $\tilde{u}_{DG}$ . To solve the differential equations corresponding to the pseudorandomly generated functions  $a_i$ , we use the function "solution" from Section 5.2.1 and compute hp-DG-FEM solutions  $u_i$ . Therefor, we choose a uniform mesh with  $n$  elements and an overall polynomial degree  $p$ .

We assume that the stochastic function  $\tilde{a}$  is of the form:

$$\tilde{a}(x) = a(x, \omega) = a_0(x) + \sigma \cdot a_1(x) \cdot Y(\omega),$$

where  $Y(\omega) \sim U(-1, 1)$  is a uniformly distributed random variable in the probability space  $(U, \mathcal{F}, P)$ . Furthermore,  $\sigma \in \mathbb{R}$  and the functions  $a_0(x)$  and  $a_1(x)$  on  $\Omega$  are chosen such that  $\tilde{a}$  satisfies the required properties (7.3) & (7.4).

```

1 function [ensemble_average,var_funct] = Monte_Carlo(N,p,n,
2     epsilon,sigma,a_0,a_1,b,f)
3     %%% Computing the "ensemble average" and the variance function
4     %%% by using the Monte Carlo method and the hp-DG-FEM with
5     uniform mesh
6     %%% and overall polynomial degree.
7
8     %% Input:
9
10    % N          the number of draws for the Monte Carlo Method
11    %            (natural number > 0)
12    % p          the overall polynomial degree for each element
13    %            (natural numbers > 0)
14    % n          the number of elements (natural number > 1)
15    % epsilon    diffusivity (real number between 0 and 1)
16    % sigma      parameter (small enough real number)
17    % a_0        first term of the stochastic function a = a_0+Y*a_1
18    %            (function handle)
19    % a_1        second term of the stochastic function a = a_0+Y*a_1
20    %            (function handle)
21    % b          losses/sources of the substance (function handle)
22    % f          external source term (function handle)
23
24    %% Output:
25
26    % ensemble_average    the first statistical moment (function
27    %                     handle)
28    % var_funct           the variance function (function handle)
29    % plot                of all solutions u_i to the corresponding
30    %                     differential equation, of the ensemble
31    %                     average and
32    %                     of the variance function
33
34    %% Generating vector of polynomial degrees:
35
36    % Initialization
37    vec_p = zeros(1,n);
38
39    % Polynomial degrees
40    for i = 1:n

```

```

39     vec_p(i) = p;
40 end
41
42
43 %% Generating nodes:
44
45 % Initialization
46 nod = zeros(1,n+1);
47 nod(1) = -1;
48 nod(end) = 1;
49     % Element-width
50     h = 2/n;
51
52 % Nodes
53 for i = 2:n
54     nod(i) = nod(i-1) + h;
55 end
56
57
58 %% Generating the pseudorandom numbers for the random variable Y
59     :
60 % Generating uniformly distributed numbers between -1 and 1
61     vec_a = -1 + 2*rand(1,N);
62
63
64 %% Generating the functions a_i:
65
66 % Initialization
67 a = sym('x',[1,N]);
68 syms x;
69 a_0 = a_0(x);
70 a_1 = a_1(x);
71
72 % Generating
73 for i = 1:N
74     a(i) = a_0 + sigma*vec_a(i)*a_1;
75 end
76
77
78 %% Solving differential equation by using hp-DG-FEM:
79
80 % Initialization
81 u = sym('x',[1,N]);
82
83 % Solutions
84 for i = 1:N
85     mfunct_a = matlabFunction(a(i));
86     u(i) = solution_MC(vec_p,nod,epsilon,mfunct_a,b,f);
87 end
88
89
90 %% Ploting the solutions u_i:
91

```

```

92 fplot(u(1),[-1,1]);
93 hold;
94
95 for i = 2:N
96     fplot(u(i),[-1,1]);
97 end
98
99
100 %% Computing the "ensemble average":
101
102 % Initialisation
103 ensemble_average = 0;
104
105 % Adding-up
106 for i = 1:N
107     ensemble_average = ensemble_average + u(i);
108 end
109
110 % Dividing by N
111 ensemble_average = ensemble_average/N;
112
113
114 %% Computing the variance function:
115
116 % Initialisation
117 var_funct = 0;
118
119 % Adding-up
120 for i = 1:N
121     var_funct = var_funct + (u(i)-ensemble_average)^2;
122 end
123
124 % Dividing by N
125 var_funct = var_funct/N;
126
127
128 %% Plotting the "ensemble average" and the variance function:
129
130 % We represent each solution u_i by the random generated number
    for the
131 % corresponding differential equation.
132 leg = zeros(1,N+2);
133 for i = 1:N
134     leg(i) = vec_a(i);
135 end
136 leg = string(leg);
137 leg(N+1) = 'ensemble average';
138 leg(N+2) = 'variance function';
139
140 % Plot "ensemble average"
141 fplot(ensemble_average,[-1,1], 'o-');
142 % Plot variance function
143 fplot(var_funct,[-1,1]);
144 % Legend

```

```
145 legend(leg);  
146  
147  
148 end
```

## 7.5 Numerical Experiments

In this section, we present some numerical experiments to visualise the Monte Carlo method and its convergence properties as shown in Lemma 7.2.

### 7.5.1 The Model Problem

We consider the following model problem:

$$\begin{aligned} -\epsilon \tilde{u}'' + \tilde{a}(x) \tilde{u}' + \tilde{u} &= 1 & \text{in } \Omega \\ \tilde{u}(\pm 1) &= 0, \end{aligned} \tag{7.7}$$

where  $\epsilon = 0.1$  and  $\tilde{a}(x) = 1 + \sigma \cdot Y(\omega) \cdot x$  with  $Y \sim U(-1, 1)$  and  $\sigma \in \mathbb{R}$  such that the assumptions (7.3) & (7.4) are satisfied.

For the following numerical experiments, we compute an accurate estimate of the mean of the exact solution  $\tilde{u}_\epsilon$  to problem (7.7) as a reference value. Therefor, we use the Gaussian quadrature rule, due to its good convergence properties (The Gaussian quadrature and its properties are explained detailed in [16]):

$$\mathbb{E}[\tilde{u}_\epsilon] = \bar{u}_\epsilon = \frac{1}{2} \int_{-1}^1 \tilde{u}_\epsilon(\cdot, y) dy \approx \frac{1}{2} \sum_{i=1}^n \omega_i \cdot u_{DG}(\cdot, y_i),$$

where  $\omega_i > 0$  are the Gaussian-weights,  $y_i \in (-1, 1)$  are the Gaussian-nodes and  $n \in \mathbb{N}$  is the number of points. Furthermore,  $u_{DG}(\cdot, y_i)$  is the approximated hp-DG-FEM solution to the problem:

$$\begin{aligned} -0.1 \cdot u'' + (1 + \sigma \cdot y_i \cdot x) \cdot u' + u &= 1 \\ u(\pm 1) &= 0, \end{aligned}$$

where the Gaussian-point  $y_i$  is an evaluation of  $Y(\omega)$ .

We use with the hp-DG-FEM and the Gaussian quadrature two approximations to compute the expected solution  $\bar{u}_\epsilon$ . But for well chosen parameters, that means a large number  $n$  of Gaussian-points and suitable polynomial degree  $p$  and element-width  $h$ , we can estimate this reference value so accurately to utilise it for the upcoming numerical experiments.

Since we work on the interval  $(-1, 1)$ , we can determine the nodes and weights of the Gauss-Legendre quadrature by using the  $n$ -th Legendre polynomial  $L_n$ :

$$\begin{aligned} y_i &= \text{"the } i\text{-th root of } L_n\text{"} \\ \omega_i &= \frac{2}{(1 - y_i^2)(L'_n(y_i))^2}. \end{aligned}$$



Hence, we can compute the expected solution  $\bar{u}_\epsilon$  of (7.7) in MATLAB accurately by determining the weights and nodes of the 10-point Gaussian quadrature and by using the implemented function "solution" with the polynomial degree  $p = 10$  and the defined 2-element-mesh (Definition 4.2) to calculate exact  $u_{DG}(\cdot, y_i)$  for each  $i \in \{1, \dots, N\}$ .

### 7.5.2 Approximation of the Mean and the Variance Function

We can use the Monte Carlo method to approximate the mean and the variance as explained in the section 7.2. To visualise, we use the implemented function "Monte\_Carlo" for the model problem (7.7) with  $\sigma = 0.95$ . For each sample solution  $u_i$ , we solved the corresponding differential equation by the hp-DG-FEM with 4 uniform elements and a common polynomial degree  $p = 6$  using the implemented function "solution".

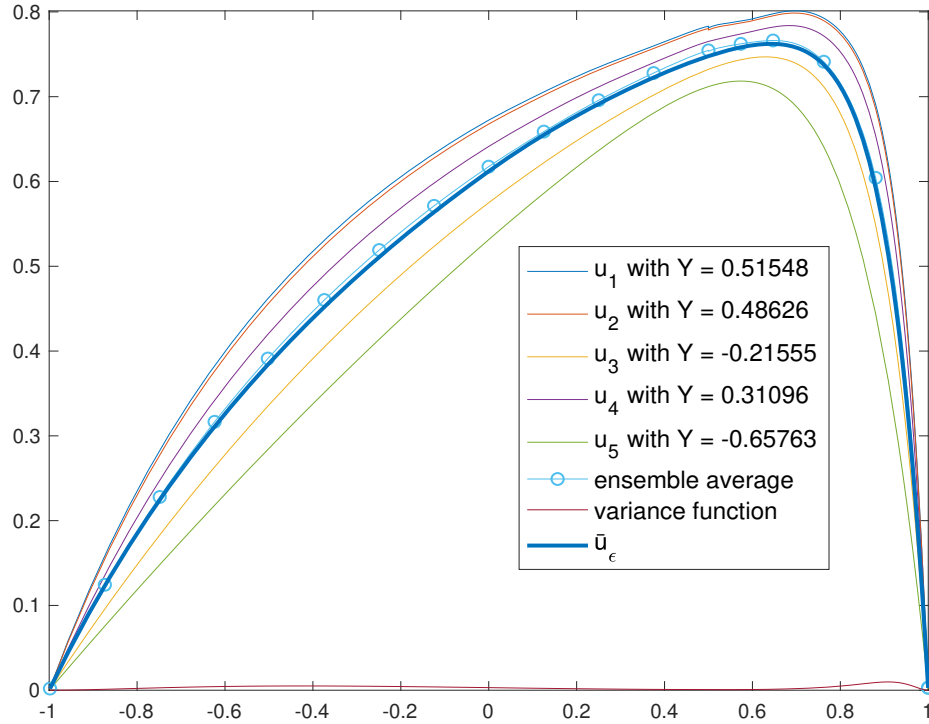


Figure 7.1: Monte Carlo approximation of the variance function and of the solution's mean  $\bar{u}_\epsilon$  to the model problem (7.7) for  $\sigma = 0.95$  with  $N = 5$  samples.

### 7.5.3 Convergence Analysis

For visualising the convergence rate of the Monte Carlo method in the  $\|\cdot\|_{DG}$ -norm proved in Lemma 7.2, we compute  $\mathbb{E}[\|\bar{u}_\epsilon - \bar{u}_{DG}^N\|_{L^2(\Omega)}]$  for the model problem (7.7), with various values of  $\sigma$ , in the  $L^2$ -Norm by utilising the estimate (7.6).

For approximating the expected  $L^2$ -error, we compute an arithmetic mean of order 10:

$$\mathbb{E}[\|\bar{u}_\epsilon - \bar{u}_{DG}^N\|_{L^2(\Omega)}] \approx \frac{1}{10} \sum_{i=1}^{10} \|\bar{u}_\epsilon - \bar{u}_{DG}^N\|_{L^2(\Omega)}.$$

We compute the  $u_i$ 's in the term  $\bar{u}_{DG}^N$  by using the hp-DG-FEM with the 2-element-mesh defined in Definition 4.2 and an overall polynomial degree  $p = 6$  such that the discretization error is small enough to not affect the Monte Carlo convergence rate. Therefor, we had to modify the function "Monte\_Carlo" slightly. The statistical error is computed by the adjusted function "L2\_error" where the number of samples  $N \in \{1, 2, \dots, 30\}$  varies in the Monte Carlo method.

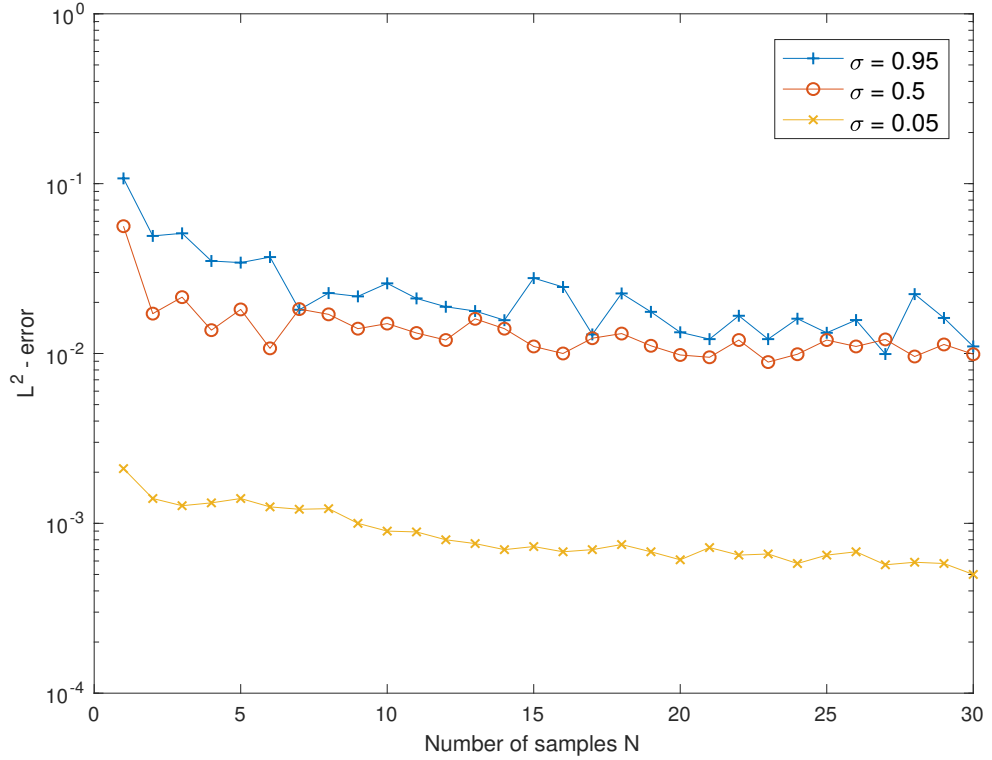


Figure 7.2:  $L^2$  performance of the Monte Carlo method.

Using the data illustrated in Figure 7.2, we can estimate the theoretical convergence rate of  $\frac{1}{2}$  stated and proven in Lemma 7.2.

We compute the following fitted values to approximate the convergence rates  $r_\sigma$  corresponding to the chosen  $\sigma$ :

$$r_{0.95} \approx 0.6050$$

$$r_{0.5} \approx 0.5104$$

$$r_{0.05} \approx 0.4219.$$

These values affirm the convergence properties of the Monte Carlo method independent of the chosen  $\sigma$ .

# Bibliography

- [1] Ch. Schwab, Lecture notes of the course "Numerical Methods for Partial Differential Equations", Chapter 1, Seminar for Applied Mathematics, ETH Zürich, October 2019.
- [2] T. P. Wihler and Ch Schwab, "Robust exponential convergence of the hp discontinuous Galerkin FEM for convection-diffusion problems in one space dimension", East-West Journal of Numerical Mathematics, 8(1), 57-70.
- [3] T. P. Wihler, "Stabilisierte hp-DG-FEM für Advektions-Diffusions-Probleme", Diplomarbeit, Seminar for Applied Mathematics, ETH Zürich, 1999.
- [4] A. Quarteroni, R. Sacco and F. Saleri, "Numerical Mathematics", Vol. 37. Springer Science & Business Media, 2010.
- [5] Ch. Schwab, "p-and hp-finite element methods: Theory and applications in solid and fluid mechanics", Oxford University Press, 1998.
- [6] J.M. Melenk and Ch. Schwab, "An hp finite element method for convection-diffusion problems in one dimension", IMA Journal of Numerical Analysis, 19(3), 425-453.
- [7] J.M. Melenk and Ch. Schwab, "An hp finite element method for convection-diffusion problems", Research Report No. 97-05, Seminar für angewandte Mathematik, ETH Zürich, February 1997.
- [8] M. Protter and H. Weinberger, "Maximum Principles in Differential Equations", Springer Verlag Heidelberg-New York, 1984.
- [9] M. Struwe, Lecture notes of the course "Funktionalanalysis 1 und 2", ETH Zürich, 2013/2014.
- [10] Ch. Schwab and M. Suri, "The p and hp version of the finite element method for problems with boundary layers", Math. Comp. (1996) 65, 1403-1429.
- [11] J.M. Melenk and Ch. Schwab, "The hp-streamline diffusion finite element method for convection dominated problems in one space dimension", East-West J.Numer. Math. (1999) 7, No. 1, 31-60.
- [12] Béatrice Rivière, "Discontinuous Galerkin Methods for Solving Elliptic and Parabolic Equations: Theory and Implementation", Society for Industrial and Applied Mathematics Philadelphia, 2008, 3-17.

- [13] A. Cohen, R. DeVore, Ch. Schwab, "Convergence Rates of Best N-term Galerkin Approximations for a Class of Elliptic sPDEs", *Found Comput Math* (2010) 10: 615–646, 2010.
- [14] J. E. Gentle, "Random Number Generation and Monte Carlo Methods", Springer, Statistics and Computing, 2003.
- [15] A. Barth, Ch. Schwab and N. Zollinger, "Multi-Level Monte Carlo Finite Element method for elliptic PDE's with stochastic coefficients", *Seminar für Angewandte Mathematik, Eidgenössische Technische Hochschule*, 2010.
- [16] Ch. Schwab, Lecture notes of the course "Numerische Mathematik I, ETH BSc MATH", ETH Zürich, 2018.
- [17] O. Steinbach, "Numerical Approximation Methods for Elliptic Boundary Value Problems", Springer Science & Business Media, 2008.



Eidgenössische Technische Hochschule Zürich  
Swiss Federal Institute of Technology Zurich

## Eigenständigkeitserklärung

Die unterzeichnete Eigenständigkeitserklärung ist Bestandteil jeder während des Studiums verfassten Semester-, Bachelor- und Master-Arbeit oder anderen Abschlussarbeit (auch der jeweils elektronischen Version).

Die Dozentinnen und Dozenten können auch für andere bei ihnen verfasste schriftliche Arbeiten eine Eigenständigkeitserklärung verlangen.

Ich bestätige, die vorliegende Arbeit selbständig und in eigenen Worten verfasst zu haben. Davon ausgenommen sind sprachliche und inhaltliche Korrekturvorschläge durch die Betreuer und Betreuerinnen der Arbeit.

**Titel der Arbeit** (in Druckschrift):

**Verfasst von** (in Druckschrift):

*Bei Gruppenarbeiten sind die Namen aller Verfasserinnen und Verfasser erforderlich.*

**Name(n):**

**Vorname(n):**


Ich bestätige mit meiner Unterschrift:

- Ich habe keine im Merkblatt „[Zitier-Knigge](#)“ beschriebene Form des Plagiats begangen.
- Ich habe alle Methoden, Daten und Arbeitsabläufe wahrheitsgetreu dokumentiert.
- Ich habe keine Daten manipuliert.
- Ich habe alle Personen erwähnt, welche die Arbeit wesentlich unterstützt haben.

Ich nehme zur Kenntnis, dass die Arbeit mit elektronischen Hilfsmitteln auf Plagiate überprüft werden kann.

**Ort, Datum**

**Unterschrift(en)**



*Bei Gruppenarbeiten sind die Namen aller Verfasserinnen und Verfasser erforderlich. Durch die Unterschriften bürgen sie gemeinsam für den gesamten Inhalt dieser schriftlichen Arbeit.*