SISSA
Scuola
Internazionale
Superiore di
Studi Avanzati

Mathematics Area - PhD course in
Mathematical Analysis, Modelling, and Applications

HDG methods and data-driven techniques for the van Roosbroeck model and its applications

Candidate:
Stefano Piani

Advisors:
Prof. Luca Heltai
Dr. Wenyu Lei

Academic Year 2021-22

Il presente lavoro costituisce la tesi presentata da Stefano Piani, sotto la direzione del Prof. Luca Heltai e del Dott. Wenyu Lei al fine di ottenere l'attestato di ricerca post-universitaria *Doctor Philosophiæ* presso la SISSA, Curriculum in Analisi Matematica, Modelli e Applicazioni. Ai sensi dell'art. 1, comma 4, dello Statuto della SISSA pubblicato sulla G.U. no. 36 del 13.02.2012, il predetto attestato è equipollente al titolo di *Dottore di Ricerca in Matematica*.

Trieste, Anno Accademico 2021/2022

Abstract

Noninvasive estimation of doping inhomogeneities in semiconductors is relevant for many industrial applications. The goal is to estimate experimentally the unknown doping profile of a semiconductor by means of reproducible, indirect and non–destructive measurements.

A number of technologies (such as LBIC, EBIC and LPS) have been developed which allow the indirect detection of doping variations via photovoltaic effects. The idea is to illuminate the sample at several positions while measuring the resulting voltage drop or current at the contacts. These technologies lead to inverse problems for which we still do not have a complete theoretical framework.

In this thesis, we present three different data-driven approaches based on least squares, multilayer perceptrons, and residual neural networks. We compare the three strategies after having optimized the relevant hyperparameters and we measure the robustness of our approaches with respect to noise.

The methods are trained on synthetic data sets (pairs of discrete doping profiles and corresponding photovoltage signals at different illumination positions) which are generated by a numerical solution of the forward problem using a physics-preserving finite volume method stabilized using the Scharfetter–Gummel scheme.

In view of the need of generating larger datasets for trainings, we study the possibility to apply high-order Discontinuous Galerkin methods to the forward problem, preserving the stability properties of the Scharfetter–Gummel scheme.

We prove that the Hybridizable Discontinuous Galerkin methods (HDG), a family of high-order DG methods, are equivalent to the Scharfetter–Gummel scheme on uniform unidimensional grids for a specific choice of the HDG stabilization parameter.

This result is generalized to two and three dimensions using an approach based on weighted scalar products, and on local Slotboom changes of variables. We show that the proposed numerical scheme is well-posed, and numerically validate that it has the same properties of classical HDG methods, including optimal convergence and superconvergence of postprocessed solutions. For polynomial degree zero, dimension one, and vanishing HDG stabilization parameter, W-HDG coincides with the Scharfetter-Gummel stabilized finite volume scheme (i.e., it produces the same system matrix).

Contents

Introduction

In the last decades, semiconductors and their applications have become pervasive in many sectors of our society. Semiconductors are an essential component of electronic devices and are employed in several different fields like communications, computing, health-care, military systems, transportation, energy, and countless others.

Developing noninvasive techniques to estimate in a non–destructive way inhomogeneities in semiconductors is therefore relevant for many applications. This is an open problem whose solution requires contributions from many disciplines: rigorous mathematical models to describe the physical phenomena occurring inside the semiconductor, efficient numerical schemes that are able to deal with intrinsic difficulties that these models exhibit, a solid mathematical theory to frame the inverse problem that rises when trying to estimate a physical quantity starting from an indirect measure, and computational algorithms to solve such problems.

The fundamental model to describe semiconductors has been developed in the early fifties of the twentieth century by van Roosbroeck (see [47]); it a nonlinear system of three equations that links together the electric potential with the movements of electrons inside the crystal and with the densities of holes that the free electrons leave inside the crystal lattice. The *van Roosbroeck* system is used to model many physical devices ranging from diodes and transistors to LEDs, solar cells, and lasers. We briefly describe the main properties of this system in Chapter 1. For a more detailed exposition that involves also Fermi-Dirac statistics, we refer to [21] and [44].

In Chapter 1 we describe also a tailored scheme to solve numerically the equations of the van Roosbroeck system. Obtaining good numerical simulations for a semiconductor device is a challenging task: the difficulty lies in the strong nonlinearities, the drift dominated nature of the underlying physics, and the presence of stiff boundary layers. Moreover, the resulting solutions must ensure consistency with several thermodynamical principles, guaranteeing the nonnegativity of the carrier densities. Many classical schemes show unstable behaviors when applied to such problems (see [21]); the first unidimensional finite difference scheme that was able to overcome these difficulties is due to Scharfetter and Gummel [43]. Later on, Fichtner, Rose, and Bank [22] extended it to higher dimensions as a finite volume scheme. One of the biggest drawback of this method is its low order nature; nevertheless, it is still a key tool for the simulation of semiconductor devices, thanks to its positivity preserving property and its capability of capturing boundary layers and other important features of the simulation.

In Chapter 2 we introduce the main topic of this thesis: the estimation of the doping concentration inside semiconductor crystals by the systematically generation of electron-hole pairs through localized electromagnetic radiation. Several different technologies use the described photovoltage mechanism to analyze doping inhomogeneities. They are

classified according to either the type of excitation used to induce the local generation of electron-hole pairs, or the contact placement to measure the generated current, or how the collected signal is related to the doping variation. Electron Beam Induced Current (EBIC) [49], Laser Beam Induced Current (LBIC) [7], scanning photovoltage (SPV) [26], and Lateral Photovoltage Scanning (LPS) [34, 28] are some of such photovoltaic technologies, using as electromagnetic source either localized electron beams (EBIC) or laser beams (LBIC, SPV, and LPS).

A typical outcome of these techniques is an image where the intensity of each pixel is proportional to the total current signal induced by a beam shone on the pixel location.

The LPS method, proposed in [34, 20], is especially useful in the context of crystal growth, where it is virtually impossible to predict the quality (i.e., the symmetry) of a semiconductor crystal *during* its growth in a furnace. During the growth process, thermal fluctuations near the the solid-liquid interface introduce local fluctuations (or striations) in the doping profile. LPS detects such doping inhomogeneities non-invasively at wafer-scale and room temperature.

Mathematically speaking, all of the discussed technologies result into inverse problems. The forward problem assumes that we know the doping profile and a set of laser spot positions. We then want to know the corresponding (laser spot dependent) photovoltage signals at the contacts of the sample. The inverse problem, on the other hand, assumes we have measured photovoltage signals at the contacts for a laser scan across the sample and we want to reconstruct the doping profile at least in the probing region (ideally in the whole domain).

Inverse (PDE) problems have been studied for a long time from an analytical and a numerical point of view. We refer to the general overviews [48, 27] and the references therein. Burger, Markowich, and others analyzed inverse semiconductor problems [29, 12]. Since the analysis is challenging, they often consider linearized/unipolar settings, no recombination/generation, only small external biases, linear diffusion as well as standard Dirichlet-Neumann boundary conditions. In [12], they discuss identifiability of the doping profile from capacitance, reduced current-voltage or laser beam induced current (LBIC) data. As for the doping reconstruction, previous numerical methods relied on optimization [40], the level set [29] or the Landweber–Kaczmarz methods [12]. Especially, the latter proved to be unstable and costly. For this reason we instead focus on three different data-driven approaches to solve the LPS problem.

Any data-driven technique heavily relies on the amount and quality of training data. Since, for example, growing crystals is exceptionally costly, we are not able to generate large real-life datasets. For this reason we generate synthetic data (measured signals and corresponding doping profiles) via a fast and efficient implementation of the Scharfetter–Gummel scheme.

Unfortunately, also generating a synthetic dataset is a demanding task, especially for the computational cost needed to produce a suitable number of samples. This is a common hurdle that LPS problems share with many other physical problems where solving multiple times a nonlinear PDE is required to generate the training dataset. This shows the importance of having high order methods at our disposal to reduce the

computational burden of each simulation and allow the creation of bigger datasets.

Finding a high–order numerical scheme with the same stability properties of the Scharfettel–Gummel scheme is still an open problem. Our contribution in this direction is described in Chapters 3 and 4, where we discuss the possibility to employ a particular class of Discontinuous Galerkin (DG) methods to obtain good numerical simulations of the van Roosbroeck system.

At the beginning of this century it has become more and more common to address convection-dominated problems using discontinuous Galerkin methods (DG methods). One of the main advantages of these methods is their ability to handle adaptive algorithms, working with meshes with hanging nodes and approximation of varying polynomial degrees. Unfortunately, when compared with the standard continuous finite elements, DG methods were often criticized for using too many globally coupled degrees of freedom.

In Chapter 3 we describe a particular class of Discontinuous Galerkin methods, the Hybridizable Discontinuous Galerkin Methods, originally introduced in [17]. The key element of these methods is that they exploit the classic technique of static condensation to reduce the total number of globally coupled degrees of freedom. In the same chapter we show a theoretical relationship that connects these methods with the Scharfetter–Gummel scheme.

While these results are interesting from a theoretical point of view, they are limited to onedimensional domains and their application to higher dimensions is still unclear. Therefore, in Chapter 4 we introduce an adapted version of the HDG method, tailored for drift–diffusion equations. We refer to this new scheme as *Weighted HDG* (W-HDG). The solution of the Scarfetter–Gummel scheme can be constructed as the limit of the solutions of the W-HDG method when the stabilization parameter goes to 0. We conclude the chapter by showing some numerical results where we compare the solutions generated by the three different methods (Scharfettel–Gummel finite volumes, standard HDG and W-HDG) on a p-i-n junction, i.e., a device with an undoped intrinsic semiconductor region between a p-type semiconductor and an n-type semiconductor region.

Chapter 1

Semiconductor device simulations

1.1 Van Roosbroeck Model

A semiconductor material is characterized by a concentration of conduction electrons that is greater than the one typically found in a insulator (which is around 10^3cm^{-3}) but is less than the one found in a common conductor (of the order of magnitude 10^{22}cm^{-3}). For example, semiconductor silicon at room temperature has a concentration that is approximately equal to 10^{10}cm^{-3}.

Moreover, if the temperature raises or if an electric field is applied, many valence electrons become conduction electrons, greatly increasing the previous number. In the following, we will indicate the spatial function that describes the density of the conduction electrons as n.

When an electron changes its state (from a valence to a conduction electron), it leaves a "hole" in the lattice of the crystal of the semiconductor. We will indicate the density of such holes as p. It is reasonable to consider these holes as positively charged carriers. Indeed, their movement is a way to model the fact that holes continuously appear and disappear because of the change of status of electrons.

A third source of charge may be present inside a semiconductor: it is the so called *doping*. This is an artificially controlled implantation of impurity atoms whose charge can be either positive or negative. We will use the symbol c to indicate the density of such impurities. It is worth noting that, while n and p are always positive, c can be positive or negative (depending if the impurities have positive or negative sign in the particular point where c is evaluated). We therefore rewrite the doping function as a difference of two positive spatial functions

$$c(\mathbf{x}) \stackrel{\text{def}}{=} N_D(\mathbf{x}) - N_A(\mathbf{x}),$$

where N_D represents the density of electrically active donor atoms (like phosphorus or arsenic) and N_A is the density of acceptor atoms (usually, boron or gallium). Moreover, doping charges are fixed while electrons and holes can move. Such movement induce currents in the material, that we indicate respectively as $\mathbf{J}_n$ (the electron current) and $\mathbf{J}_p$ (the hole current). These quantities describe the rate of flow of electrons (or, respectively, holes) past a point. Since electrons and holes are the only moving charges, the total current $\mathbf{J}$ of the device is simply $\mathbf{J}_n + \mathbf{J}_p$.

The last element that we need to introduce the stationary van Roosbroeck model is the the electric potential V together with its associated electric field

$$\mathbf{E} \stackrel{\text{def}}{=} -\nabla V,$$

and with the electric displacement $\mathbf{D}$, for which we assume that the following relation holds

$$\mathbf{D} = \varepsilon \mathbf{E}, \tag{1.1}$$

where ε is an $L^\infty(\Omega)$ function, i.e., we assume that our materials are linear, homogeneous, isotropic with "instantaneous" response to changes in the electric field. In a more general setup, the permittivity can be a thermodynamic state function that depends on several state variables like frequency, magnitude, and direction of the applied field, adding some unnecessary complications to our model.

Using Gauss's law for static fields, we have

$$\nabla \cdot \mathbf{D} = \rho,$$

where ρ is the density of charges. Since our charges come from the doping, the conduction electrons, and the holes and taking into account assumption (1.1), we can rewrite the previous equation as

$$\nabla \cdot (\varepsilon \mathbf{E}) = q(c - n + p),$$

where q is the elementary charge, i.e., the electric charge of a single proton or, equivalently, the magnitude of the negative electric charge of a single electron, which has charge $-q$.

Finally, we want to enforce the charge continuity equation (for the stationary case), i.e.,

$$\nabla \cdot \mathbf{J} = 0,$$

or, equivalently,

$$\nabla \cdot \mathbf{J}_n = -\nabla \cdot \mathbf{J}_p.$$

Because of the previous equation, we can introduce a new function $H(n, p)$, the so-called *generation-recombination term*, such that

$$\begin{cases} \nabla \cdot \mathbf{J}_n = qH(n, p), \\ \nabla \cdot \mathbf{J}_p = -qH(n, p). \end{cases}$$

Indeed, a "generation" occurs when a valence electron becomes a conduction electron and leaves a hole. Recombination, instead, is the opposite phenomenon and occurs when a conduction electron neutralizes a hole becoming a valence electron. When the semiconductor device is in thermal equilibrium, generations and recombinations appear at the same rate, resulting in a dynamic balance. External perturbations (like an external electric field), however, can lead the device away from this ideal situation. From a physical point of view, the role of H is to describe how far the device is from equilibrium; $H > 0$ indicates a situation where new electric carriers are generated and, vice versa, H is negative when recombination is the dominating effect.

Many different models have been developed during the second half of the 20th century, each representing the effect of a different energy transition process: the most common models used in the literature are the Shockley-Read-Hall model, the Auger model, and the Avalanche model. See [35, Chapter 2] for an in-depth description of these and other models. The overall representation for H is usually obtained by linearly superimposing these models, neglecting the possibly occurring nonlinear interactions.

What we still miss is some relationships between $\mathbf{J}_n$ and n and, in the same way, between $\mathbf{J}_p$ and p. In particular, we want to able to express $\mathbf{J}_n$ and $\mathbf{J}_p$ as functions of n, p and $\mathbf{E}$. In the following, we only obtain an expression for $\mathbf{J}_n$ but, for $\mathbf{J}_p$, an analogous argument holds (beside the signs). Our assumption is that $\mathbf{J}_n$ is a superimposition of two effects: one due to diffusion of electrons which generates a current $\mathbf{J}_n^{\text{diff}}$ proportional to the gradient of n, and another one due to the drift of the electrons forced by the electric field; this last effect generates a current $\mathbf{J}_n^{\text{drift}}$ which is proportional to both the electric field and to the density of electrons. Therefore, we define

$$\mathbf{J}_n^{\text{diff}} \stackrel{\text{def}}{=} qD_n\nabla n,$$

$$\mathbf{J}_n^{\text{drift}} \stackrel{\text{def}}{=} q\mu_n n\mathbf{E},$$

where μ_n and D_n are mobility and the diffusion coefficients, respectively: both of them are $L^\infty(\Omega)$ functions that depend on the material.

For many applications, μ_n can be approximate by a piecewise constant function, i.e. it is reasonable to assume that μ_n and μ_p only change when crossing a junction between different materials. For the applications for which it is important to take into account the scattering of charge carriers with doping, more detailed models have been developed. One of the most widely used is the*Arora mobility model* (see [20]): if we denote by T the spatial function that describes the temperature of the semiconductor and we define the coefficient

$$k_T \stackrel{\text{def}}{=} \frac{T}{300\,\text{K}},$$

the Arora model describes μ_n as

$$\mu_n \stackrel{\text{def}}{=} k_T^{\beta_1}\mu_{n,\min} + k_T^{\beta_2}\mu_{n,0}\left(\left(\frac{N_D+N_A}{N_D k_T^{\beta_3}}\right)^{\alpha_T}+1\right)^{-1},$$

where the exponents β_i, $\mu_{n,\min}$, and $\mu_{n,0}$ are constants that depend on the material. The exponent α_T, instead, is defined as

$$\alpha_T \stackrel{\text{def}}{=} \alpha_0 k_T^{\beta_4}$$

In a similar way, μ_p is written as

$$\mu_p \stackrel{\text{def}}{=} k_T^{\beta_1}\mu_{p,\min} + k_T^{\beta_2}\mu_{p,0}\left(\left(\frac{N_D+N_A}{N_A k_T^{\beta_3}}\right)^{\alpha_T}+1\right)^{-1}.$$

As far as the diffusion coefficient is concerned, we follow Einstein's relation that implies a linear dependency between D_n and μ_n:

$$D_n = U_T \mu_n,$$

where the coefficient U_T is the so called *thermal voltage*: if we denote by k_B the Boltzmann's constant, the thermal voltage can be written as

$$U_T \stackrel{\text{def}}{=} \frac{k_B T}{q}.$$

We have therefore that

$$\mathbf{J}_n = q\left(\mu_n n \mathbf{E} + D_n \nabla n\right)$$

$$\mathbf{J}_p = q\left(\mu_p p \mathbf{E} - D_p \nabla p\right)$$

and, also for p, we assume that $D_p = U_T \mu_p$. It is worth noting that μ_p is usually much smaller than μ_n; this is related to the fact that electrons can move much more freely than holes inside the lattice of a semiconductor crystal.

In the following, it will often be useful to refer explicitly to the opposite of the gradient of n and p; therefore, we define

$$\mathbf{W}_n \stackrel{\text{def}}{=} -\nabla n$$

$$\mathbf{W}_p \stackrel{\text{def}}{=} -\nabla p.$$

If we put all of these considerations together, we get a system of differential equations that is known as the (stationary) van Roosbroeck model; if we consider the gradient as a primal variable,we obtain the following formulation:

$$\begin{cases} \nabla V + \mathbf{E} = 0 & \text{(1.2a)} \\ \nabla \cdot (\varepsilon \mathbf{E}) = q(c - n + p) & \text{(1.2b)} \\ \nabla n + \mathbf{W}_n = 0 & \text{(1.2c)} \\ \nabla \cdot (\mu_n \mathbf{D} n - D_n \mathbf{W}_n) = qH(n,p) & \text{(1.2d)} \\ \nabla p + \mathbf{W}_p = 0 & \text{(1.2e)} \\ \nabla \cdot (\mu_p \mathbf{D} p + D_p \mathbf{W}_p) = -qH(n,p) & \text{(1.2f)} \end{cases}$$

while if we use as a primal variable the current, we get

$$\begin{cases} \nabla V + \mathbf{E} = 0 & \text{(1.3a)} \\ \nabla \cdot (\varepsilon \mathbf{E}) = q(c - n + p) & \text{(1.3b)} \\ \mathbf{J}_n - q\mu_n n \mathbf{E} - qD_n \nabla n = 0 & \text{(1.3c)} \\ \nabla \cdot \mathbf{J}_n = qH(n,p) & \text{(1.3d)} \\ \mathbf{J}_p - q\mu_p p \mathbf{E} + qD_p \nabla p = 0 & \text{(1.3e)} \\ \nabla \cdot \mathbf{J}_p = -qH(n,p) & \text{(1.3f)} \end{cases}$$

In this work, we assume that the previous equations are defined on a Lipschitz domain Ω. We use Γ to refer to the boundary of Ω. Of course, eqs. (1.2a) and (1.3a) need to be coupled with suitable boundary conditions. We assume that Γ is a union of 3 disjoint subsets

$$\Gamma = \Gamma_{\text{ins}} \cap \Gamma_{\text{A}} \cap \Gamma_{\Omega}.$$

Γ_{ins} represents the portion of the boundary that is coated by an insulating material. We model that insulator as ideal, i.e. we neglect interface charges and surface recombination and assume that the electric field in the thick insulating layer vanishes. Then we have

$$\mathbf{E} \cdot \nu = 0 \qquad \mathbf{J}_n \cdot \nu = 0 \qquad \mathbf{J}_p \cdot \nu = 0 \tag{1.4}$$

on every point of Γ_{ins}, where ν is the normal pointing outside of Ω.

Γ_{A} is nonempty only for devices embedded in integrated circuits and consists of boundary segments artificially introduced to separate the device from adjacent devices. This kind of boundaries have to be introduced so that they do not significantly perturb the model; moreover, artificial boundaries are sometimes introduced to simplify the numerical simulation by "cutting off" regions of the device with insignificant action. In order to make the device self-contained, we impose conditions that are analogous to the one that we have imposed for Γ_{ins}, i.e., vanishing outward electric field and vanishing outward current density components, as already described by eq. (1.4). Finally, Γ_{Ω} represents the metal semiconductor contacts, fabricated by bringing a metal into intimate contact with the semiconductor. In this thesis we assume that the contacts are Ohmic contacts, i.e., they have a negligible contact resistance. To describe mathematically the boundary conditions imposed by this kind of contacts, we need to study a little bit in more detail what happens to a semiconductor when it reaches the thermal equilibrium.

Let us introduce two important quantities: the quasi Fermi potentials for electrons and holes respectively:

$$\varphi_n \stackrel{\text{def}}{=} V - U_T \log\left(\frac{n}{N_c}\right) - \frac{E_c}{q}, \qquad \varphi_p \stackrel{\text{def}}{=} V + U_T \log\left(\frac{p}{N_v}\right) - \frac{E_v}{q}, \tag{1.5}$$

where N_c and N_v (the conduction and valence band densities of states, respectively) and E_c and E_v (the conduction and valence band-edge energies) are assumed to be constant and depending only on the material. It is worth noting that $\nabla\varphi_n$ (and, as well $\nabla\varphi_p$) do not depend on the values of these four constants, which, therefore, just modify the value of φ_n (or φ_p) by a constant. Moreover, it is easy to verify that

$$\mathbf{J}_n = -q\mu_n n \nabla\varphi_n, \qquad \mathbf{J}_p = -q\mu_p p \nabla\varphi_p. \tag{1.6}$$

The previous relations show the importance of the quasi fermi potentials: from an intuitive point of view, they are the driving forces of the currents.

A semiconductor is in *thermal equilibrium* when it is not attached to a circuit nor it is under the influence of an external electric field. In this situation, there is a perfect balance between the recombination and generation rates, i.e., H is 0 on every point of

the domain. Moreover, the quasi Fermi potentials are also 0 in every point and, because of (1.6), both currents are vanishing. From (1.5) we have that

$$n = N_c \exp\left(U_T(V - \varphi_n) - \frac{U_T E_c}{q}\right), \qquad p = N_v \exp\left(U_T(\varphi_p - V) + \frac{U_T E_v}{q}\right),$$

and using these relations inside the first two equations of system (1.2a) we obtain:

$$-\nabla \cdot (\varepsilon \nabla V) = q\left(c - N_c \exp\left(U_T V - q^{-1} U_T E_c\right) + N_v \exp\left(-U_T V + q^{-1} U_T E_v\right)\right) \quad (1.7)$$

which describes the *built-in potential*, i.e., the latent electric potential inside a semiconductor which is essentially produced by the doping. We will refer to this potential as V_e. The boundary conditions for eq. (1.7) are the same that we have described in eq. (1.4) for Γ_{ins} and Γ_{A}, while we assume that, on metal contacts, there is no charge, i.e., $(c - n + p) = 0$; this can be rewritten in terms of φ_n, φ_p, and V by requiring that, on every point of Γ_Ω, the right hand side term of eq. (1.7) is 0.

To compute the values of n and p at the thermal equilibrium, we simply solve (1.3c) and (1.3e) respectively, taking into account that at equilibrium $\mathbf{J}_n$ and $\mathbf{J}_p$ vanish. These equations can be solved analytically and we have that

$$n_e = A\mathrm{e}^{\frac{V_e}{U_T}}, \qquad p_e = B\mathrm{e}^{-\frac{V_e}{U_T}},$$

where n_e and p_e are spatial functions that describe the densities of electrons and holes (respectively), and A and B are constants that can be computed by imposing that

$$\varphi_n(n_e) = 0, \qquad \varphi_p(p_e) = 0.$$

What is interesting for us is that the product $n_e p_e$ is constant because of the cancellation of the dependency on V_e. We define the *intrinsic carrier density* as

$$n_i \stackrel{\text{def}}{=} \sqrt{n_e p_e}. \quad (1.8)$$

The relation exposed in eq. (1.8), together with the fact that the product $n_e p_e$ is a constant, is sometimes called *action mass law*.

As far as the charge carrier densities are concerned, we require that on every point of Γ_Ω, both the space charge vanishes (i.e. $c - n + p = 0$) and the action mass law holds ($np = n_i^2$). Combining these two conditions we obtain

$$n|_{\Gamma_\Omega} = \frac{1}{2}\left(\sqrt{c^2 + 4n_i^2} + c\right), \qquad p|_{\Gamma_\Omega} = \frac{1}{2}\left(\sqrt{c^2 + 4n_i^2} - c\right).$$

Instead, as far as the V is concerned, generally we prescribe a relationship between the potential and the out-flow current at the contact. One of the most common conditions is represented by the pure voltage drive, where the potential at the contact is prescribed. In this case, we have Dirichlet boundary conditions on V; if ψ is the imposed voltage, the condition reads

$$V|_{\Gamma_\Omega} = \psi|_{\Gamma_\Omega} + V_e|_{\Gamma_\Omega}$$

On the other side, it is also possible to have Neumann boundary conditions if the contact is a pure current drive. Finally, in some applications, mixed current-voltage driven contacts occur. An example is described in Chapter 2.

1.2 Discretization

In this section, we show a common approach (for example, used in [20]) to solve numerically the problem we exposed in Section 1.1. Let us assume that our domain Ω is a polytope and that $\mathcal{T}^{(\mathcal{V})}$ is a Voronoi tessellation of Ω. We will refer to the seeds points of $\mathcal{T}^{(\mathcal{V})}$ as the centers of the volumes T_i of $\mathcal{T}^{(\mathcal{V})}$.

We proceed in the following way:

- The "continuous" problem is replaced by a "discrete" system of nonlinear equations
- The system of nonlinear equations is then solved by an iterative scheme, for example with Newton's method

For the discretization, a common choice is given by the Finite Volume method (FVM). Indeed, by integrating eq. (1.3b) on each volume $T \in \mathcal{T}^{(\mathcal{V})}$, we obtain that

$$\int_T \nabla \cdot \mathbf{D} \,\mathrm{d}x = \int_T q(c - n + p) \,\mathrm{d}x.$$

Applying the divergence theorem we obtain

$$\int_{\partial T} \mathbf{D} \cdot \nu_T \,\mathrm{d}\gamma = \int_T q(c - n + p) \,\mathrm{d}x,$$

where ν_T is the outward-pointing normal vector. Denoting $\mathcal{F}_T$ the set of faces of the volume T, we have

$$\sum_{F \in \mathcal{F}_T} \int_F \mathbf{D} \cdot \nu_T \,\mathrm{d}\gamma = \int_T q(c - n + p) \,\mathrm{d}x. \tag{1.9}$$

The very same technique can be applied on equations (1.3d) and (1.3f), obtaining

$$\sum_{F \in \mathcal{F}_T} \int_F \mathbf{J}_{\{n,p\}} \cdot \nu_T \,\mathrm{d}\gamma = \pm q \int_T H(n,p) \,\mathrm{d}x. \tag{1.10}$$

The solution of the FVM method is a piecewise constant approximation $(V^{(\mathcal{V})}, n^{(\mathcal{V})}, p^{(\mathcal{V})})$ that associate to each volume a fixed value. Let $V_T^{(\mathcal{V})}$, $n_T^{(\mathcal{V})}$ and $p_T^{(\mathcal{V})}$ be the values that the functions $V^{(\mathcal{V})}$, $n^{(\mathcal{V})}$, and $p^{(\mathcal{V})}$ (respectively) assume on T. The variables that describe the fluxes ($\mathbf{E}$, $\mathbf{J}_n$, and $\mathbf{J}_p$) are condensed away and rewritten as a function of $V^{(\mathcal{V})}$, $n^{(\mathcal{V})}$, and $p^{(\mathcal{V})}$ respectively.

Let c_T be the average of c on the volume T:

$$c_T \stackrel{\text{def}}{=} \frac{1}{|T|} \int_T c \,\mathrm{d}x$$

The finite volume discretization of (1.9) reads

$$\sum_{F \in \mathcal{F}_T} \int_F \widetilde{\mathbf{D}}(V^{(\mathcal{V})}) \cdot \nu_T \,\mathrm{d}\gamma = q\,|T|\,(c_T - n_T^{(\mathcal{V})} + p_T^{(\mathcal{V})}). \tag{1.11}$$

Concerning the approximation of $\widetilde{\mathbf{D}}$ on the face F, we can rely on a finite difference scheme; let us assume that F is an interior face and that T_1 and T_2 are its adjacent volumes. We define

$$\int_F \widetilde{\mathbf{D}}(V^{(\mathcal{V})}) \cdot \nu_{T_1} \stackrel{\text{def}}{=} |F|\, \varepsilon_F \frac{V_{T_1}^{(\mathcal{V})} - V_{T_2}^{(\mathcal{V})}}{l_F},$$

where

- l_F is the distance between the center of the cell T_1 and the center of T_2;
- $|F|$ is the measure of the face F (in dimension $n-1$);
- ε_F is equal to ε if the two volumes T_1 and T_2 are made of the same material. Otherwise, usually ε_F is chosen as an average of the two values that ε assumes on the two volumes weighted accordingly to the distance of the centers from F.

The approach that we have followed is not strictly related to the van Roosbroeck problem, and it works as a general approach that can be used in every problem that can be written as a conservative problem (see, for example, [42]).

As far as the eq. (1.10) is concerned, we follow an analogous procedure:

$$\sum_{F \in \mathcal{F}_T} \int_F \widetilde{\mathbf{J}_n}(n^{(\mathcal{V})}, V^{(\mathcal{V})}) \cdot \nu_T \, \mathrm{d}\gamma = |T|\, qH\left(n_T^{(\mathcal{V})}, p_T^{(\mathcal{V})}\right),$$

$$\sum_{F \in \mathcal{F}_T} \int_F \widetilde{\mathbf{J}_p}(p^{(\mathcal{V})}, V^{(\mathcal{V})}) \cdot \nu_T \, \mathrm{d}\gamma = -|T|\, qH\left(n_T^{(\mathcal{V})}, p_T^{(\mathcal{V})}\right),$$

where, again, we have to define in an appropriate way the fluxes $\widetilde{\mathbf{J}_n}$ and $\widetilde{\mathbf{J}_p}$. In principle, we could define again these quantities by using a classical finite difference scheme but this will lead to a method whose results are often unreliable from a practical point of view. Indeed, if our problem is transport dominated, this scheme may be subjected to numerical instability (see, e.g.[8]). For this reason, some techniques have been developed to reduce this problem.

A typical approach, for example, consists into increasing artificially the diffusion term with a coefficient that depends on the size of the grid and that goes to zero decreasing the size of the cells. This idea leads to a big family of methods; we are particularly interested in one of them: the *Scharfetter–Gummel method* (sometimes also called *exponential fitting*).

The Scharfetter–Gummel method approximates the flux of the solution u considering the function u between the two centers of the cells T_1 and T_2 as an exponential function. The approach can be justified taking into account that, for dimension 1, if μ_n and $\mathbf{E}$ are constant along the edge that connects the two centers of the volumes and we neglect the effects of the recombination term H, then the analytical solution of eq. (1.3d) is indeed an exponential function (and the same goes for eq. (1.3f)).

The flux defined by the Scharfetter–Gummel scheme is

$$\int_F \widetilde{\mathbf{J}_n}(n^{(\mathcal{V})}, V^{(\mathcal{V})}) \cdot \nu_{T_1} \, \mathrm{d}\gamma \stackrel{\mathrm{def}}{=} -\frac{|F| \, q\mu_{n,F} U_T}{l_F} \left[B\left(-\frac{V_{T_2}^{(\mathcal{V})} - V_{T_1}^{(\mathcal{V})}}{U_T} \right) n_{T_1}^{(\mathcal{V})} - B\left(\frac{V_{T_2}^{(\mathcal{V})} - V_{T_1}^{(\mathcal{V})}}{U_T} \right) n_{T_2}^{(\mathcal{V})} \right]$$

$$\int_F \widetilde{\mathbf{J}_p}(p^{(\mathcal{V})}, V^{(\mathcal{V})}) \cdot \nu_{T_1} \, \mathrm{d}\gamma \stackrel{\mathrm{def}}{=} \frac{|F| \, q\mu_{p,F} U_T}{l_F} \left[B\left(\frac{V_{T_2}^{(\mathcal{V})} - V_{T_1}^{(\mathcal{V})}}{U_T} \right) p_{T_1}^{(\mathcal{V})} - B\left(-\frac{V_{T_2}^{(\mathcal{V})} - V_{T_1}^{(\mathcal{V})}}{U_T} \right) p_{T_2}^{(\mathcal{V})} \right]$$

where we have introduced the *Bernulli function* $B(x)$, defined as

$$B(x) \stackrel{\mathrm{def}}{=} \frac{x}{\mathrm{e}^x - 1}. \tag{1.12}$$

For deeper and more accurate description of the Scharfetter–Gummel scheme, see [21].

The previous two equations, together with eq. (1.9), provide a nonlinear system of equations. The nonlinearity is due to the computation of the carrier currents, which involves the values of $V^{(\mathcal{V})}$. The solution of such a system can be computed using classical techniques like, for example, the Newton method.

Chapter 2

Data driven solution of inverse and optimization problems in semiconductor devices

2.1 Introduction

In this chapter we apply the van Roosbroeck model introduced in Chapter 1 to a specific industrial application that requires the solution of an ill-posed inverse problem.

In particular, we introduce a data-driven strategy for the solution of general ill-posed inverse problems arising from photovoltage technologies where measured photovoltage signals are used to reconstruct local doping fluctuations in a semiconductor crystal. Noninvasive estimation of doping inhomogeneities in semiconductors is relevant for many industrial applications, ranging from controlling the semiconductor crystal purity during and after growth to detecting defects in semiconductor devices, employed, for example in solar cells.

Doping variations lead to local electrical fields which separate electron-hole pairs generated in their vicinity: experimentalists may exploit this mechanism to identify inhomogeneities, variations, and defects in the doping profile by systematically generating electron-hole pairs via some form of electromagnetic radiation. If we connect our semiconductor sample to a circuit, some of the new generated charge carriers will flow through the external circuit, and induce a current which may be measured. In this chapter, we use the van Roosbroeck system to model the semiconductor device and Ohm's law for the external circuit.

Concerning the electromagnetic source that we use to generate the charge carriers, different techniques have been developed that use different kind of sources (see [49, 7, 26, 34, 28]). Here we focus mainly on the LPS method (see [34, 20]), where the source is an optical laser.

This setup naturally leads to an inverse problem where the doping parameter c, defined in Section 1.1, must be reconstructed starting from a measure that depends on the solution of the van Roosbroeck system (1.2a). We solve this problem using three different data-driven approaches: least squares (Section 2.4.5), multilayer perceptrons (Section 2.4.6) and residual neural networks (Section 2.4.7).

The methods were trained on synthetic datasets (pairs of discrete doping profiles and corresponding photovoltage signals at different illumination positions). We describe the

generation of synthetic doping profiles in Section 2.4.2. These profiles are used to generate synthetic signals by solving the forward problem as described in Section 2.2. The final results obtained with the three data-driven strategies are presented in Sections 2.4.5, 2.4.6 and 2.4.7.

2.2 The inverse LPS problem

In this section we describe the combination of the drift-diffusion charge transport model with the circuit. When combining both models, we are able to formulate forward photovoltage models which may be used to predict a measured signal (either a current or a voltage) for a given doping profile and electromagnetic source.

The semiconductor crystal is modeled by a bounded domain $\Omega \subset \mathbb{R}^3$ where we solve the equations described in Section 1.1. The doping profile c is given by the difference of donor and acceptor concentrations,

$$c(\mathbf{x}) \stackrel{\text{def}}{=} N_D(\mathbf{x}) - N_A(\mathbf{x}),$$

where $\mathbf{x} = (x, y, z)^T \in \Omega$. We assume that c is a bounded function, and we call $\mathcal{C} \subseteq L^\infty(\Omega)$ the space of all admissible doping profiles.

The electromagnetic source (a laser or an electron beam) is modeled by adding a generation term $G(\mathbf{x}; \mathbf{x}_0)$ to the equations (1.3d) and (1.3f), obtaining

$$\nabla \cdot \mathbf{J}_n = q\left(H(n,p) + G(\mathbf{x};\mathbf{x}_0)\right), \qquad \text{and} \qquad \nabla \cdot \mathbf{J}_p = -q\left(H(n,p) + G(\mathbf{x};\mathbf{x}_0)\right). \quad (2.1)$$

Indeed, when the laser hits the crystal at the point $\mathbf{x}_0 := (x_0, y_0, z_0)^T$, some photons impinge the valence electrons of the crystal, creating new electron-hole pairs and resulting in a generation rate defined as follows

$$G(\mathbf{x}; \mathbf{x}_0) \stackrel{\text{def}}{=} \kappa S(\mathbf{x} - \mathbf{x}_0), \quad (2.2)$$

where $S(\mathbf{x})$ is the shape function of the laser (normalized by $\int_{\mathbb{R}^3} S(\mathbf{x}) d\mathbf{x} = 1$), while κ is a constant given by

$$\kappa \stackrel{\text{def}}{=} \frac{P_L \lambda_L}{h}(1 - r). \quad (2.3)$$

where P_L is the laser power, λ_L is the wave length of the laser, $h = 6.6 \times 10^{-34}\,\mathrm{J\,s}$ is the Planck constant, and r is the reflectivity rate of the crystal.

We assume that the area of influence of the electromagnetic source decays exponentially fast from the incident point $\mathbf{x}_0$. In particular, we take a laser profile function S defined as

$$S(\mathbf{x}) \stackrel{\text{def}}{=} \frac{1}{2\pi\sigma_L^2 d_A} \exp\left[-\frac{1}{2}\left(\frac{x}{\sigma_L}\right)^2\right] \exp\left[-\frac{1}{2}\left(\frac{y}{\sigma_L}\right)^2\right] \exp\left[-\frac{|z|}{d_A}\right]. \quad (2.4)$$

Here σ_L is the laser spot radius, while d_A is the penetration depth (or the reciprocal of the absorption coefficient), which heavily depends on the laser wave length. Figure 2.1

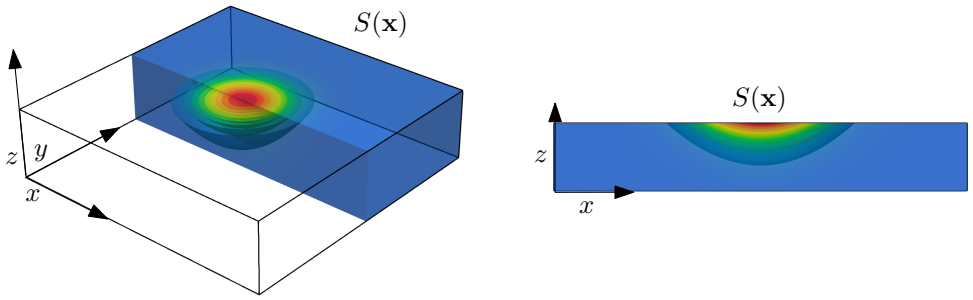


Figure 2.1: A zoom-in of the area around the point $\mathbf{x}_0$, where the laser hits the crystal (left), and a cross section of the device at $y = 0$ used in the 2D simulation (right).

shows a typical configuration, where the laser beam hits the crystal on the top surface, and shows the exponential decay of the laser shape function S.

As far as the recombination term H is concerned, in this chapter we assume that it is the sum of the direct recombination, the Auger recombination, and the Shockley-Read-Hall recombination, that is, respectively:

$$\begin{aligned} H_{\text{dir}} &\stackrel{\text{def}}{=} C_d(np - n_i^2), \\ H_{\text{Aug}} &\stackrel{\text{def}}{=} C_n n(np - n_i^2) + C_p p(np - n_i^2), \\ H_{\text{SRH}} &\stackrel{\text{def}}{=} \frac{np - n_i^2}{\tau_p(n + n_T) + \tau_n(p + p_T)}. \end{aligned}$$

Different types of crystal samples (such as silicon, germanium, gallium arsenide) have different life times τ_n, τ_p, and reference densities n_T, p_T. In our model problem we consider typical parameters for silicon, which we took from [20] and report in Table 2.1.

A prototypical setting for photovoltage measurements is given by a cuboid sample attached to a voltmeter with resistance R, and the electromagnetic source aiming at its top surface. In this work, we consider a crystal sample occupying the domain

$$\Omega \stackrel{\text{def}}{=} [-\ell/2, \ell/2] \times [-w/2, w/2] \times [-h, 0],$$

with length $\ell = 3\,\text{mm}$, width $w = 0.5\,\text{mm}$, and height $h = 5 \times 10^{-5}\,\text{mm}$, with ohmic contacts at $x = -\ell/2$ (referred as Γ_{D_1}) and $x = \ell/2$ (Γ_{D_2}, see Figure 2.2).

2.2.1 Boundary conditions

Our model is supplemented with Dirichlet and Neumann boundary conditions. The boundary $\partial\Omega$ is the union of two disjoint parts: Γ_{ins} and Γ_Ω. On Γ_{ins}, as we already

Physical Quantity	Symbol	Value	Units
Reference temperature	T	300	K
Density of states in the conduction band	N_c	1.04×10^{19}	$1/\text{cm}^3$
Density of states in the valence band	N_v	2.8×10^{19}	$1/\text{cm}^3$
Laser power	P_L	0 to 20	mW
Laser wave length	λ_L	685	nm
Laser penetration depth	d_A	4.8	µm
Laser spot radius	σ_L	$\geq$1.25	µm
Direct recombination factor	C_d	1×10^{-20}	cm^3/s
Auger recombination factor electrons	C_n	2.8×10^{-31}	cm^6/s
Auger recombination factor holes	C_p	2.8×10^{-31}	cm^6/s
SRH lifetime electrons	τ_n	1×10^{-9} to 1×10^{-3}	s
SRH lifetime holes	τ_p	1×10^{-9} to 1×10^{-3}	s
Arora mobility parameter	$\mu^{\text{ref}}_{\text{n},0}$	1323	cm^2/Vs
Arora mobility parameter	$\mu^{\text{ref}}_{\text{p},0}$	429	cm^2/Vs
Arora mobility parameter	$\mu^{\text{ref}}_{\text{n,min}}$	89	cm^2/Vs
Arora mobility parameter	$\mu^{\text{ref}}_{\text{p,min}}$	55	cm^2/Vs
Arora mobility exponent	α_0	0.88	–
Arora mobility exponent	β_1	-0.57	–
Arora mobility exponent	β_2	-2.33	–
Arora mobility exponent	β_3	2.4	–
Arora mobility exponent	β_4	-0.146	–

Table 2.1: The values of the physical constants that we used for our model. These values are coherent with a silicon semiconductor

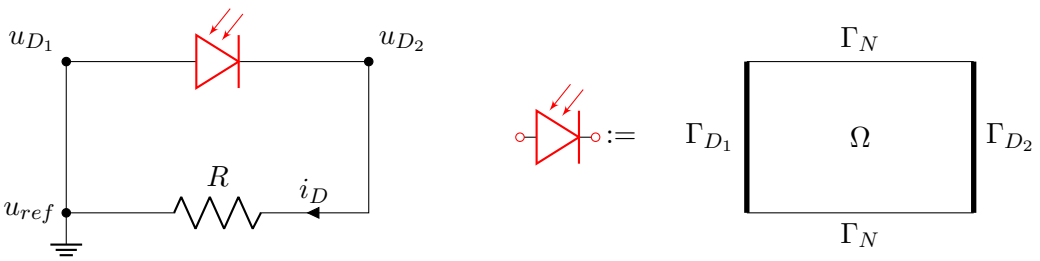


Figure 2.2: On the left we show a sample of photo-sensitive silicon crystal (in red) coupled with the voltage meter having resistance R. On the right, we show the domain Ω and the ohmic contacts Γ_{D_1} and Γ_{D_2}.

stated in Section 1.1, we assign homogeneous Neumann boundary conditions. On Γ_Ω, instead, we assign Dirichlet-type boundary conditions. More precisely, we assume that there are two ohmic contacts, i. e. $\Gamma_\Omega = \Gamma_{D_1} \cup \Gamma_{D_2}$. The ohmic boundary conditions can be summarized by

$$V - V_e = \varphi_n = \varphi_p = u_{D_i} - u_{\text{ref}} \quad \text{on} \quad \Gamma_{D_i}, \quad i = 1, 2, \tag{2.5}$$

where V_e is the built-in potential defined in eq. (1.7).

The terms u_{D_i} denote the contact voltages at the corresponding ohmic contacts; their difference is a-priori unknown. We define a reference value u_{ref} of the potential and set it to $u_{\text{ref}} = 0 = u_{D_1}$.

The electric current i_D flowing through the 2-nd ohmic contact Γ_{D_2} is defined by the surface integral

$$\begin{aligned} i_D\colon \quad \Omega \times \mathbb{R} \times \mathcal{C} \quad &\to \quad \mathbb{R} \\ (\mathbf{x}_0, u_{D_2}, c) \quad &\mapsto \quad i_D(\mathbf{x}_0, u_{D_2}, c) \stackrel{\text{def}}{=} \int_{\Gamma_{D_2}} \nu \cdot (\mathbf{J}_n(\mathbf{x}) + \mathbf{J}_p(\mathbf{x}))\, \mathrm{d}\sigma(\mathbf{x}). \end{aligned} \tag{2.6}$$

The current i_D is a function of the contact voltage u_{D_2}, of the laser spot position $\mathbf{x}_0$, and of the doping profile c.

In order to close the system, we model the voltage/ampere meter as a simple circuit having a resistance R. This structure is visualized in Figure 2.2. The network has two nodes, in which the potentials are respectively $u_{D_1} = 0 = u_{\text{ref}}$ and u_{D_2}. When the external circuit is connected to the crystal, the boundary condition u_{D_2} can no longer be chosen arbitrarily and must balance the voltage difference at the voltmeter due to the current i_D, which is equal to $R\, i_D$. A generalized theory of this coupling can be found in [3] and in the references therein.

Let $\mathcal{U}$ be the function space of all possible photovoltage signals that can be measured given a doping profile c. We define the *laser-voltage* (L-V) map as the map that associates to each laser spot location $\mathbf{x}_0$ the corresponding photo-voltage signal u_P that would balances the voltage difference across R when solving the van Roosbroeck system with boundary data u_P, i.e.,

$$\begin{aligned} u_P\colon \quad \Omega \times \mathcal{C} \quad &\to \quad \mathbb{R} \\ (\mathbf{x}_0, c) \quad &\mapsto \quad u \in \mathcal{U} \text{ such that } u - R\, i_D(\mathbf{x}_0, u, c) = 0. \end{aligned} \tag{2.7}$$

The laser-voltage map (2.7) is an implicit equation for u since i_D depends implicitly on u, via the van Roosbroeck system (1.3a) modified as seen in eq. (2.1).

An existence result for solutions of the laser-voltage map is provided in [13], with the assumption that G is small enough. For a fixed doping profile c, the laser-voltage map $u_P(\mathbf{x}_0, c)$ provides a different photovoltage signal u_P for each laser spot location $\mathbf{x}_0$.

2.2.2 Computational complexity

It is worth pointing out that performing a numerical simulation of the entire LPS process requires the solution of the model described in Section 1.2 thousands of times. Indeed, for a specific laser spot position $\mathbf{x}_0 \in \Omega$, we need to solve a non linear differential problem.

Our goal is to produce a synthetic dataset with thousands of accurate 3D simulations; this requires a computational power that is far beyond what we have at our disposal. A possible solution to this problem consists in improving considerably the efficiency of the numerical solution of the forward problem, for example employing high-orders algorithms to solve the van Roosbroeck system. In order for this strategy to be applicable, such high order methods should provide the same stability properties that are enjoyed by the Scharfetter–Gummel scheme described in Section 1.2 and, at the same time, should achieve a higher accuracy using a coarser grid, drastically reducing the required computational resources. Finding such an algorithm is still an open problem, but our contribution in this direction is presented in Chapters 3 and 4.

Here, to avoid the excessive computational complexity, we only simulate a two-dimensional section of the domain where we fix $y = 0$. This setting is reasonable for applications in crystal growth, where most of the doping variations happen along the growth direction, and one can assume little or no variations along the y direction. With this setting we are able to simulate the exponential decay of the laser when it reaches the surface of the samples, and the x–axis, that is the most relevant dimension of our sample and the dimension on which, experimentally, the doping profile exhibits the most relevant variations.

2.2.3 The global forward photovoltage problem

Although the presented forward PDE model is well-posed for low laser power intensity and for all laser spot positions $\mathbf{x}_0 \in \Omega$, not all positions in Ω can be illuminated by a laser. The major limiting factor is the fact that the laser power intensity decays exponentially away from the surface of the crystal sample, leaving in fact as the only feasible positions those that are near or on the surface of the crystal sample.

We denote with $\Sigma \subset \overline{\Omega}$ the set of all admissible laser spot positions $\mathbf{x}_0$. Our *global forward photovoltage problem* then associates to each doping profile c the function u that maps each laser spot position $\mathbf{x}_0 \in \Sigma$ to the corresponding photovoltage signal $u_P(\mathbf{x}_0, c)$, i.e.,

$$\begin{array}{rccll} U: & \mathcal{C} & \to & \mathcal{U} & \\ & c & \mapsto & u : \Sigma \to \mathbb{R}, \qquad u(\mathbf{x}) = u_P(\mathbf{x}, c), & \mathbf{x} \in \Sigma. \end{array} \tag{2.8}$$

2.3 Inverse photovoltage problems

From an industrial perspective, more interesting than the forward photovoltage problem is the inverse one. How do doping inhomogeneities influence the measured voltage difference u_P, defined in the previous section? Assume we have measured the photovoltage signal for several different laser spot positions $\mathbf{x}_0$, how does the doping profile look like that leads to this signal? Since the doping profile enters as a volumetric source term defined in the whole domain in the van Roosbroeck system, but we can only probe part of the domain with the laser signal, answering such a question implies solving an ill-posed inverse problem: different global doping profiles may lead to the *same* signal. Hence, we

will first formulate an idealized global inverse photovoltage problem and then a practically relevant local inverse photovoltage problem.

2.3.1 Global inverse photovoltage problem

Ideally, we would like to find the *global doping reconstruction operator*

$$\widetilde{F} := U^{-1} \colon \mathcal{U} \to \mathcal{P}(\mathcal{C}),$$

that, for a given photovoltage signal measurement u, returns the preimage $U^{-1}(u)$. Here we indicate with $\mathcal{P}(\mathcal{C})$ the power set of $\mathcal{C}$, i.e., the set of all possible subsets of $\mathcal{C}$. The *global inverse photovoltage problem* reads

$$\widetilde{F}(u) = \mathcal{C}_u, \qquad \mathcal{C}_u := \{c \in \mathcal{C} \text{ such that } u_P(\cdot, c) = u\}, \tag{2.9}$$

that is, $\widetilde{F}$ is a function from $\mathcal{U}$ to $\mathcal{P}(\mathcal{C})$ (i.e., a multi-valued function of the signal u) such that $c \subseteq \widetilde{F}(u_P(\cdot, c))$ for all $c \in \mathcal{C}$.

2.3.2 Local inverse photovoltage problem

In practice, however, the construction of the operator $\widetilde{F}$ (even the construction of an approximation of $\widetilde{F}$) is nontrivial, and we would like to simplify our problem by building a local inverse problem that is better posed. A key point when defining a *local inverse problem*, is the fact that we cannot probe the entire domain Ω with the laser, but only the subset of all possible laser spot positions Σ near or on the surface of Ω. In the following subsections, we will make some assumptions on the structure of technologically feasible doping profiles. Technological feasibility depends on the growth process, on the technology used to inject doping in the crystal, and on the specific photovoltage technology. Here, we assume variation only in the x-direction. That is, $c(\mathbf{x}) = c(x)$. This class of doping profiles is relevant, for example, for crystal growth, where striations in the doping profile indicate fluctuations of the temperature field during the growth process, which are dominant in the growth direction.

We define the set of technologically feasible (TF) doping profiles and the set of corresponding signals as

$$\mathcal{C}_{\mathrm{TF}} = \{c \in \mathcal{C} \text{ such that } c(\mathbf{x}) = c(x)\}, \qquad \mathcal{U}_{\mathrm{TF}} := u_P(\cdot, \mathcal{C}_{\mathrm{TF}}). \tag{2.10}$$

With these assumptions on the doping profile, it is reasonable to presume that the photovoltage signal will be influenced more by variations of the doping profile in the vicinity of the laser spot position, which is where most of the charge carriers are generated, and therefore we define the *local inverse photovoltage problem* by restricting the global inverse problem (2.9) both in terms of technologically feasible doping profiles, as well as in terms of probing domain:

$$F(u) := (\widetilde{F}(u) \cap \mathcal{C}_{\mathrm{TF}})|_\Sigma = (\mathcal{C}_u \cap \mathcal{C}_{\mathrm{TF}})|_\Sigma =: \mathcal{C}_{\mathrm{TF},u,\Sigma}, \tag{2.11}$$

where the restriction operator | is applied to each element in the set. The goal of the local inverse problem is to reconstruct doping profiles only in the probing area Σ and not on the whole domain Ω. However, the underlying assumption is that the local doping reconstruction will give us information in a neighborhood of Σ, or even on all of Ω, when one makes additional assumptions (for example on doping periodicity). In general, the set $\mathcal{C}_{\mathrm{TF},u,\Sigma} = F(u)$ is much smaller than the set $\mathcal{C}_u := \widetilde{F}(u)$, since it restricts the family of admissible doping profiles to $\mathcal{C}_{\mathrm{TF}}$, and discards all of the information outside of Σ. Two doping profiles in $\mathcal{C}_u$ correspond to the same element in $\mathcal{C}_{\mathrm{TF},u,\Sigma}$ as soon as their restrictions on Σ coincide.

In principle, it should be possible to formulate a well-posed local inverse problem by further reducing the admissible doping profiles $\mathcal{C}_{\mathrm{TF}}$ until the set $\mathcal{C}_{\mathrm{TF},u,\Sigma}$ contains just one single element for any admissible input signal $u \in U(\mathcal{C}_{\mathrm{TF}})$. We do not know, however, if this procedure is feasible, and what the necessary and sufficient conditions on $\mathcal{C}_{\mathrm{TF}}$ and on Σ are to ensure that the local inverse problem $F(u)$ is well-posed. We leave these questions for future investigations, and concentrate on numerical approximations of the local inverse problem based on well-posed finite dimensional approximations of F. In Section 2.4 we propose three different strategies (of increasing complexity) to build an approximate inverse operator F_h starting from a collection of doping profiles restricted to a discrete set of probing points Σ_h and their corresponding discrete photovoltage signals.

It is worth noting that the numerical approximations that we construct always produce a unique answer for each finite sampling of a signal u in $\mathcal{U}_{\mathrm{TF}}$.

We do not attempt to construct the full set $\mathcal{C}_{\mathrm{TF},u,\Sigma}$ of all possible doping profiles that would generate u, but only provide the sampling of a single doping profile C, which is, in some sense, a representative of the set $\mathcal{C}_{\mathrm{TF},u,\Sigma}$.

2.4 Data-driven approximation of the inverse photovoltage problem

Inverse problems for charge transport equations have often been tackled with standard techniques (see, for example, [12, 29, 40]). However, in other fields such as image recognition or weather prediction, data-driven approaches have gained significant momentum to solve a variety of inverse problems (see, for example, [6, 30, 1]).

In order to formulate a data-driven approach for the numerical approximation of the operator F, we leverage the well-posedness of the forward problem U and its discrete approximation described in [20] to formulate a discrete inverse problem F_h as a well-posed minimization problem in a finite dimensional space.

2.4.1 Discrete local inverse problem

Let $\Sigma \subseteq \overline{\Omega}$ be the subdomain of admissible laser spot positions. We assume we sampled both the photovoltage signal u_P and the doping profile c at $n \in \mathbb{N}$ discrete laser spot positions contained in the discrete mesh $\Sigma_h \subseteq \Sigma$. With the boldface symbols $\mathbf{u}$ and $\mathbf{C}$ we indicate the discrete samplings of the signal $u_P(\cdot, c)$ and the doping profile c evaluated on

Σ_h, respectively. Notice that the generation of a single pair of signal and doping samples $(\mathbf{u}, \mathbf{C}) \in \mathbb{R}^{n\times 2}$ from an admissible doping profile $c \in \mathcal{C}_{\mathrm{TF}}$ requires in fact the solution of n discrete van Roosbroeck system (1.3a), one for each laser spot position $\mathbf{x}_0 \in \Sigma_h$.

The *local discrete inverse problem* F_h can be interpreted as a (generally nonlinear) function that maps a vector of signal measurements $\mathbf{u} \in \mathbb{R}^n$ to a vector of doping profile values $\mathbf{C} \in \mathbb{R}^n$:

$$\begin{aligned} F_h \colon \mathbb{R}^n &\to \mathbb{R}^n \\ \mathbf{u} &\mapsto \mathbf{C}. \end{aligned} \tag{2.12}$$

Ideally, we would like to build F_h in such a way that for all u in $\mathcal{U}_{\mathrm{TF}}$, there exists a $c \in \mathcal{C}_{\mathrm{TF},u,\Sigma} = F(u)$ such that $F_h(\mathbf{u} := u|_{\Sigma_h}) = \mathbf{C} := C|_{\Sigma_h}$. However, this procedure suffers from the non-uniqueness of $F(u)$, which we mitigate by constructing F_h through a well-posed minimization problem.

In practice, we build F_h by minimizing the mean square error loss

$$\mathrm{MSEL}\left(\{\mathbf{u}_j, \mathbf{C}_j\}_{j=1}^{N_{\mathrm{train}}}\right) \stackrel{\mathrm{def}}{=} \frac{1}{N_{\mathrm{train}}} \sum_{j=1}^{N_{\mathrm{train}}} \|F_h(\mathbf{u}_j) - \mathbf{C}_j\|_{\ell^2}^2, \tag{2.13}$$

on a given *training dataset* $\{\mathbf{u}_j, \mathbf{C}_j\}_{j=1}^{N_{\mathrm{train}}}$ consisting of a collection of N_{train} pairs of signal samples $\mathbf{u}_j$ and doping samples $\mathbf{C}_j$. The generation process of the data is detailed in Section 2.4.2.

The quality of the approximation of F_h is evaluated by measuring the prediction capabilities of F_h on a *test dataset* that is independent of the training dataset. For the evaluation of the prediction quality, we use a perhaps more physically relevant average ℓ^∞ norm, and define the *test error* as

$$\mathrm{TE}\left(\{\mathbf{u}_j, \mathbf{C}_j\}_{j=1}^{N_{\mathrm{test}}}\right) \stackrel{\mathrm{def}}{=} \frac{1}{N_{\mathrm{test}}} \sum_{j=1}^{N_{\mathrm{test}}} \|F_h(\mathbf{u}_j) - \mathbf{C}_j\|_{\ell^\infty}. \tag{2.14}$$

We develop three approaches to build F_h:

- *Least squares* (LS): we approximate $F_h(\mathbf{u})$ by the matrix vector product of an $n \times n$ matrix $\mathbf{A}_h$ with $\mathbf{u}$. We find the matrix by minimizing the MSEL error defined in (2.13) over all $n \times n$ matrices;
- *Multilayer perceptron* (MLP): we approximate $F_h(\mathbf{u})$ by a multilayer perceptron, and use Stochastic Gradient Descent (SGD) to minimize the MSEL;
- *Residual neural network* (ResNet): we approximate $F_h(\mathbf{u})$ by a residual neural network, and use SGD to minimize the MSEL.

2.4.2 Data generation

For our data-driven approaches, we need a suitable amount of data; unfortunately, generating enough data from real life requires large budgets. Instead, we will generate

synthetic data from the numerical model that we have described in Section 1.2. This data is physically meaningful in the sense that our finite volume approximation of the LPS problem correctly incorporates physically meaningful behavior, namely

- the signal intensity depends linearly on local doping variations for low laser powers;
- the signal saturates for higher laser intensities due to a screening effect;
- the signal depends logarithmically on moderate laser intensities.

All these consideration are coherent with the theoretical results exposed in [20].

As pointed out in Section 2.2, we consider a family of doping profiles that vary only along the x direction, and that are constant in both y and z directions. This class of doping profiles is relevant, for example, for crystal growth, where striations in the doping profile indicate fluctuation of the temperature during the growth process, and present these variations mainly along the growth direction.

To generate our synthetic datasets, we use an algorithm that produces fluctuating doping profiles. Since within the LPS framework the doping profile is roughly periodic (see [20]) we assume a superposition of sinusoidal functions from which we randomly sample physically reasonable amplitudes and wavelengths.

Therefore, we define

$$c\big(\mathbf{x} = (x, y, z)^T; \boldsymbol{\beta} = (c_0, \boldsymbol{\alpha}, \boldsymbol{\lambda})^T\big) = c_0 \left(1 + \sum_{i=1}^{f} \alpha_i \sin \frac{2\pi x}{\lambda_i}\right), \tag{2.15}$$

where $\boldsymbol{\beta}$ is a vector of parameters that includes a fixed average doping value

$$c_0 = 1.0 \times 10^{16}\,\mathrm{cm}^{-3},$$

which is a typical value for silicon crystals, while the amplitudes $\boldsymbol{\alpha} \overset{\text{def}}{=} \{\alpha_i\}$ are defined such that for every index i

$$\alpha_i \subset \{0\} \cup [0.05, 0.2],$$

and the wavelengths $\boldsymbol{\lambda} \overset{\text{def}}{=} \{\lambda_i\}$ are chosen such that

$$\lambda_i \subset [10\,\mu\mathrm{m}, 1000\,\mu\mathrm{m}].$$

In our simulation, we set the number of sine terms to $N_b = 5$, which seems to strike a good balance between complexity and real-life situations for striations in doping profiles.

To generate an element $\{\mathbf{u}, \mathbf{c}\}$ of our dataset, we randomly sample a parameter $\boldsymbol{\beta}$ with which we generate a continuous function $c(\mathbf{x}, \boldsymbol{\beta})$ and we compute the discrete counterpart of $u \overset{\text{def}}{=} U(c)$ defined in Equation (2.8) using the finite volume scheme described in Section 1.2. Finally, we restrict both c and u to the discrete laser spot mesh Σ_h.

In realistic physical scenarios, the doping profiles contain noises, and they cannot be represented exactly by Equation (2.15). To simulate such scenarios, we perturb some of the doping profiles and generate an additional *noisy* dataset, used to test the robustness of our networks in the presence of noise. In what follows, we say that a dataset is "noisy" or "clean" if c has or has not been perturbed. The algorithm used to generate noisy datasets is described in Section 2.4.3.

2.4.3 Generation of noise

In order to introduce noise in the doping profile (2.15), we perturb both the doping amplitudes as well as the wavelengths. To perturb the doping amplitudes, we start from some elements of $\mathcal{C}_{\mathrm{TF}}$. We introduce imperfections of the doping profile that may be interpreted physically with a slight perturbation during the growth process (for example, due to a fluctuation of the temperature), we choose a *random* function $f_n(x)$ and define a perturbed doping of the form $\tilde{c}(x) \stackrel{\text{def}}{=} c(x) + f_n(x)$. We require that $f_n(x) \ll c(x)$ for any x. To generate $f_n(x)$, we pick 129 equally-spaced points x_i across the silicon sample and randomly sample 129 values from a normal distribution (with 0 mean and standard deviation equal to 1), obtaining a family of values s_i. Denoting the maximal variation of c with

$$\Delta_c \stackrel{\text{def}}{=} \max_{x\in[-\ell/2,\ell/2]} c(x) \quad - \min_{x\in[-\ell/2,\ell/2]} c(x),$$

we define

$$f_n(x_i) \stackrel{\text{def}}{=} k\, s_i\, \Delta_c,$$

for $0 < k \ll 1$. For any other point $x \neq x_i$ across the sample, we compute $f_n(x)$ by cubic spline interpolation. Next, we perturb the wavelengths. In (2.15), we assumed that the doping has fluctuations with constant wavelength across the entire domain. We weaken this assumption by introducing a non periodic perturbation in the argument of the sinusoidal functions: we consider a differentiable function $t\colon [-\ell/2, \ell/2] \longrightarrow [-\ell/2, \ell/2]$, such that $t(-\ell/2) = -\ell/2$, $t(\ell/2) = \ell/2$, and $t'(x) > 0$ for every x; then we define the perturbed doping as $\bar{C}(x) \stackrel{\text{def}}{=} \tilde{C}(t(x))$.

In order to generate t, we impose that $p(x) \stackrel{\text{def}}{=} t(x) - x$ is a polynomial of degree 2 or 3. Due to the properties of the function t, we obtain that $p(-\ell/2) = 0$ and $p(\ell/2) = 0$. We randomly decide whether to use a polynomial of degree 2 or of degree 3, that is, we use

$$p(x) = k\,(x + \ell/2)\,(x - \ell/2) \qquad \text{or} \qquad p(x) = k\,(x + \ell/2)\,(x - \ell/2)\,(x - \alpha)$$

where k and α are random constants chosen so that $p'(x) > -1$ for every x.

Applying the transformation on a doping function, we can generate new samples for our dataset whose doping can not be described simply by choosing some suitable parameters in (2.15).

2.4.4 The datasets

In the following, we summarize the datasets used to construct and test the learning models introduced in Section 2.4.5–2.4.7. The datasets are available in the repository [41].

CleanDataSET

Consisting of:

• **CleanTrainingDataSET**: a dataset with $N_{\text{train}} = 22,500$ clean samples used to train our initial model;

• **CleanTestDataSET**: a dataset with $N_{\text{test}} = 7,500$ clean samples. With this dataset we test the performance of our models trained by **CleanTrainingDataSET**. Note that testing errors generated by this dataset (cf. (2.14)) implies the quality of a trained model. This will help use tune the corresponding hyperparameters;

• **CleanValidationDataSET**: a dataset with the same size of the **CleanTestDataSET** that will be used to validate the performance of our model on the clean case after performing the tuning.

NoisyDataSET

Consisting of:

• **NoisySmallTestDataSET**: again a dataset of $N_{\text{test}} = 7,500$ samples, but this time with noise. This is used to test the robustness to noise of the models trained by the **CleanTestDataSET**;

• **NoisyTrainingDataSET**: a dataset with $N_{\text{train}} = 150,000$ noisy samples that we use to train our models to be robust to noise;

• **NoisyTestDataSET**: a dataset with $N_{\text{test}} = 50,000$ noisy samples. This is the dataset that we use to test the performances of our models trained with the **NoisyTrainingDataSET**.

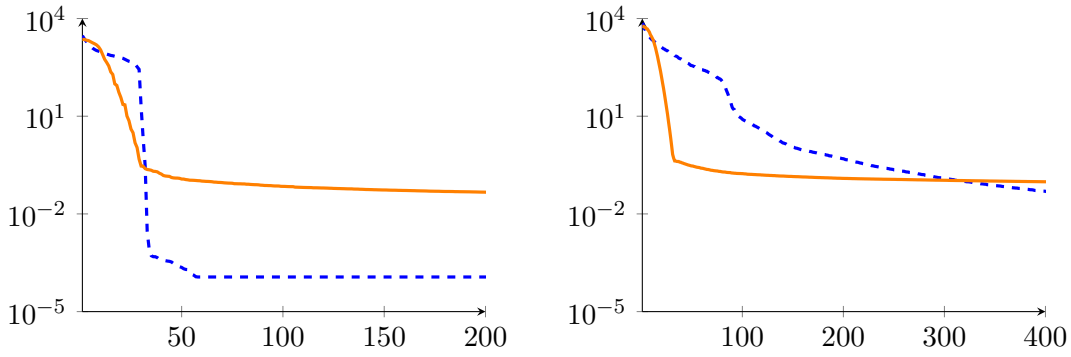


Figure 2.3: SVD analysis of **CleanTrainingDataSET** and **NoisyTrainingDataSET**. The figure shows the magnitude of the first 200 singular values for **CleanTrainingDataSET** (left) and the first 400 singular values for **NoisyTrainingDataSET**. The singular values for the doping profile matrices are shown in dashed blue lines and the singular values for the photovoltage signals are shown as solid orange lines.

We provide a qualitative analysis of **CleanTrainingDataSET** and **NoisyTrainingDataSET** in Figure 2.3, where we compute the Singular Value Decomposition (SVD) of both datasets interpreted as $n \times N_{\text{train}}$ matrices. The number of dominant singular values is a crude indication of the actual dimension of the spaces $\mathcal{C}_{\text{TF}}$, and $\mathcal{U}_{\text{TF}}$, and shows that, roughly, their dimension is close to 50 when using the clean data generation described above, see Figure 2.3 left panel. The two dimensions mismatch significantly when adding

noise, see Figure 2.3 right panel. While the dimension of the clean photovoltage signals follows roughly that of the clean doping profiles, the same cannot be said for the noisy cases. This difference is a hint that the inverse operator F is not well-posed, due to a mismatch in the dimension of the input and output spaces.

2.4.5 Least squares

In [20], the authors show that, under certain restrictive conditions, the operator F can be approximated by a linear one. Indeed, in case of an n or p doped semiconductor that varies only along the x direction, we expect that

$$u_P(\mathbf{x}, c) \sim -\mathbf{e}_x \cdot \nabla c(\mathbf{x}). \tag{2.16}$$

If the support of the laser source is larger than the wavelength of oscillation of the doping profile, measuring the voltage difference may actually result in a signal that does not capture a single oscillation, but a floating average. We could represent this floating average of doping fluctuations by a convolution of the doping gradient with some (unknown) profile f, i.e.

$$u_P(x, c) \sim (f * (\mathbf{e}_x \cdot \nabla c))(x) = \frac{\mathrm{d}}{\mathrm{d}x} (f * c)(x). \tag{2.17}$$

Hence, it may seem plausible to relate the photovoltage signal to the doping profile via *linear* operations such as convolution and differentiation. However, it is not clear whether this profile function f is actually independent from the doping fluctuations, the shape of the laser beam or the laser spot position.

The least square analysis is useful to understand how the inverse problem behaves, and what type of operation the inverse operator $F_h(\mathbf{u}) = \mathbf{A}_h\mathbf{u}$ performs. Let us express the training data for the photovoltage signals and doping profiles generated according to Section 2.4.2 with the two dense matrices $\mathbf{U}^{n\times N_{\text{train}}}$ and $\mathbf{C}^{n\times N_{\text{train}}}$. Then we wish to solve the least square problem

$$\mathbf{A}_h = \underset{\mathbf{B}\in\mathbb{R}^{n\times n}}{\arg\min} \frac{1}{2} \left\| \mathbf{B}\mathbf{U}^{n\times N_{\text{train}}} - \mathbf{C}^{n\times N_{\text{train}}} \right\|_{\ell^2}^2. \tag{2.18}$$

In Figure 2.4 we show $\mathbf{A}_h$ obtained with **CleanDataSET** (left) and with **NoisyDataSET** (right). We observe a highly non-local behavior of F_h. In the **CleanDataSET** case, this behavior is more pronounced than in the **NoisyDataSET** case. The observation of some of the rows and columns of the matrices (see Figures 2.6 and 2.7) does shed some light on what the linear operator $\mathbf{A}_h$ is actually doing.

The rows shown in Figure 2.6 could be interpreted as a way to compute a sort of Fourier transform of the input signal $\mathbf{u}$, that relates the frequency content of the input signal with the value of the doping profile at a specific sample point.

Similarly, the columns shown in Figure 2.7 could be seen as a collection of basis functions for the space of the doping profiles $\mathcal{C}_{\text{TF}}$, and show how a doping profile would look like, when only one single sample point produces a signal. Indeed, this is a very

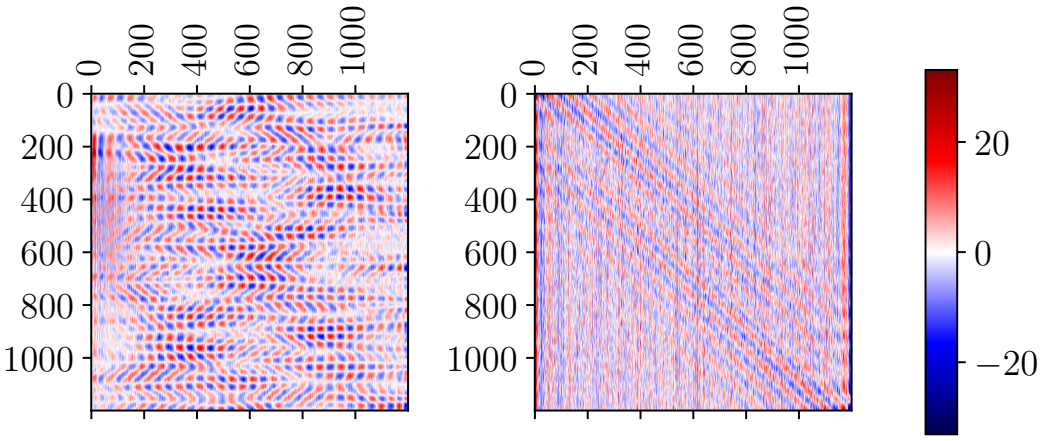


Figure 2.4: Magnitude of the entries of the least square matrices $\mathbf{A}_h$ obtained for **CleanDataSET** (left) and **NoisyDataSET** (right).

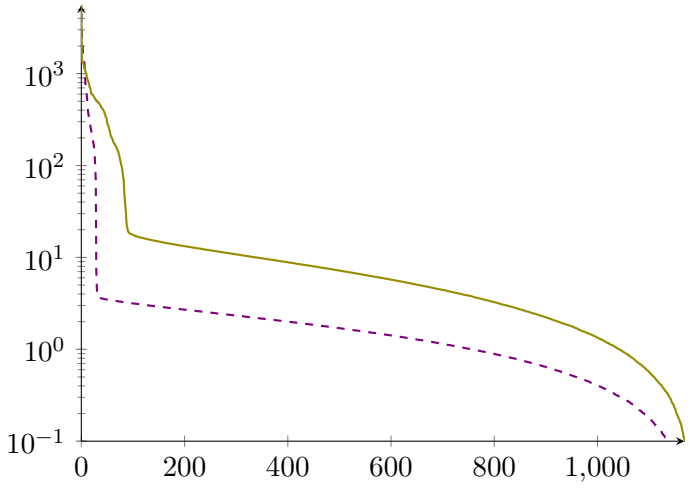


Figure 2.5: Magnitude of singular values of the least square matrices $\mathbf{A}_h$ obtained for **CleanDataSET** (violet dashed line) and **NoisyDataSET** (green continuous line).

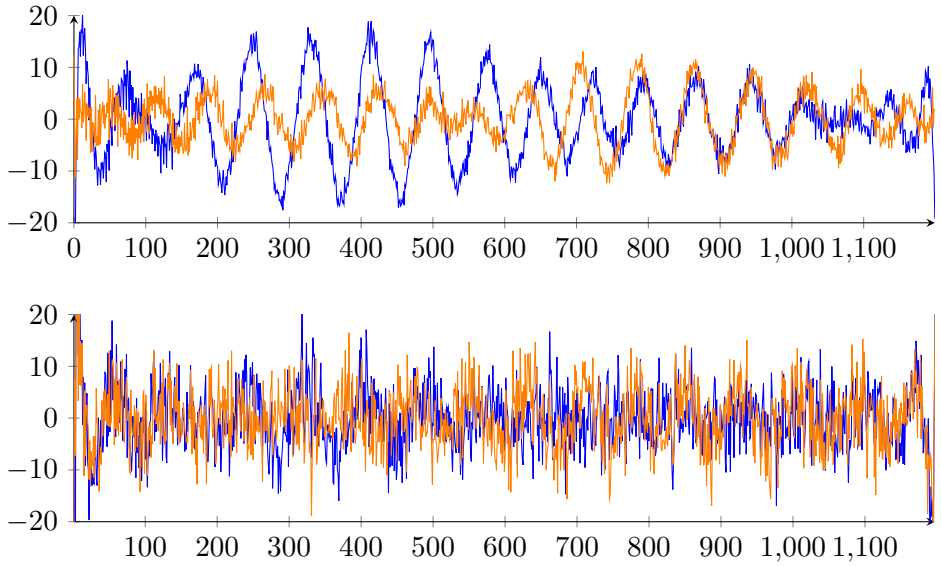


Figure 2.6: The rows 201 (blue) and 801 (orange) of the least square matrices obtained with the **CleanDataSET** (top) and **NoisyDataSET** (bottom).

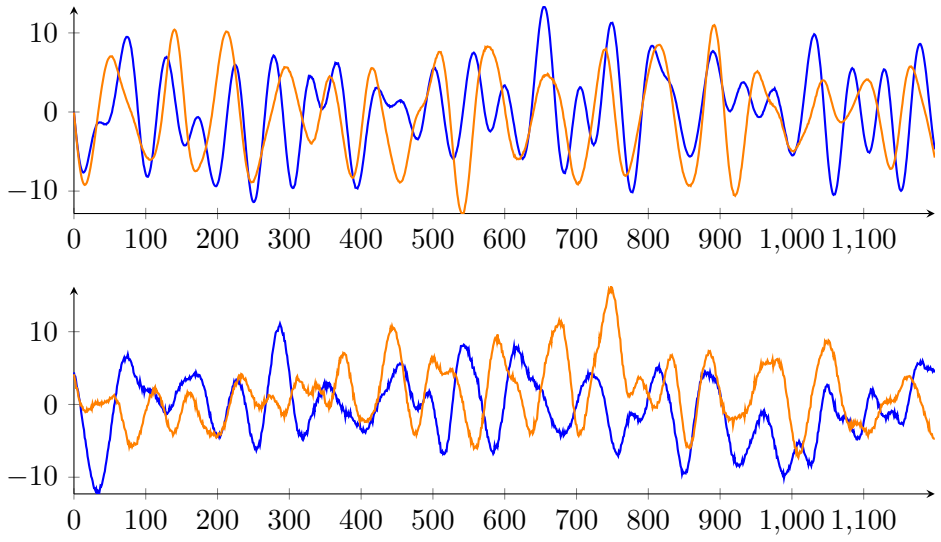


Figure 2.7: The columns 201 (blue) and 801 (orange) of the least square matrices obtained with the **CleanDataSET** (top) and with the **NoisyDataSET** (bottom).

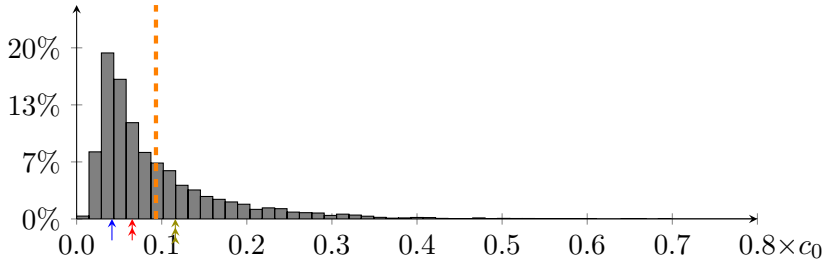


Figure 2.8: Statistical distribution of the errors on the predictions of our least square model, trained on the **CleanTrainingDataSET** and tested on the **CleanTestDataSET**. The dashed lines represent the average value of the error, while the arrows point to the 25, 50, and 75-percentile of the error of the histogram.

crude and biased interpretation of the operator F_h, which improves when adding noise, but it is still too tightly linked with the way the input datasets are constructed.

If we compare the singular values of the matrix $\mathbf{A}_h$ in Figure 2.5 with those of the two datasets in Figure 2.3, we observe that the dominant singular values of the least square matrix computed with the **CleanDataSET** are the first thirty (similar to what happens in the singular values of the **CleanDataSET** itself in Figure 2.3), while those that are most relevant for the matrix constructed with the **NoisyDataSET** are the first one hundred. While this information is only qualitative, it does show that a non-negligible part of the local inverse problem can be approximated relatively well by the linear operator $F_h(\mathbf{u}) = \mathbf{A}_h\mathbf{u}$, with an intrinsic dimension of around one hundred, when including noise, and much smaller when considering a clean dataset.

Train	Test	w/o mean	Average	25-th percentile	Median	75-th percentile
clean	clean		9.32%	4.15%	6.53%	1.16%
clean	noisy		21.5%	11.5%	17.9%	28.4%
clean	noisy	$*$	15.5%	6.75%	10.5%	21.0%
noisy	noisy		11.4%	5.72%	8.46%	13.8%
noisy	noisy	$*$	9.81%	4.20%	6.47%	11.9%

Table 2.2: Absolute ℓ^∞ errors w.r.t $c_0 = 1.0 \times 10^{16}\,\mathrm{cm}^{-3}$ of the least square model, corresponding to the colored arrows as well as dashed lines showing in Figures 2.8 to 2.10. The third column describes if we remove or not the mean from the prediction and from the expected result before computing the error.

The statistical distribution of the test error in the **CleanDataSET** (i.e., the error defined in (2.14) for the **CleanTestDataSET**) is reported in Table 2.2 and depicted in Figure 2.8. It shows an error which is, on average (orange dashed line), around 9%. Even though such a model presents a relatively high error, we observe that it is still capable of capturing the overall qualitative behavior of the doping profiles (see Figure 2.11).

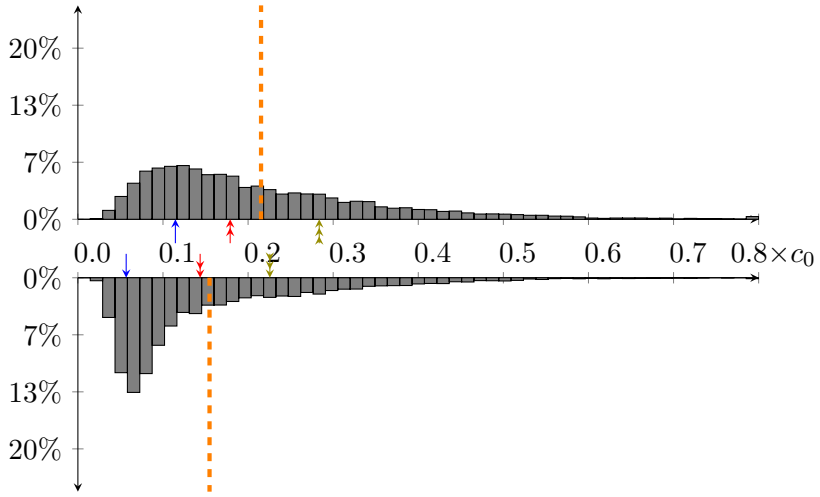


Figure 2.9: Statistical distribution of the errors on the predictions of our least square model, trained on the **CleanTrainingDataSET** and tested again the **NoisySmallTestDataSET**. The dashed lines represent the average value of the error, while the arrows point to the 25, 50, and 75-percentile of the error on the top histogram, and show how these change when removing the average in the bottom histogram.

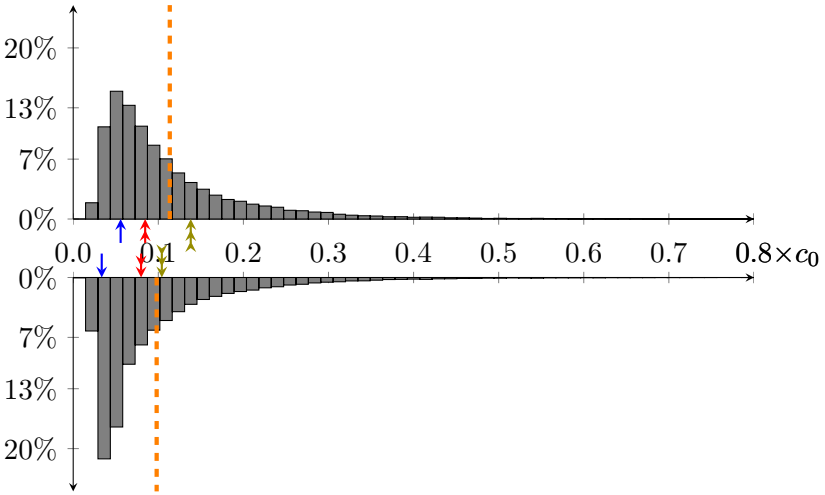


Figure 2.10: Statistical distribution of the errors on the predictions of our least square model, trained on the **NoisyTrainingDataSET** and tested again the **NoisySmallTestDataSET**. The dashed lines represent the average value of the error, while the arrows point to the 25, 50, and 75-percentile of the error on the top histogram, and show how these change when removing the average in the bottom histogram.

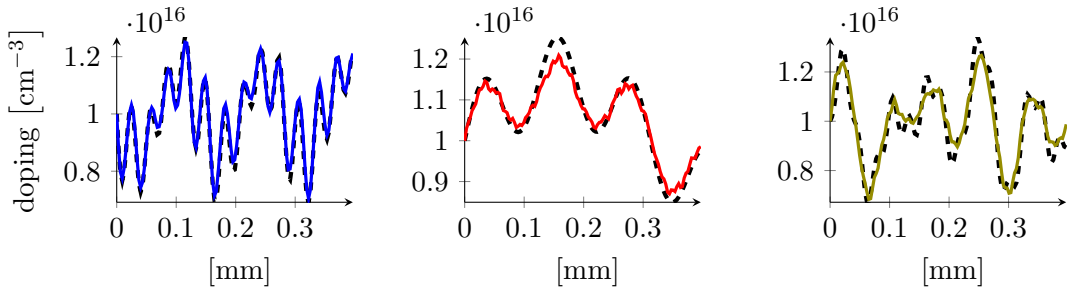


Figure 2.11: Three examples of predictions obtained using the least squares model applied to the **CleanDataSET**. The dashed gray line is the expected result, the continuous colored lines are our predictions. The three plots represent samples whose error is close to the 25, 50 and 75-percentile (from left to right), and corresponds to the three arrows in the left histogram in Figure 2.8 of the same color.

Next, we check how robust our networks are with respect to noise. As a first test, we use the linear model generated with the **CleanTrainingDataSET** to predict the doping of the samples in the **NoisyTestDataSET**. Since the networks have not been trained with noise, we observe a significant deterioration in the quality of the predictions, shown in Figure 2.9, where the average error is now around 20%.

We believe that the main reason why the error increases so much is the fact that the noise introduces a shift in the average of the doping function that can not be physically estimated by our setup. In other words, eq. (2.15) defines the admissible doping profiles with an average of (roughly) equal to $c_0 = 1.0 \times 10^{16}\,\mathrm{cm}^{-3}$ on the overall domain. When we introduce noise, this assumption does not hold anymore and we have no way to predict whether the average of the doping is the one we expect in the entire crystal sample. This is also coherent with the qualitative analysis in eq. (2.16), where we show that the value of the current is mostly related to the local variation of the doping and not to its average value.

Removing the average from both the output of the model, and from the reference doping during testing leads to the results in the bottom plot of Figure 2.9, where the error drops from 20% to 15%. Indeed, we expect still a larger error w.r.t. to the histogram presented in Figure 2.8, since the test dataset includes noisy data which are not included in the training dataset. A few examples of predictions of the doping profile on Σ_h with noisy input data are shown in Figure 2.12.

An improvement of the error for the least square problem can be obtained by performing the training on the **NoisyTrainingDataSET**. The histogram of the error on the **NoisyTestDataSET** is shown in the right plot of Figure 2.10, where the error is now around 10% (see Table 2.2 for the details). It becomes apparent that training with noisy doping profiles also significantly reduces the need to remove the average doping value.

In summary, the least square model is able to predict doping profiles with an average error of around 10% for clean test data or noisy but properly average adjusted test data.

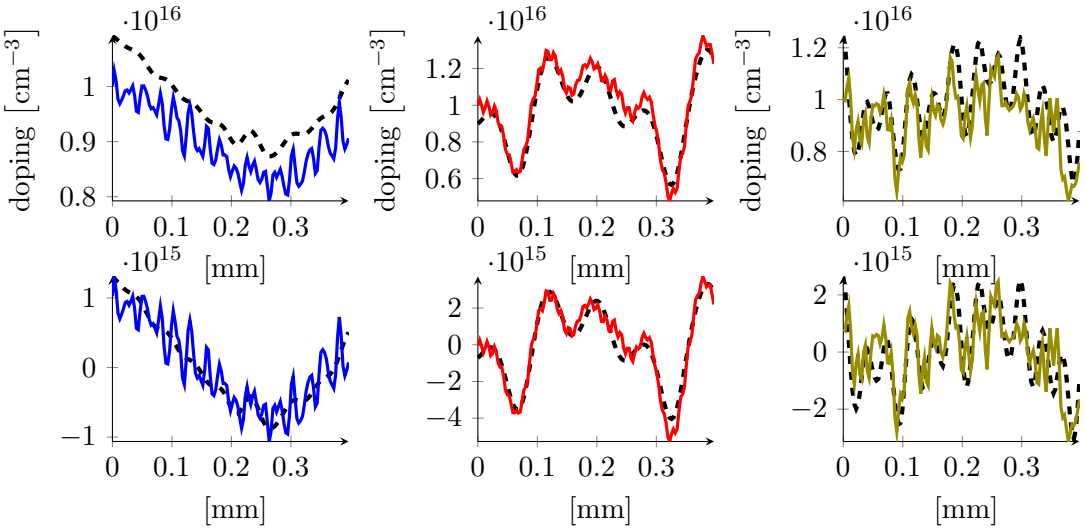


Figure 2.12: Three examples of predictions on the **NoisyTestDataSet**, using the least square model trained on the **CleanTrainingDataSet**, taken from the 25, 50, and 75 percentile of the error (from left to right), without removing the average of the doping (top), and removing the average (bottom). The samples in the plots have an error that corresponds to the arrows in the histograms in the middle of Figure 2.9 (including the color).

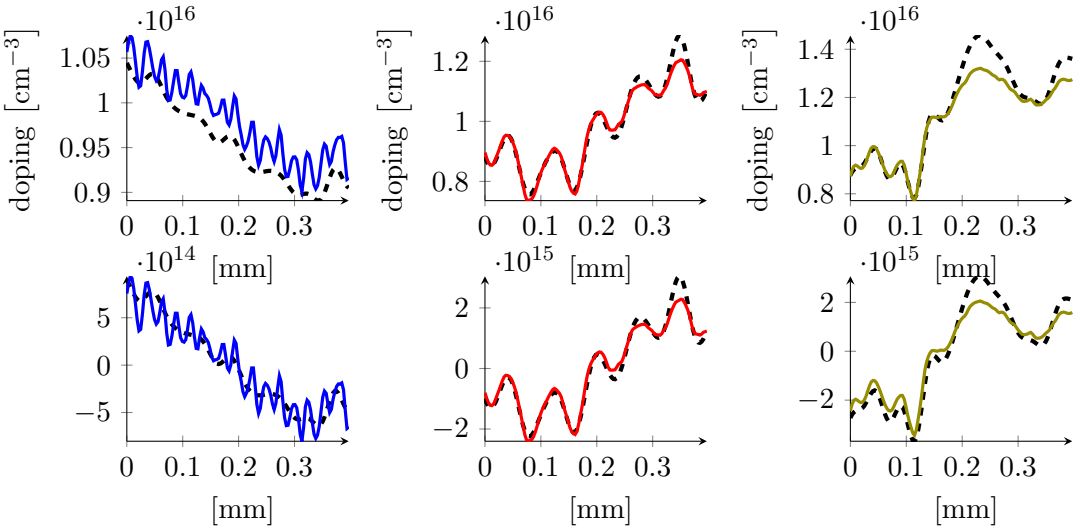


Figure 2.13: Three examples of predictions on the **NoisyTestDataSet**, with the least square model trained on the **NoisyTrainingDataSet**, taken from the 25, 50, and 75 percentile of the error, without removing the average (left), and removing the average (right). We note that the errors of plots in the top (bottom) row correspond to the arrows pointing the errors in the upper (lower) histogram in the right of Figure 2.10.

However, it is possible to further improve the results by introducing nonlinearities in our models, for example using neural networks. We focus on multilayer perceptrons and residual neural networks in the following sections.

2.4.6 Multilayer perceptron

While the least square approach is a good starting point, its efficiency is only good if the inverse operator is close to a linear operator. For inverse problems associated to the van Roosbroeck system, this is not necessarily the case, and one may choose to introduce some nonlinearities in the approximation of F_h. Multilayer perceptrons [23] are among the most widely used feedforward neural networks, and they are the first natural candidate for general nonlinear function approximations. We consider networks consisting of a down-sampler (L_1), a multilayer perceptron with six layers ($L_2 \dots L_7$), and an up-sampler (L_8). The number of neurons of the input layer of the MLP (which coincides with the size of the output of the down-sampler) is chosen from the set $\{100 + i50 : i = 0, \dots, 8\}$. The number of neurons of the MLP output layer L_7, which in principle can be different from the previous one, is chosen from the same set.

Let $\#(L_i)$ denote the number of neurons in L_i. For each hidden layer L_i of the MLP with $i = 3, \dots, 6$, we randomly choose $\#(L_i)$ in the set $\{\#(L_{i-1}), \#(L_{i-1}) \pm 50, \#(L_{i-1}) \pm 100, \#(L_{i-1}) - 200\}$ with the constraint that $\#(L_i) > 0$. In total, there are $71,118$ admissible configurations. There are fewer than $6^4 \cdot 9^2$ configurations because some of them would lead to a negative amount of neurons. The multilayer perceptrons are implemented in PyTorch [39].

For a given configuration of the neural network, we apply the stochastic gradient descent (SGD) algorithm (without momentum and weight decay) to find the MLP that minimizes the mean square error loss defined in Equation (2.13). For each network configuration, we fix the learning rate to a value that is chosen randomly in the interval $[10^{-3}, 1]$ and the batch size to 64. The number of training epochs is set to $1,000$.

We randomly selected $10,000$ configurations where we vary both the structure of the MLP (choosing one of the 71,118 possible MLPs) as well as the learning parameters of SGD. To reduce the computational cost associated with the training of all the resulting neural networks, we apply the Asynchronous Successive Halving algorithm (ASHA) [31], a multi-armed bandit algorithm that has been optimized to run on a massive amount of parallel machines. The ASHA algorithm discards the worst 50% performers of the neural network population at each check point, and proceeds with the training only for the most promising neural networks. Our implementation is based on the Ray library [37], which also takes care of distributing the jobs and coordinating the execution among different nodes during the computation.

The multilayer perceptron with the smallest ℓ^∞ testing error defined in Equation (2.14) (trained with **CleanDataSET** and tested with **CleanTestDataSET**), has six layers with $250, 250, 150, 100, 1000, 350$ nodes, a batch size of 64, and a learning rate of 0.06. The statistical distribution of the error is reported in of Table 2.3, and depicted in Figure 2.14. The ℓ^∞ error is, on average, around 4.67%. Even with such a simple neural network, we reduce the average error with respect to the least square model by of a factor two. Also,

Train	Test	w/o mean	Average	25-th percentile	Median	75-th percentile
clean	clean		4.67%	1.03%	1.83%	3.98%
clean	noisy		21.2%	91.1%	17.3%	29.3%
clean	noisy	∗	13.7%	4.15%	8.89%	19.7%
noisy	noisy		5.92%	2.14%	3.60%	6.20%
noisy	noisy	∗	4.96%	1.44%	2.49%	4.67%

Table 2.3: Absolute ℓ^∞ errors w.r.t $c_0 = 1.0 \times 10^{16}\,\mathrm{cm}^{-3}$ of the MLP model, corresponding to the colored arrows as well as dashed lines showing in Figures 2.8 to 2.10. The third column describes if we remove or not the mean from the prediction and from the expected result before computing the error, exactly as in Table 2.2.

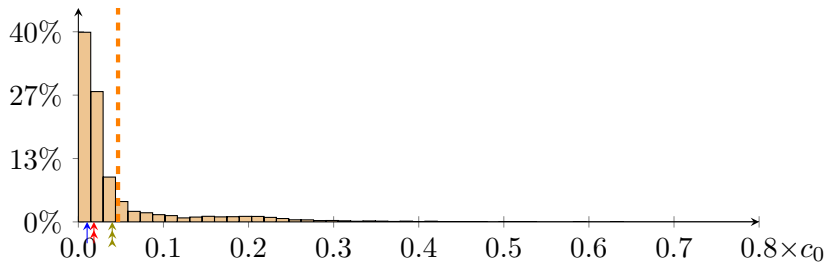


Figure 2.14: Statistical distribution of the errors on the predictions of our best multilayer perceptron model, trained on the **CleanTrainingDataSET** and tested on the **CleanValidationDataSET**. The dashed lines represent the average value of the error, while the arrows point to the 25, 50, and 75-percentile of the error of the histogram.

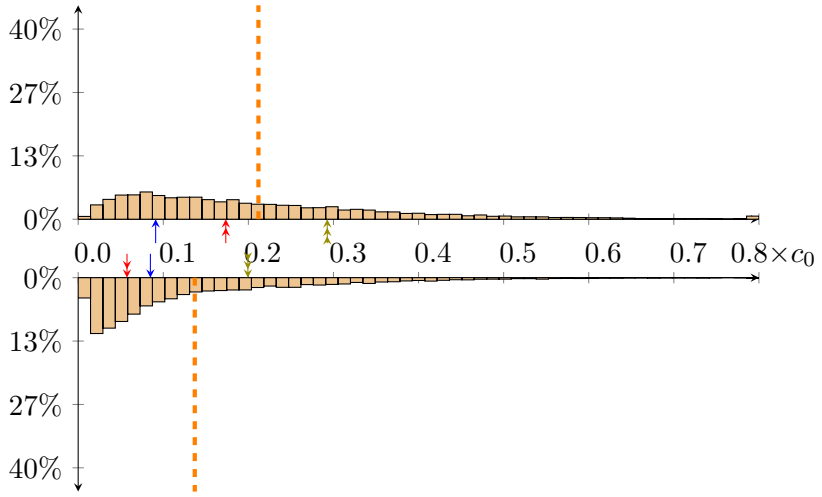


Figure 2.15: Statistical distribution of the ℓ^∞ errors on the predictions of our best multilayer perceptron model, trained on the **CleanTrainingDataSET** and tested again the **NoisySmallTestDataSET**. The bottom histogram is obtained by removing the average value of the doping. The dashed lines represent the average value of the error, while the arrows point to the 25, 50, and 75-percentile of the error on the top histogram, and show how these change when removing the average in the bottom histogram.

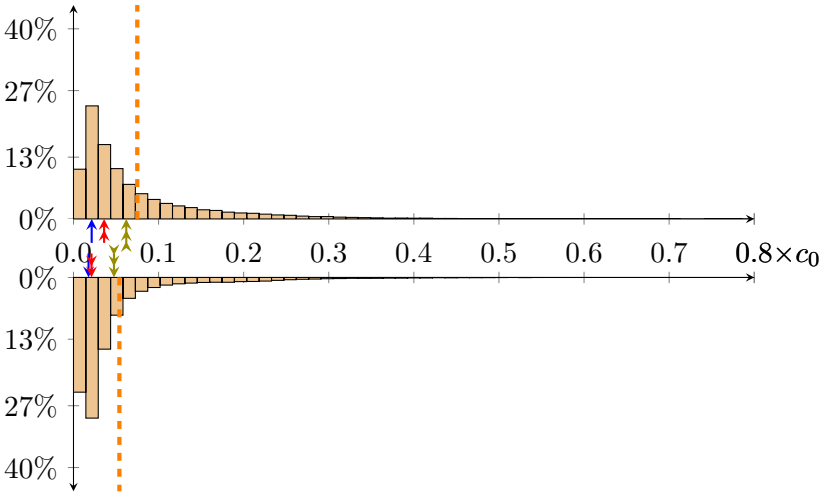


Figure 2.16: Statistical distribution of the ℓ^∞ errors on the predictions of our best multi layer perceptron model, trained on the **NoisyTrainingDataSET** and tested again the **NoisyTestDataSET**. The bottom histogram is obtained by removing the average value of the doping. The dashed lines represent the average value of the error, while the arrows point to the 25, 50, and 75-percentile of the error on the top histogram, and show how these change when removing the average in the bottom histogram.

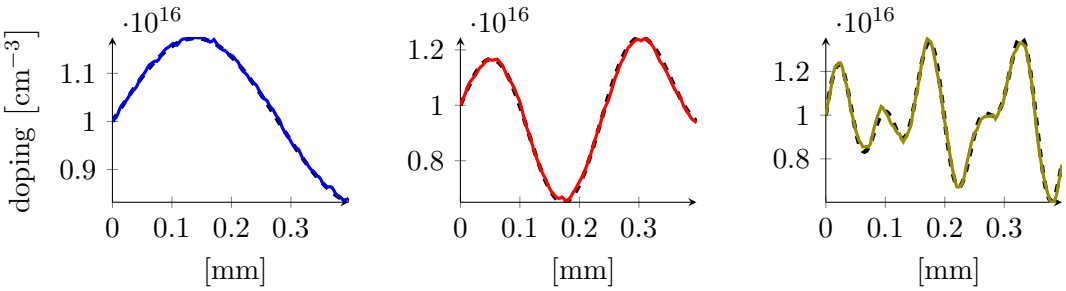


Figure 2.17: Three examples of predictions obtained using our best MLP model applied to the **CleanDataSET**. The dashed gray line is the expected result, the continuous colored lines are our predictions. The three plots represent samples whose error is close to the 25, 50 and 75-percentile (from left to right), and corresponds to the arrows with the same color in the histogram in Figure 2.14.

we obtain a much better statistical distribution, clearly shown in Figure 2.14.

Furthermore, the resulting MLP is robust to noise, provided that we properly filter the average of the doping profile. A comparison between the statistical distribution of the errors (both with and without average) as well as the corresponding statistical distributions are shown in Table 2.3 and in Figure 2.15. Testing with the **NoisyTestDataSET** while the MLP was trained with **CleanDataSET** shows (as expected) a large increase in the average error.

More robust results with respect to noise can be obtained by training again the best MLP on the full **NoisyTrainingDataSET**. We do not perform an additional hyperparameter tuning, but simply repeat the training stage on the same MLP. In this case, the larger dataset and the presence of noise induce an error when tested with the **NoisyTestDataSET** of around 5.92%. This error can, again, be improved by removing the average as we did before, leading to a final error which is very close to the clean case (4.96% vs 4.67%). We show three doping reconstructions for this case in the 25, 50, and 75 percentile in Figure 2.18.

In summary, introducing a nonlinear MLP model has reduced the error roughly by a factor of two compared to the linear least square model. The average MLP is just below 5% for clean test data or noisy but properly average adjusted test data.

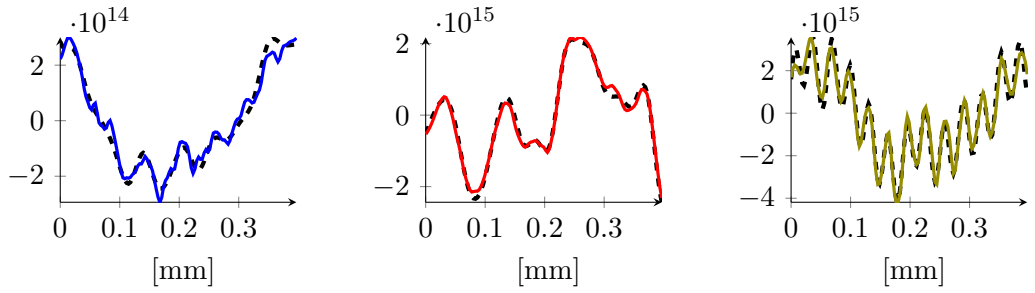


Figure 2.18: Three examples of predictions on the **NoisyTestDataSet**, with the best MLP model trained on the **NoisyTrainingDataSet**, taken from the 25, 50, and 75 percentile of the error, removing the average. The errors of the plots correspond to the arrows in the lower histogram of Figure 2.16.

2.4.7 Residual neural network (ResNet)

In 2015, Kaiming et al.[25] introduced ResNets, feedforward convolutional neural network models, which, since then, have been widely used to solve different kinds of problems. One of the biggest advantages of ResNets (or, generally, of convolutional neural networks) over the multilayer perceptrons is their ability to learn 2D images. Interpreting the doping profiles in 2D probing region as an image, this is precisely what we need to achieve, too. Moreover, the proportionality shown in Equation (2.17), valid under additional assumptions, suggests that the photovoltage signal and the doping profile may be related by convolutions. Even assuming now a nonlinear relationship described via ResNets, we may still wish to exploit a convolutive structure to encompass the relationship in Equation (2.17). Our reference implementation for residual neural networks (ResNet) is described in [25], with some important differences regarding the structure of the network.

First, in our LPS setup from Section 2.4.1, we consider one-dimensional data vectors $\mathbf{C}$ and $\mathbf{u}$ instead of two-dimensional images. The dimension is particularly relevant to choose the size of the network: In ResNets, the downsampling blocks reduce by a factor of two the size of the input, by first halving the size of the input in each direction (reducing the spatial indices by a factor four) and then doubling the number of channels. In our setup, this dimension reduction associated to the downsampling block does not happen because in the first step we only have one dimension to divide by two, and we end up with the same number of neurons after doubling the numbers of channels. Moreover, while the ImageNet database used to train the network described in [25] contains about one million images, our datasets are significantly smaller. We expect to face some overfitting and therefore we aim for a model with fewer parameters than the one described in [25]. Finally, we solve a regression problem instead of a classification one. The major consequences for our model are that we cannot use drop-out layers to reduce overfitting, and it is unclear if there is any benefit in using batch normalization layers. We keep the batch normalization layers because we expect the statistical distribution of our batches to be similar to each other, and the statistical distribution of the error for a simpler

network (the MLP) shows an improvement in the performance of the model with noise when removing the average, suggesting that batch normalization layers are at least not harmful.

We build several ResNet models using PyTorch ([39]), and develop a strategy to find the best ones by adapting the model described in [25], and using the insight obtained from the MLP model presented in Section 2.4.6. Our best multilayer perceptron had 160,950 parameters that had to be optimized. We try to keep the number of parameters of our ResNet around the same order of magnitude. Of course, this means that we need to drastically simplify the model developed in [25], to find a model whose number of parameters is compatible with the size of our **CleanTrainingDataSET**.

Here we describe the allowed configurations that we have implemented. All of our ResNet models are preceeded by a down-scaling interpolation layer, as we did in the MLP model, with the constraint that the number of output elements is a power of 2; this ensures that the downscaling blocks of the ResNet always deal with an even number of neurons: we use an interpolation layer that goes from $n = 1200$ to 256 points. This number as been chosen because it is the closest power of two respect to the size of the first layer of our best multilayer perceptron defined in Section 2.4.6.

Similarly, the last layer of the ResNet is followed by an up-scaling interpolation layer, going from 256 to 1200. As far as the components of the ResNet are concerned, these are the possible configurations that we allow for each ResNet element:

Gate: This is the first layer of our network (after the down-sampling layer). It consists of a convolutional layer, a batch normalization layer, and an activation layer. In the convolutional layer, we choose the kernel sizes from the set $\{3, 5, 7, 9\}$. The number of output channels, instead, is chosen from the set $\{8, 16, 24, 32\}$. Finally, the stride of the convolution layer is chosen from the set $\{1, 2, 4\}$. This leads to $4 \times 4 \times 3 = 48$ possible convolutional networks. Taking into account the fact that the activation layer always applies a ReLU (that does not require parameters) and that also the normalization layer is fixed, we have a total of 54 possible configurations of our gate.

Encoder: This is built by stacking several blocks of the same type. We consider two different kinds of blocks: the "basic blocks" described in [25] and the "fixed channel block". Both blocks are made of the following layers: a convolutional layer, a batch normalization, a ReLU activation layer, another convolution, and, finally, a normalization. The two convolutional layers have a fixed kernel size of three: they do not have bias and the padding is chosen so that the size of the output is preserved (therefore, in our case, the padding is equal to 1).

Each block can be configured to perform a "downsampling", i.e., it may halve the size of its output. When downsampling is active, basic blocks increase the number of channels by a factor two, while the fixed channel blocks do not. In this case the shortcut operation of both blocks is a convolutional layer with kernel size equal to one and stride equal to two (basic block), or kernel size equal to two and stride equal to two (fixed channel block), followed by a normalization block. The convolutional layer of the shortcut of the fixed

channel block forbids communication between channels, i.e., each element of the output tensor depends only on the values of the elements of the input tensor that share the same index for the channel (or, in other words, using PyTorch we impose that the number of groups of the convolutional layer is equal to the number of channels). Concerning the computation of the output (and not the shortcut), the dimensional reduction is obtained by setting the stride of the first convolutional layer to 2.

When downsampling is not active, the shortcut operation of both blocks is the identity tensor. Therefore, the only free parameters that we have left for our encoder are the number of blocks, their kind, and a downsampling flag for each block. Our encoders consist of one, two, or three blocks of the same kind. For encoders made of basic blocks, we allow two different configurations of the downsample flag: true for all blocks or false for all blocks. For fixed channel blocks, we always set the downsample flag to true. We have a total of 6 configurations for the encoders with the basic blocks and 3 configurations that use the fixed channels blocks, for a total of 9 possible configurations.

Decoder: This is essentially a multilayer perceptron, and we allow 1 or 2 hidden layers. If we choose a configuration with 1 layer, the size of this layer could be 100, 150 or 200. With 2 layers, there is a total of 15 possible configurations obtained by choosing the size of the first layer in $\{100, 150, 200, 250, 300\}$ and the second one in $\{100, 150, 200\}$. In total, we have therefore 18 different configurations for the decoder.

Now that we have described the different possibilities for the architecture of the model, we must spend some words about the training procedure. Since training a ResNet is significantly more expensive compared to training a multilayer perceptron, we use a two-step strategy to reduce the computational cost of the hyperparameter tuning.

In the first step, we choose the model structure and some parameters for the SGD minimizer. The possible configurations of the ResNet that we allow correspond to a total of 48 configurations for the gate, 9 configurations for the encoder, and 18 configurations for the decoder, for a total of 7776 possible configurations. For the minimizer, we choose the learning rate in the interval $[5 \times 10^{-3}, 1 \times 10^{-1}]$ and the batch size in the set $\{64, 128, 256, 384, 512\}$. Other relevant parameters, like the gradient clipping and the weight decay, are fixed to 1 and 0 (respectively). In this first run, we choose randomly 1800 configurations by choosing 300 different models out of the 7776 we proposed and coupling each one of them with 6 sets of random parameters for the SGD minimizer chosen accordingly with the constraints described above. Again, we rely on the ASHA algorithm as we did in Section 2.4.6. We compare the performances of the models after 5000 epochs, and retain the best four configurations selected by the ASHA algorithm, whose ResNet models are described in Table 2.4.

In the second stage of the hyperparameter tuning, we keep the four structures of the ResNet models fixed, and enlarge the search space in the optimization parameters by defining a new set of admissible parameters: we choose the learning rate in the interval $[10^{-3}, 10^{-1}]$, the batch size in the set $\{32, 64, 128, 256, 384, 512\}$, the gradient clip in $\{10^{-2}, 10^{-1}, 1\}$, and we introduce some weight decay, with decay parameter chosen in

	RN1	RN2	**RN3**	RN4
N. of channels	24	16	**24**	16
Gate conv. kernel size	9	5	**7**	3
Gate conv. stride	4	4	**4**	4
N. of encoder blocks	3	3	**3**	3
Block type:	FixedChannel	Basic	**FixedChannel**	FixedChannel
Downsampling:	`True`	`True`	`True`	`True`
Size of decoder layers	(100, 200)	(150, 150)	**(200, 200)**	(250, 200)
N. of parameters	102,188	324,048	**141,440**	138,994

Table 2.4: The four ResNet models (RN1–RN4) identified in the first stage of the hyperparameter tuning by the ASHA algorithm. RN3 is the one selected in the second stage.

the set $\{0, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$.

For each one of the four models, we perform 200 different trainings applying different parameters of the SGD optimizer, and finally select a winner, corresponding to RN3 in Table 2.4.

The role of the different parameters on the training is shown in Figures 2.19 and 2.20. From Figure 2.20b, for example, we see that the weight decay, if used at all, must be chosen with a really small coefficient. In particular, we see that, when the weight decay is chosen equal to 10^{-1}, all the models commit an error that is 5 times bigger compared to the one of our best candidate. Figure 2.20a, instead, seems to suggest to choose big batch size, like 250 or 512. Finally, Figure 2.20c shows that the parameter chosen for the gradient clip is not so relevant.

In Figures 2.21 to 2.23 we present the statistical distribution of the error for ResNet RN3 selected with the strategy outlined above. We observe that the average error we obtain using the **CleanDataSET** for both training and testing is around 5.47%. In Table 2.5 we report the errors for the ResNet model for the different trainings (with or without noise). For comparison, in that table we also report the errors of the other two methods, the least squares model and the MLP. There, we see that the performance of the ResNet when we do not introduce the noise is slightly worse compared to the MLP case introduced in Section 2.4.6. Moreover, we observe that the ResNet RN3 trained with the **CleanDataSET** is more sensitive to noise (see Figure 2.22) when compared to the MLP (29.7% vs 21.2%). This difference decreases when removing the average, but for the ResNet remains surprisingly worse than the least square model (16% vs 15.5%).

When we keep the structure of RN3, and train the network on the **NoisyTrainingDataSET**, we obtain the results shown in Figure 2.23 on the right. An example of reconstruction obtained with RN3 with a sample from the **NoisyTestDataSET** is shown in Figure 2.24. In this case the average error on the **NoisyTestDataSET** is around 6.72%, which is slightly worse than the result obtained with the MLP network.

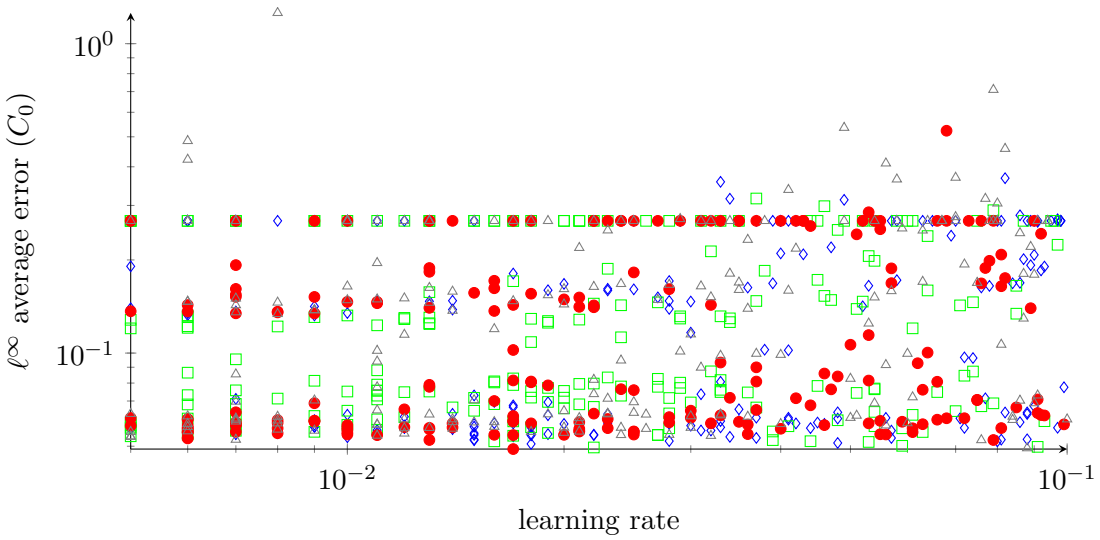


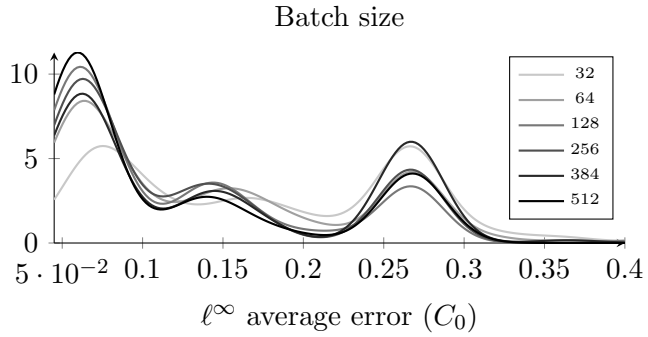
Figure 2.19: ℓ^∞ errors of our models with respect to the learning rate of the optimizer. The colors correspond to the networks RN1-RN4, as seen in Table 2.4.

2.5 Outlooks

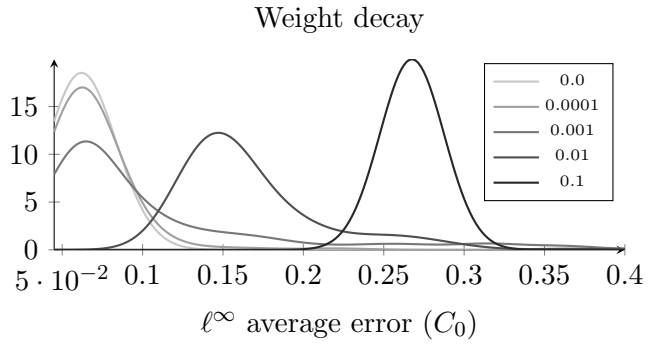
From Table 2.5, we see that, if we do not take into account the presence of noise, the linear least square method yields an average absolute ℓ^∞ error of 9.3%, while the nonlinear networks roughly halve this error to 4.7% and 5.5%, respectively, after optimizing relevant hyperparameters. This shows that the linear approximation is relatively good, but that some parts of the problem are intrinsically nonlinear and require a more advanced method to be taken into account.

When we perturb the datasets, our methods turned out to be robust with respect to noise, provided that training is repeated with larger, noisy, datasets. In this case, the error is around 9.8%, 5%, and 6.7% respectively. Removing the average doping value from the data was more important to reduce the testing error for clean datasets than for noisy ones. The datasets, python codes, and resulting trained networks are available in the repository [41].

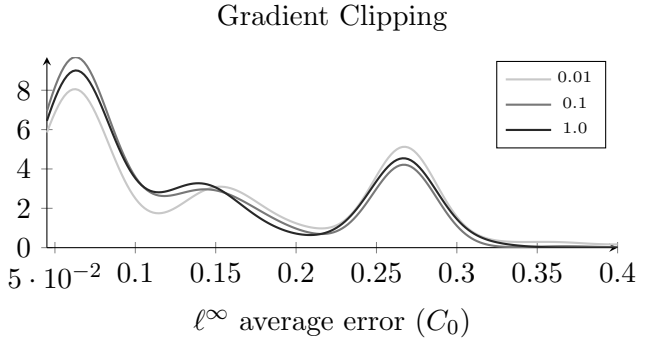
It is clear that different doping profiles may correspond to similar signals. On the one hand, this is intrinsically dependent on the technology that we apply to recover the signal: we hit the crystal with a laser that has a finite laser spot radius (in our case, around 20 mm), and we cannot expect to resolve oscillations with smaller wave lengths. On the other hand, our models do not have a way to distinguish between two doping profiles that deliver the same signal, but are exposed, during training, to many such cases. In Figure 2.25, for example, we show two examples of doping profiles that have errors on the far right regions of the histograms in Figures 2.14 and 2.21. The figure shows how in some cases, the inverse problem is oblivious to higher oscillations, and both neural network types will have cases in which they return an answer which is either too smooth



(a) Density plots respect to the batch size



(b) Density plots respect to the weight decay



(c) Density plots respect to the gradient clip

Figure 2.20: In each one of these plots, we show "how many" models obtain a particular ℓ^∞ average error on the test dataset when fixing a specific value for one of the parameters. These density plots are computed via the `KernelDensity` function of `sklearn.neighbors` with a Gaussian kernel and a bandwith of 0.02. Darker line colors correspond to bigger parameter values (batch size, weight decay and gradient clip).

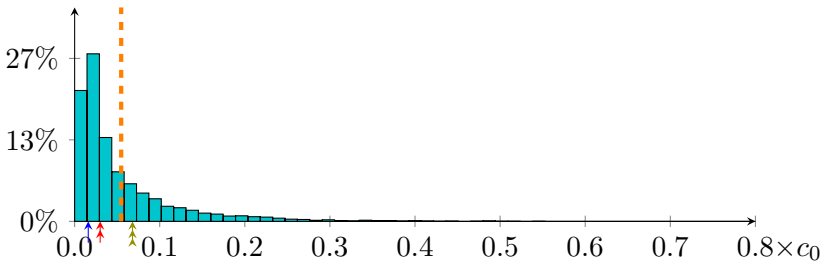


Figure 2.21: Statistical distribution of the errors on the ResNet model described in the third column of Table 2.4 (RN3), trained on the **CleanTrainingDataSET** and tested on the **CleanValidationDataSET**. The dashed lines represent the average value of the error, while the arrows point to the 25, 50, and 75-percentile of the error of the histogram.

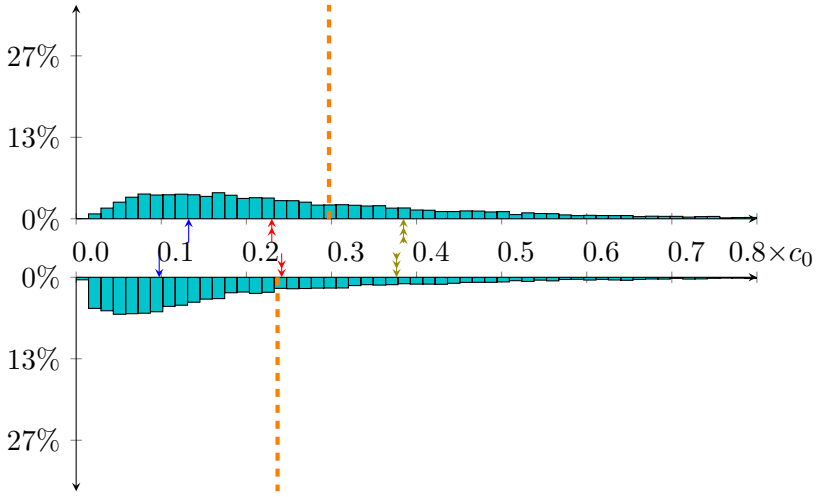


Figure 2.22: Statistical distribution of the errors on the ResNet model RN3, trained on the **CleanTrainingDataSET** and tested on the **NoisySmallTestDataSET**. The bottom histogram is obtained by removing the average value of the doping. The dashed lines represent the average value of the error, while the arrows point to the 25, 50, and 75-percentile of the error on the top histogram, and show how these change when removing the average in the bottom histogram.

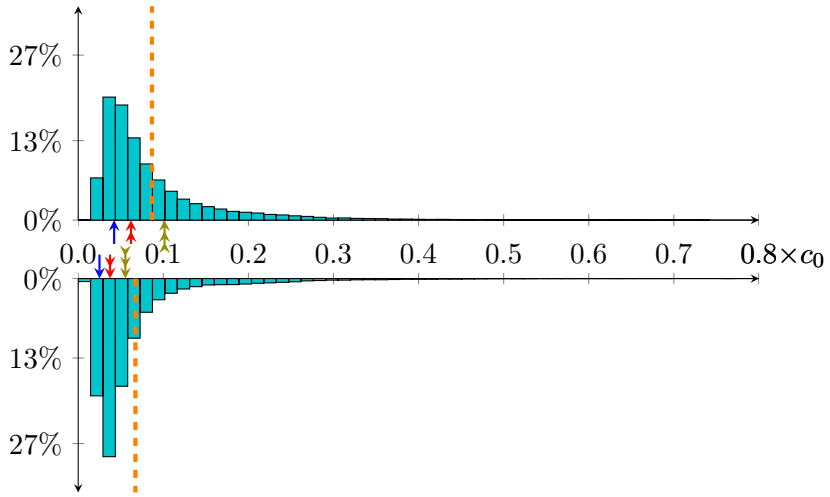


Figure 2.23: Statistical distribution of the errors on the ResNet model RN3, trained on the **NoisyTrainingDataSET** and tested on the **NoisyTestDataSET**. The bottom histogram is obtained by removing the average value of the doping. The dashed lines represent the average value of the error, while the arrows point to the 25, 50, and 75-percentile of the error on the top histogram, and show how these change when removing the average in the bottom histogram.

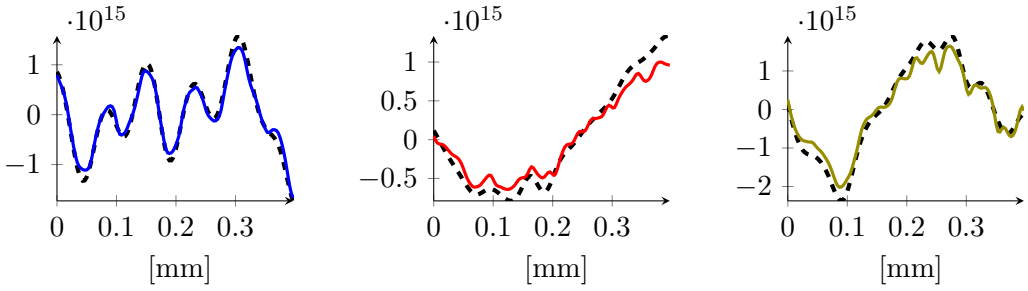


Figure 2.24: Three examples of predictions on the **NoisyValidationDataSet**, with the best ResNet RN3 model trained on the **NoisyTrainingDataSet**, taken from the 25, 50, and 75 percentile of the error, removing the average of the doping. The colors of the plots indicate the arrows in histogram of Figure 2.23.

Absolute error ($\times C_0$)		Average	25-th percentile	Median	75-th percentile
LS	**C.C.**	9.32%	4.15%	6.53%	1.16%
	C.N. w/ mean	21.5%	11.5%	17.9%	28.4%
	C.N. w/o mean	15.5%	6.75%	10.5%	21.0%
	N.N. w/ mean	11.4%	5.72%	8.46%	13.8%
	N.N. w/o mean	9.81%	4.20%	6.47%	11.9%
MLP	**C.C.**	4.67%	1.03%	1.83%	3.98%
	C.N. w/ mean	21.2%	91.1%	17.3%	29.3%
	C.N. w/o mean	13.7%	4.15%	8.89%	19.7%
	N.N. w/ mean	5.92%	2.14%	3.60%	6.20%
	N.N. w/o mean	4.96%	1.44%	2.49%	4.67%
ResNet	**C.C.**	5.47%	1.60%	3.00%	6.80%
	C.N. w/ mean	29.7%	13.2%	22.3%	38.5%
	C.N. w/o mean	16.0%	5.27%	10.4%	21.2%
	N.N. w/ mean	8.67%	4.22%	6.20%	10.1%
	N.N. w/o mean	6.72%	3.15%	4.44%	7.06%

Table 2.5: Absolute errors w.r.t $c_0 = 1.0 \times 10^{16}\,\mathrm{cm}^{-3}$ corresponding to the colored arrows as well as dashed lines showing in Figures 2.8 to 2.10, 2.14 to 2.16 and 2.21 to 2.23. Here the second column categorized the training and testing datasets mentioned in the captions of the histogram figures. For instance, the row for "**C.N.** w/ mean" in the block of MLP shows the errors in the top histogram in Figure 2.15 which is trained/tested on **CleanTraining DataSET/NoisySmallTestDataSET**.

w.r.t. the expected result, (left panel of Figure 2.25) or too oscillatory (right panel of Figure 2.25).

A possible focus of future research could be to study how to resolve this ambiguity, which is intrinsic to the ill-posedness of the discrete local photovoltage inverse problem.

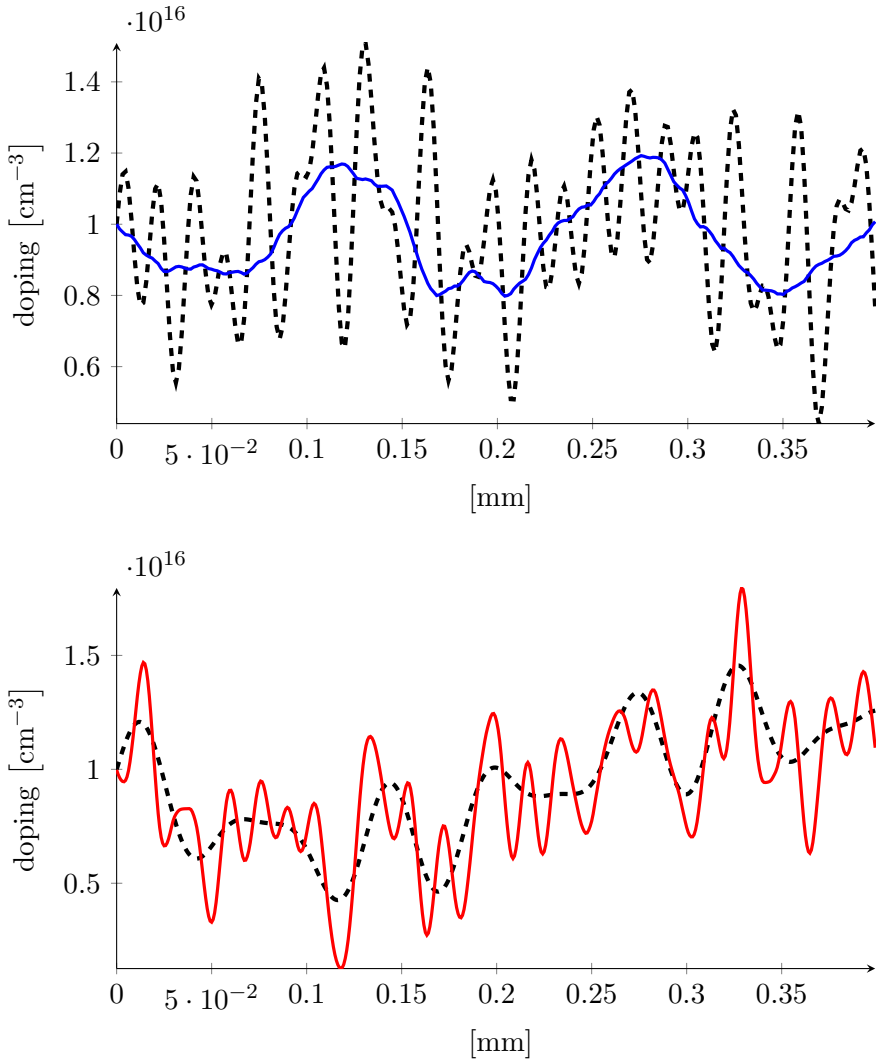


Figure 2.25: An example of an outliers using the MLP model (left) and the ResNetmodel RN3 (right). The dashed line represents the expected doping profile sample, and the continuous line is the output of the neural networks.

Chapter 3

Hybridizable Discontinuous Galerkin Methods

3.1 Introduction

In the previous chapters we introduced the van Roosbroeck system, together with some numerical techniques tailored for its numerical solution. In particular, in Section 1.2 we introduced the Scharfetter–Gummel scheme which is able to generate, even for extremely high Péclet numbers (i.e., when transport dominates diffusion), accurate results. Unfortunately, the Scharfetter–Gummel scheme is a low order method. In Section 2.2.2, we pointed out the need for methods that are able to reduce the computational cost of the problem.

In this chapter we focus on the Hybridizable Discontinuous Galerkin methods (HDG), a family of high-order DG methods (described in Section 3.3.1) and we show a connection between these schemes and the Scharfetter–Gummel scheme.

In Section 3.4 we take into account HDG of degree 0, i.e., methods where the solution is defined as a piecewise constant function that assumes one single value for each cell of the triangulation. We show that, under some strict hypothesis, a specific choice of the stabilization parameter leads to a method that is equivalent to the Scharfetter–Gummel scheme, and that recovers exactly the analytical solution (without introducing any error) when the diffusion and drift coefficients are constants.

Finally, in Section 3.5 we extend the result we obtained for degree 0 in section 3.4 to an arbitrary degree $k \in \mathbb{N}$.

3.2 Model problem

In this chapter we focus on a general problem that can be described in differential form as

$$\nabla \cdot \mathbf{J}(u) = f, \tag{3.1}$$

where u is the transported quantity, $\mathbf{J}(u)$ is its flux, and f is a given generation term (which may or may not be dependent on u).

Many physical phenomena that describe the transport of some quantity, or more in general, conservation laws, may be described by eq. (3.1): in particular, in Chapter 1 we

have used this equation to model the electron and hole densities inside the van Roosbroeck system. Often, $\mathbf{J}$ takes the specific form:

$$\mathbf{J}_u \stackrel{\text{def}}{=} -\alpha\nabla u + u\boldsymbol{\beta}, \tag{3.2}$$

where the coefficients $\boldsymbol{\beta}$ and α describe two of the most common effects in physical phenomena, respectively, convection and diffusion. In many other problems, $\mathbf{J}$ is not linear and obtaining a numerical approximation requires the solution of (sequences) of linear subproblems with the same form described by eq. (3.2).

According to the specific application one has in mind, the model problem above may be discretized using a single variable u, resulting in a second order elliptic equation (primal or standard formulations), or it may be discretized by splitting it into two first order equations, explicitly introducing additional intermediate variables, like $\mathbf{J}_u$ or $\mathbf{W}_u \stackrel{\text{def}}{=} -\nabla u$ (mixed formulations).

Our goal is to exploit some well known properties of the finite volume discretization of the standard formulation (and, in particular, of the Scharfetter–Gummel scheme exposed in Section 1.2) to gain additional insights on stabilization mechanisms for Hybridizable Discontinuous Galerkin (HDG) formulations, which are naturally written in mixed formulations.

3.2.1 Primal formulation

Let $\Omega \subset \mathbb{R}^n$ be a Lipschitz domain with boundary $\Gamma \stackrel{\text{def}}{=} \partial\Omega$, and let

$$\alpha \in L^\infty(\Omega)\,, \qquad \boldsymbol{\beta} \in [L^\infty(\Omega)]^n, \qquad f \in L^2(\Omega) \tag{3.3}$$

be three assigned functions over Ω. We assume that α is a piecewise constant function, i.e., that there exists a finite number m of Lipschitz domains $\Omega_1, \ldots, \Omega_m$, with $\Omega_i \subseteq \Omega$ and

$$\Omega \subseteq \bigcup_{i=1}^{m} \overline{\Omega_i}, \tag{3.4}$$

such that $\alpha_{|\Omega_i}$ is a constant function for each $i \in \{1, \ldots, m\}$. Furthermore, we assume that there exists a number $\alpha_0 > 0$ such that $\alpha(x) > \alpha_0$ for every $x \in \Omega$.

The standard, or primal, formulation then reads:

Problem 1 (Primal formulation)**.** *Given a source term $f \in L^2(\Omega)$, and a boundary data $u_0 \in H^{1/2}(\Gamma)$, find $u \in H^1(\Omega)$ such that*

$$\begin{aligned} &\nabla \cdot (-\alpha\nabla u + u\boldsymbol{\beta}) = f \quad && \text{in } \Omega, \\ &u = u_0 && \text{on } \Gamma. \end{aligned} \tag{3.5}$$

Under some regularity conditions for coefficients α and $\boldsymbol{\beta}$, existence and uniqueness of a solution to Problem 1 is well known, and can be found in any text book of functional analysis (see, e.g., [42]).

3.2.2 Mixed formulations

When one is interested in derived quantities, like computing the flux across a boundary surface, or in some part of the domain, direct discretisations of Problem 1 may result in poor approximation properties of the derived quantities, and one may resort to approximating derived quantities explicitly.

For drift-diffusion problems, there are different ways to choose a mixed formulation. Here we present the two most commonly used ones, i.e., a *displacement* based formulation, and a *flux* based formulation.

Problem 2 (Displacement mixed formulation)**.** *Given a source term $f \in L^2(\Omega)$ and a boundary data $u_0 \in H^{1/2}(\Gamma)$, find $(\mathbf{W}_u, u) \in H(\mathrm{div};\, \Omega) \times H^1(\Omega)$ such that*

$$\begin{cases} \mathbf{W}_u + \alpha \nabla u = 0, & \text{in } \Omega, \\ \nabla \cdot (\mathbf{W}_u + u\boldsymbol{\beta}) = f & \text{in } \Omega, \\ u = u_0 & \text{on } \Gamma. \end{cases} \tag{3.6}$$

One may also choose as the auxiliary variable directly the definition of the *flux*. In this case, an equivalent mixed problem can be written as

Problem 3 (Flux mixed formulation)**.** *Given a source term $f \in L^2(\Omega)$ and a boundary data $u_0 \in H^{1/2}(\Gamma)$, find $(\mathbf{J}_u, u) \in H(\mathrm{div};\, \Omega) \times H^1(\Omega)$ such that*

$$\begin{cases} \mathbf{J}_u + \alpha \nabla u - u\boldsymbol{\beta} = 0 & \text{in } \Omega, \\ \nabla \cdot \mathbf{J}_u = f & \text{in } \Omega, \\ u = u_0 & \text{on } \Gamma. \end{cases} \tag{3.7}$$

It is worth pointing out that Problem 2 and Problem 3 correspond to the two formulations exposed in Equations (1.2a) and (1.3a) for the van Roosbroeck system.

Given a solution to Problem 1, then the pair $(\mathbf{W}_u \overset{\text{def}}{=} -\alpha \nabla u, u)$ is a solution to Problem 2, and the pair $(\mathbf{J}_u \overset{\text{def}}{=} -\alpha \nabla u + u\boldsymbol{\beta}, u)$ is a solution to Problem 3. When rewriting the problems in their weak form, one need to be more careful, see, e.g., [9].

3.3 Discretizations

In this section we present some discretization techniques to solve numerically the problem described in Section 3.2.

We restrict our exposition to the case where the domain Ω is a polytope. Let $\{\mathcal{T}_h\}_{h>0}$ be a family of quasi-uniform subdivisions of Ω made of simplices with maximum diameter h. We also denote $\mathcal{F}_h$ the collection of faces of $\mathcal{T}_h$, satisfying

$$\mathcal{F}_h = \mathcal{F}_h^i \cup \mathcal{F}_h^\partial,$$

where $\mathcal{F}_h^i$ is the set of the interior faces and $\mathcal{F}_h^\partial$ is the collection of the faces on the boundary. For convenience, we omit the subscript h in the rest of this thesis.

As a reference, we rewrite here the flux of the Scharfetter–Gummel scheme for the general problem we introduced in Section 3.2:

$$\int_F \mathbf{J}_u^{\mathrm{SG}} \cdot \nu \, \mathrm{d}\gamma = \frac{|F| \, \alpha_F}{l_F} \left(B\left(-\frac{\boldsymbol{\beta}_F \cdot \mathbf{v}_F}{\alpha_F} \right) u_1 - B\left(\frac{\boldsymbol{\beta}_F \cdot \mathbf{v}_F}{\alpha_F} \right) u_2 \right), \tag{3.8}$$

where all the notations we introduced in Section 3.2 for F, ν, l_F, B, and $|F|$ still hold and u_i is the degree of freedom on T_i for $i \in \{1, 2\}$. α_F and $\boldsymbol{\beta}_F$ are the values of the functions α and $\boldsymbol{\beta}$ evaluated at the intersection between the hyperplane that contains F and the segment that connects the centers of the cells T_1 and T_2. If α and $\boldsymbol{\beta}$ are not regular enough, this definition must be adapted, for example using some mean values weighted on the distance from the center of the cells as we did in Section 1.2. Finally, the vector $\mathbf{v}_F$ is the vector that goes from the center of the cell T_1 to the center of T_2.

Another common choice for the finite volume method is to discretize the flux using finite difference. For its theoretical relevance, we show here the flux obtained applying this strategy on the model problem we described in Section 3.2:

$$\int_F \mathbf{J}_u^{\mathrm{FD}} \cdot \nu_{T_1} \, \mathrm{d}\gamma = |F| \left(-\alpha_F \frac{u_2 - u_1}{l_F} + \left(\frac{u_2}{2} + \frac{u_1}{2} \right) \boldsymbol{\beta}_F \cdot \nu_F \right). \tag{3.9}$$

3.3.1 The hybridizable discontinuous Galerkin method

Assuming that the solution to (3.6) or (3.7) is smooth enough, our goal is to extend the finite volume schemes (3.8) and (3.9) to some higher order schemes. Since our model problem (3.6) or (3.7) could be convection dominated, i.e. $\|\boldsymbol{\beta}\|_{L^\infty(\Omega)}$ could be large (cf. [11]), a common choice is to consider discontinuous Galerkin methods (DG); we refer to [5] for a general discussion for various DG methods. In terms of the numerical simulation of drift-diffusion problems using DG methods, we refer to [33, 32] for local DG discretisations in one dimensional space and to [14] for a hybridizable DG (HDG) method. Here we will also develop a HDG discretization scheme for (3.6) or (3.7), which was originally introduced in [17] for elliptic problems. One of the advantages of HDG schemes is that, even though the degrees of freedom are defined on both cells and faces, via hybridization (or static condensation), the degrees of freedom on cells can be eliminated from the discrete system. When using a finite dimensional space with higher order polynomials as the approximation space, the size of system matrix is significantly smaller than the ones generated by conventional CG and DG methods. In the following sections, we will introduce classical HDG approaches for the approximation of our model problem as an extension of the FVM scheme (3.9).

We write eq. (3.6) in a weak form as: find $u \in H^1(\Omega)$ and $\mathbf{W}_u \in H(\mathrm{div}; \Omega)$ such that for every $v \in H_0^1(\Omega)$ and for every $\mathbf{Q} \in H(\mathrm{div}; \Omega)$

$$\begin{cases} (\mathbf{W}_u, \mathbf{Q})_\Omega - (u, \nabla \cdot (\alpha \mathbf{Q}))_\Omega + \langle u, \alpha \mathbf{Q} \cdot \nu \rangle_{\partial\Omega} = 0, \\ -(\mathbf{W}_u, \nabla v)_\Omega + \langle \mathbf{W}_u \cdot \nu, v \rangle_{\partial\Omega} - (u\boldsymbol{\beta}, \nabla v)_\Omega + \langle u\boldsymbol{\beta} \cdot \nu, v \rangle_{\partial\Omega} = (f, v)_\Omega, \\ u|_{\partial\Omega} = u_0, \end{cases} \tag{3.10}$$

where

$$(u,v)_\Omega \stackrel{\text{def}}{=} \int_\Omega u(x)v(x)\,\mathrm{d}x$$

is the standard L^2 product over Ω and

$$\langle u,v\rangle_{\partial\Omega} \stackrel{\text{def}}{=} \int_{\partial\Omega} u(x)v(x)\,\mathrm{d}\gamma$$

is the L^2 product over the space of dimension $n-1$ represented by the boundary of Ω, and ν is the outward-pointing normal vector.

Similarly, formulation (3.7) can be rewritten as: find $u \in H^1(\Omega)$ and $\mathbf{J}_u \in H(\mathrm{div};\,\Omega)$ such that for every $v \in H^1_0(\Omega)$ and for every $\mathbf{Q} \in H(\mathrm{div};\,\Omega)$

$$\begin{cases} (\mathbf{J}_u,\mathbf{Q})_\Omega - (u,\nabla\cdot(\alpha\mathbf{Q}))_\Omega + \langle u,\alpha\mathbf{Q}\cdot\nu\rangle_{\partial\Omega} - (u\boldsymbol{\beta},\mathbf{Q})_\Omega = 0, \\ -(\mathbf{J}_u,\nabla v)_\Omega + \langle \mathbf{J}_u\cdot\nu, v\rangle_{\partial\Omega} = (f,v)_\Omega\,, \\ u|_{\partial\Omega} = u_0|_{\partial\Omega}, \end{cases} \tag{3.11}$$

Given a non-negative integer k and a cell $T \in \mathcal{T}$, we denote with $\mathbb{V}^k(T)$ to be the Lagrange finite element space in T of degree at most k. Set

$$\mathbb{V}^k(\mathcal{T}) \stackrel{\text{def}}{=} \{v \in L^2(\Omega) : v|_T \in \mathbb{V}^k(T) \text{ for } T \in \mathcal{T}\}.$$

Similarly, for each face $F \in \mathcal{F}$, we define $\mathbb{M}^k(F)$ to be the Lagrange finite element space in F of degree at most k and

$$\mathbb{M}^k(\mathcal{F}) \stackrel{\text{def}}{=} \{\hat{v} \in L^2(\mathcal{F}) : \hat{v}|_F \in \mathbb{M}^k(F) \text{ for } F \in \mathcal{F}\}.$$

Given $g \in L^2(\partial\Omega)$, denote $\mathbb{M}^k_g(\mathcal{F})$ as the affine subspace of $\mathbb{M}^k(\mathcal{F})$ so that for each $F \in \mathcal{F}^\partial$, the function $\hat{v} \in \mathbb{M}^k_g(\mathcal{F})$ satisfies

$$\hat{v}|_F = \pi_F g,$$

where π_F is the orthogonal projection onto $\mathbb{M}^k(\mathcal{F})$.

The HDG discretization of (3.10) reads: find $(u^h,\mathbf{W}^h_u,\hat{u}^h) \in \mathbb{V}^k(\mathcal{T})\times[\mathbb{V}^k(\mathcal{T})]^n\times\mathbb{M}^k_{u_0}(\mathcal{F})$, such that for all $(v,\mathbf{Q},\mu) \in \mathbb{V}^k(\mathcal{T})\times[\mathbb{V}^k(\mathcal{T})]^n\times\mathbb{M}^k_0(\mathcal{F})$,

$$\begin{cases} \big(\mathbf{W}^h_u,\mathbf{Q}\big) - \big(u^h,\nabla\cdot(\alpha\mathbf{Q})\big) + \big\langle \hat{u}^h,(\alpha\mathbf{Q})\cdot\nu\big\rangle = 0, \\ -\big(\mathbf{W}^h_u,\nabla v\big) - \big(u^h\boldsymbol{\beta},\nabla v\big) + \Big\langle(\hat{\mathbf{W}}^h_u + u^h\boldsymbol{\beta})\cdot\nu, v\Big\rangle = (f,v)\,, \\ \Big\langle\hat{\mathbf{W}}^h_u + u^h\boldsymbol{\beta})\cdot\nu,\mu\Big\rangle = 0, \end{cases} \tag{3.12}$$

where the numerical flux $\hat{\mathbf{W}}^h_u$ on F is defined as:

$$(\hat{\mathbf{W}}^h_u + u^h\boldsymbol{\beta})\cdot\nu \stackrel{\text{def}}{=} \mathbf{W}^h_u\cdot\nu + u^h\boldsymbol{\beta}\cdot\nu + \tau(u^h-\hat{u}^h).$$

where τ is a stabilization parameter that can be a constant or a spatial function and that, usually, is chosen greater or equal than 0 everywhere. We will discuss about the role of the stabilization parameter in the next sections of this chapter.

We proceed similarly for system (3.11): find $(u^h, \mathbf{J}_u^h, \hat{u}^h) \in \mathbb{V}^k(\mathcal{T}) \times [\mathbb{V}^k(\mathcal{T})]^n \times \mathbb{M}_{u0}^k(\mathcal{F})$such that, for all $(v, \mathbf{Q}, \mu) \in \mathbb{V}^k(\mathcal{T}) \times [\mathbb{V}^k(\mathcal{T})]^n \times \mathbb{M}_0^k(\mathcal{F})$,

$$\begin{cases} \left(\mathbf{J}_u^h, \mathbf{Q}\right) - \left(u^h, \nabla \cdot (\alpha \mathbf{Q})\right) - \left(u^h \boldsymbol{\beta}, \mathbf{Q}\right) + \left\langle \hat{u}^h, \alpha \mathbf{Q} \cdot \nu \right\rangle = 0, \\ -\left(\mathbf{J}_u^h, \nabla v\right) + \left\langle \hat{\mathbf{J}}_u^h \cdot \nu, v \right\rangle = (f, v)\,, \\ \left\langle \hat{\mathbf{J}}_u^h \cdot \nu, \mu \right\rangle = 0, \end{cases} \tag{3.13}$$

the numerical flux $\hat{\mathbf{J}}_u^h \cdot \nu$ on F is defined as:

$$\hat{\mathbf{J}}_u^h \cdot \nu \stackrel{\text{def}}{=} \mathbf{J}_u^h \cdot \nu + \tau(u^h - \hat{u}^h). \tag{3.14}$$

Formulation (3.12) is the most widely used in literature; see, for example [14]. Here we focus mainly on formulation (3.13) because we are especially interested in the values of the auxiliary variable $\mathbf{J}_u$. On the other hand, the results that we expose in the next sections hold for both the formulations in view of the following proposition.

Proposition 1. *Let* $(u^h, \mathbf{W}_u^h, \hat{u}^h) \in \mathbb{V}^k(\mathcal{T}) \times [\mathbb{V}^k(\mathcal{T})]^n \times \mathbb{M}_{u0}^k(\mathcal{F})$ *be the solution of eq.* (3.12) *and let us assume that* $\boldsymbol{\beta} \in [\mathbb{V}^k(\mathcal{T})]^n$. *Define* $\mathbf{J}_u^h \stackrel{\text{def}}{=} \mathbf{W}_u^h + \boldsymbol{\beta} u^h$. *Then* $(u^h, \mathbf{J}_u^h, \hat{u}^h)$ *solves* (3.13).

Proof. Indeed, from $\boldsymbol{\beta} \in [\mathbb{V}^k(\mathcal{T})]^n$ we have that also $\mathbf{J}_u^h \in [\mathbb{V}^k(\mathcal{T})]^n$. It is now enough to substitute the definition of $\mathbf{J}_u^h$ inside the system (3.13) to obtain system (3.12). □

In the next section, we will assume that $\boldsymbol{\beta}$ is a constant and, therefore, the hypotheses of Proposition 1 are always satisfied and it will not be so important to specify what formulation we are using. When it is important to specify the degree k of the polynomial spaces, we refer to the previous methods as HDG_k methods, where k.

3.4 Relationship between FVM and HDG_0

The aim of this section is to show that the coefficient τ plays a fundamental role in stabilizing the method. In particular, we study a simple one–dimensional case using polynomials of degree 0 (i.e., $k = 0$).

Under these conditions, using a uniform grid and with an appropriate choice of τ, (3.12) and (3.13) are equivalent to the finite volume methods (3.9) and (3.8). In what follows, we simply set $\Omega = (0, 1)$ and denote

$$0 = x_0 < x_1 < \ldots < x_{N-1} < x_N = 1$$

are the grid points of $\mathcal{T}$. For $i = 0, \ldots, N-1$, we set $T_i = (x_i, x_{i+1})$ and $h_i = |T_i|$. We further assume that α, $\boldsymbol{\beta}$ and f are constants in Ω. We also use β instead of the bold font $\boldsymbol{\beta}$ to indicate that the drift is a scalar.

Under these assumptions, system (3.12) becomes

$$\begin{cases} h_i W_i + \alpha(\hat{u}_{i+1} - \hat{u}_i) = 0 & \forall i \in \{0, \dots, N-1\}, \\ \tau(2u_i - \hat{u}_{i+1} - \hat{u}_i) = h_i f & \forall i \in \{0, \dots, N-1\}, \\ W_{i-1} - W_i + \beta(u_{i-1} - u_i) + \tau(u_i + u_{i-1} - 2\hat{u}_i) = 0 & \forall i \in \{1, \dots, N-1\}, \\ \hat{u}_0 = u_0(0), \\ \hat{u}_N = u_0(1), \end{cases} \tag{3.15}$$

where u_i and W_i are the restrictions of u^h and $\mathbf{W}_u^h$ on T_i, respectively. Notice, however, that we will never explicitly assemble this system: one of the advantages of the HDG schemes (3.12) and (3.13) is the so called *static condensation*, i.e., the possibility to eliminate the unknowns defined on the cell from the discrete systems. To do this, in terms of (3.15), we write

$$W_i = \alpha \frac{\hat{u}_i - \hat{u}_{i+1}}{h_i}, \qquad u_i = \frac{\hat{u}_{i+1} + \hat{u}_i}{2} + \frac{h_i f}{2\tau}. \tag{3.16}$$

Replacing W_i and u_i into the third equation of (3.15) with the above identities, we get

$$\left(\frac{\tau+\beta}{2} + \frac{\alpha}{h_{i-1}}\right)\hat{u}_{i-1} - \left(\tau + \frac{\alpha}{h_i} + \frac{\alpha}{h_{i-1}}\right)\hat{u}_i + \left(\frac{\tau-\beta}{2} + \frac{\alpha}{h_i}\right)\hat{u}_{i+1} = \frac{\beta f}{2\tau}(h_i - h_{i-1}) - \frac{h_i + h_{i-1}}{2} f. \tag{3.17}$$

The same approach can be applied to eq. (3.13), obtaining the same result thanks to Proposition 1, .

3.4.1 HDG$_0$ and standard finite volume method

We now clarify the role of the stabilization parameter τ, used to define the numerical flux between two cells of the problem. We point out a similarity between the HDG method and finite volume schemes, where the definition of the flux is used to introduce a stabilization in the system, exploiting knowledge of the exact solution in special cases. In this prospective, our main result is Proposition 3, showing that under certain assumptions, the HDG discrete solutions defined on the trace are the same as those obtained from the Scharfetter–Gummel scheme (3.8). To this end, we first introduce the *dual triangulation*:

Definition 1. *Let $\mathcal{T}$ be a uniform triangulation of Ω with N cells and let $\{x_i\}_{i=0}^N$ be the set of all its faces, so that $x_0 \stackrel{\text{def}}{=} 0$, $x_N \stackrel{\text{def}}{=} 1$ and for every $i \in \{1, \dots, N\}$,*

$$x_i - x_{i-1} = 1/N.$$

Let $\{y_i\}_{i=0}^{N+1}$ be a collections of points such that

$$y_0 \stackrel{\text{def}}{=} x_0, \quad y_{N+1} \stackrel{\text{def}}{=} x_N, \quad \text{and} \quad y_i \stackrel{\text{def}}{=} \frac{x_i + x_{i-1}}{2} \text{ for } i \in \{1, \dots, N\}.$$

We call dual triangulation *$\mathcal{T}^d$ of $\mathcal{T}$ the collection of cells $S_i = (y_i, y_{i+1})$ for $i \in \{0, \dots, N\}$.*

Clearly, $\{x_i\}_{i=1}^N$ are the centers of the cells S_i and we will use their coordinates to apply the finite volume method. In order to simplify our argument, we say that x_0 and x_N are also the centers of S_0 and S_N, respectively.

Proposition 2. *Let us assume that the triangulation $\mathcal{T}$ is uniform with the mesh size h. Let also $\hat{u}_{i;\tau} \stackrel{\text{def}}{=} \{\hat{u}^h(x_i)\}_{i=0}^{N+1}$, where $\hat{u}^h$ is the HDG$_0$ discrete solution defined on the trace. We also denote v^h the discrete solution of the finite volume scheme using the standard flux* (3.9) *under the dual triangulation $\mathcal{T}^d$ and denote $v_i = v^h(x_i)$. Then we have that for every $i \in \{0, \dots, N\}$,*

$$\lim_{\tau \to 0} \hat{u}_{i;\tau} = v_i$$

Proof. Equations (3.17) that define the values of $\hat{u}_{i;\tau}$, on a uniform grid, become

$$\left(\frac{\tau+\beta}{2} + \frac{\alpha}{h}\right)\hat{u}_{i-1} - \left(\tau + \frac{2\alpha}{h}\right)\hat{u}_i + \left(\frac{\tau-\beta}{2} + \frac{\alpha}{h}\right)\hat{u}_{i+1} = -hf. \tag{3.18}$$

We denote with $\mathbf{M}_\tau$ the matrix of the linear system of equations (3.18).

The flux that we defined in (3.9) between the faces of $\mathcal{T}^d$ becomes

$$\mathcal{F}_i \stackrel{\text{def}}{=} -\alpha\frac{v_{i+1}-v_i}{h} + \beta\left(\frac{1}{2}v_{i+1} + \frac{1}{2}v_i\right),$$

where we have taken into account the fact that the distance between the centers of two dual cells S_i and S_{i+1} is h. If we sum the flux of each cell on both the sides, we get a system of equations like the following:

$$-\left(\frac{\alpha}{h} + \frac{\beta}{2}\right)v_{i-1} + \frac{2\alpha}{h}v_i - \left(\frac{\alpha}{h_i} - \frac{\beta}{2}\right)v_{i+1} = hf. \tag{3.19}$$

As we did before, we call $\mathbf{M}_0$ the matrix of the previous system. In $\mathbb{R}^{n\times n}$, we have that

$$\lim_{\tau \to 0} \mathbf{M}_\tau = -\mathbf{M}_0.$$

Because of the fact that the finite volume method is well posed, we have that there exists a neighborhood I of 0 such that $\mathbf{M}_\tau$ is invertible for each $\tau \in I$. From the continuity of the inverse, we get that the solution of the system of the HDG$_0$ method must converge to the one of the finite volume method and, hence, the proof is concluded. □

Is is worth to note that it is not possible to apply the HDG method with $\tau = 0$. Indeed, when we decrease the value of τ, the solution on the trace converges to the solution of the FVM, but the reconstruction on the cells become more and more unstable. When $\tau = 0$, the local system defined on each cell is not invertible anymore.

3.4.2 HDG$_0$ and the Scharfetter–Gummel FVM scheme

The Scharfetter–Gummel scheme offers a way to stabilize the previous finite volume method by approximating the flux using exponential functions. Let v^h be the finite volume approximation on $\mathcal{T}^d$ using the Scharfetter–Gummel flux (3.8) and denote $v_i = v^h(x_i)$. Recall from (3.8) that

$$\widetilde{\mathcal{F}}_i \stackrel{\text{def}}{=} \frac{\alpha}{s_i}\left(B\left(-\frac{\beta s_i}{\alpha}\right)v_i - B\left(\frac{\beta s_i}{\alpha}\right)v_{i+1}\right), \tag{3.20}$$

where $B(x)$ is the Bernoulli function defined in (1.12) and $s_i \stackrel{\text{def}}{=} x_{i+i} - x_i$. Applying this numerical flux, we obtain the following system:

$$-\frac{\alpha}{s_{i-1}}B\left(-\frac{\beta s_{i-1}}{\alpha}\right)v_{i-1} + \left[\frac{\alpha}{s_{i-1}}B\left(\frac{\beta s_{i-1}}{\alpha}\right) + \frac{\alpha}{s_i}B\left(-\frac{\beta s_i}{\alpha}\right)\right]v_i - \frac{\alpha}{s_i}B\left(\frac{\beta s_i}{\alpha}\right)v_{i+1} = \frac{s_{i-1}+s_i}{2}f. \tag{3.21}$$

In this section we prove that there is a relationship between the HDG$_0$ methods (with an appropriate choice of τ) and the finite volume method when the Scharfetter–Gummel scheme is applied; this relationship can be described with the next definition.

Definition 2. *Let $\mathcal{T}$ be uniform with mesh size h. Let $\hat{u}_i \stackrel{\text{def}}{=} \{\hat{u}^h(x_i)\}_{i=0}^N$ denote an HDG discrete solution defined on the trace, and let v^h be a finite volume approximation on $\mathcal{T}^d$ and denote $v_i = v^h(x_i)$. We say that the HDG method is* dual equivalent *to the finite volume method if for every $i \in \{0, \dots, N\}$,*

$$\hat{u}_i = v_i.$$

Note that the above definition is similar to Proposition 2 but here we choose a positive τ so that values of $\hat{u}_i$ and v coincide.

The following proposition shows the dual equivalence between HDG$_0$ and (3.20) for a particular stabilization parameter τ.

Proposition 3. *Let $\mathcal{T}$ be uniform. HDG$_0$ with the stabilization parameter*

$$\tau_0 \stackrel{\text{def}}{=} \frac{\alpha}{h}\left[2B\left(\frac{\beta h}{\alpha}\right) + \frac{\beta h}{\alpha} - 2\right] \tag{3.22}$$

is dual equivalent to (3.21)*.*

Proof. Let us consider eq. (3.17) and, in particular, the coefficient of $\hat{u}_{i-1}$ that, when $h_j = h$ for every j, becomes

$$\frac{\tau+\beta}{2} + \frac{\alpha}{h}.$$

If we replace τ with τ_0, we obtain

$$\frac{\alpha}{h}\left(B\left(\frac{\beta h}{\alpha}\right)+\frac{\beta h}{\alpha}\right),$$

which can be easily manipulated to obtain the opposite of the coefficient of v_{i-1} in eq. (3.21). In the very same way, the coefficient of $\hat{u}_i$

$$-\left(\tau+\frac{2\alpha}{h}\right)$$

becomes

$$-\frac{\alpha}{h}\left[2B\left(\frac{\beta h}{\alpha}\right)+\frac{\beta h}{\alpha}\right]. \tag{3.23}$$

The coefficient of v_i in the eq. (3.21) instead is

$$\frac{\alpha}{h}\left[B\left(\frac{-\beta h}{\alpha}\right)+B\left(\frac{\beta h}{\alpha}\right)\right],$$

which is again the opposite of the coefficient (3.23).

The same argument holds for the coefficient of $\hat{u}_{i+1}$

$$\left(\frac{\tau-\beta}{2}+\frac{\alpha}{h}\right)$$

that, after substituting τ, becomes the opposite of the coefficient of v_{i+1} inside eq. (3.21).

Finally, the right hand side of both the equations does not depend on τ when the elements have the same length. Therefore, it is easy to check that they are opposite of each other. □

Remark 4. *τ_0 is an non–negative even function with respect to β.*

Proof. The fact that τ_0 is an even function is a trivial consequence of the following identity:

$$B(x)-B(-x)=-x.$$

For the non–negativity, the result follows by computing the derivative of the function respect to β. □

We can rewrite rewrite the value of τ_0 as

$$\tau_0=\frac{\alpha}{h}\left[2B(\mathcal{P})+\mathcal{P}-2\right]$$

where $\mathcal{P}$ is the *mesh Péclet number* differing from a factor 2 with respect to more common definitions (see, e.g., [42]):

$$\mathcal{P}\stackrel{\text{def}}{=}\frac{\beta h}{\alpha}.$$

The results that we just exposed, together with the exactness of the Scharfetter–Gummel scheme for constant α and β imply that τ_0 is the best possible parameter to minimize the error on the trace of the HDG_0 scheme. Indeed, τ_0 is the unique value for which the numerical result of the method coincides with the exact solution of the problem in the specific case of α and β constant.

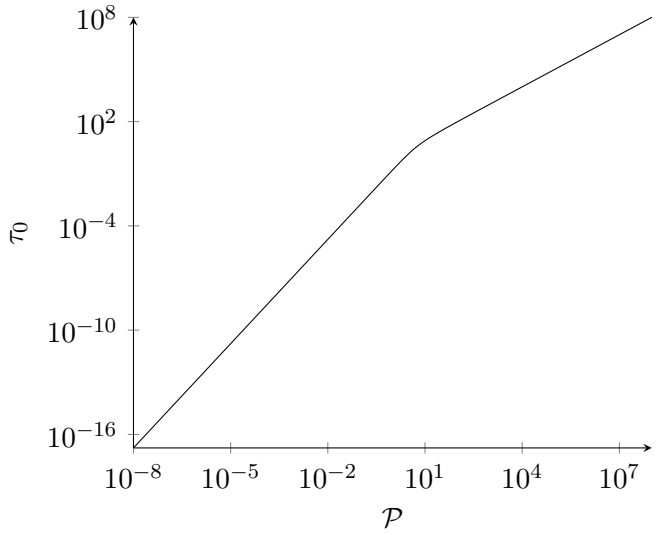


Figure 3.1: The values of τ_0 respect to $\mathcal{P}$ when $\frac{\alpha}{h} = 1$.

3.4.3 Numerical considerations for HDG$_0$

In this section we illustrate some experiments we performed related to the error of the HDG$_0$ method applied to the equation

$$\frac{\partial}{\partial x}\left(\beta u - \frac{\partial}{\partial x} u\right) = 0$$

in the domain $\Omega \stackrel{\text{def}}{=} [0, 1]$ with Dirichlet boundary conditions

$$\hat{u}(0) = 0 \quad \text{and} \quad \hat{u}(1) = 1.$$

In Figure 3.2 we applied the HDG$_0$ method for different values of β and for different mesh sizes h on a uniform triangulation $\mathcal{T}_h$. We define the error functions

$$e_u \stackrel{\text{def}}{=} \left| u - u^h \right|, \qquad e_{\mathbf{W}_u} \stackrel{\text{def}}{=} \left| \mathbf{W}_u - \mathbf{W}_u^h \right|, \qquad e_{\hat{u}} \stackrel{\text{def}}{=} \left| \hat{u} - \hat{u}^h \right|.$$

Because of Proposition 3, we know that, when $\tau = \tau_0$, $e_{\hat{u}}$ is identically zero on each point of the trace. This can be seen also numerically, where we identify a lower peak in the L^∞ error of the trace. It is interesting to note that the peak corresponds to a value of τ that decrease for smaller values of β or h (and, therefore, for smaller values of $\mathcal{P}$), in accordance with the definition of τ_0 given in (3.22).

As far as the cell error e_u is concerned, the plots in Figure 3.2 seem to suggest to take the smallest possible value for τ. However, this conflicts with the fact that the condensed linear system becomes ill conditioned as soon as τ becomes close to zero (where the matrix is singular). In this prospective, τ_0 is a value where the error $\|e_u\|_{L^2\mathcal{T}}$ is still reasonably small but the system is well conditioned.

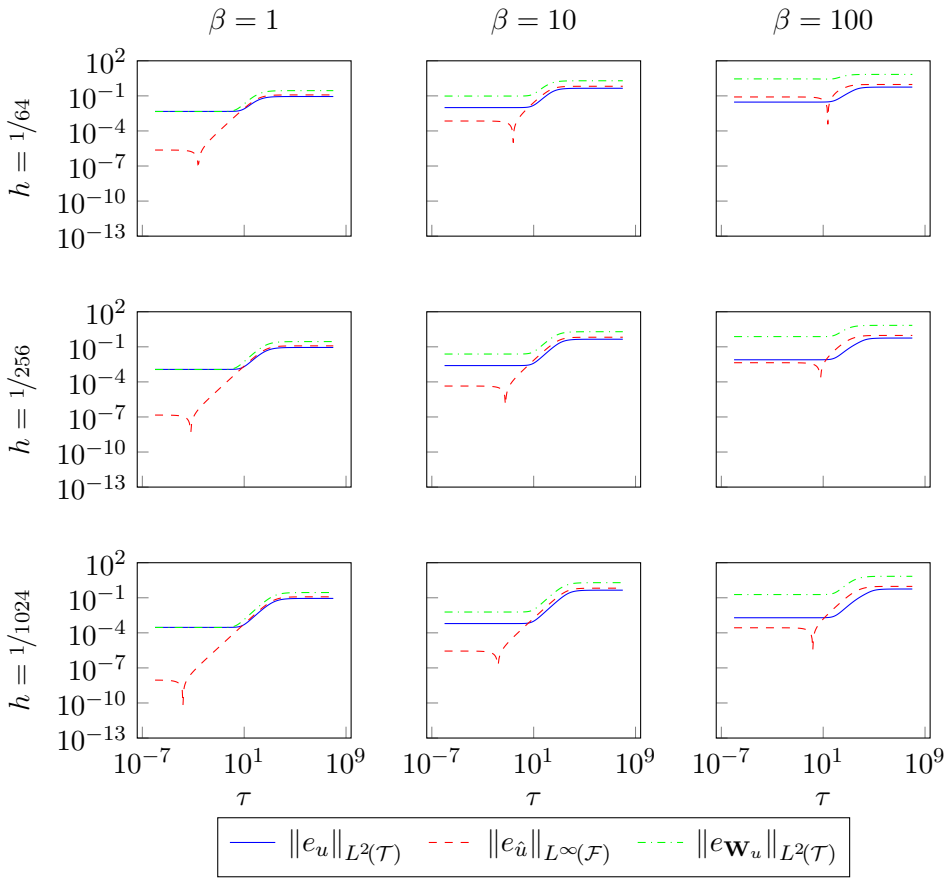


Figure 3.2: Errors obtained solving $\nabla \cdot (\boldsymbol{\beta} u - \nabla u) = 0$ on domain $\Omega = (0, 1)$ with $\hat{u}(0) = 0$ and $\hat{u}(1) = 1$ with HDG_0.

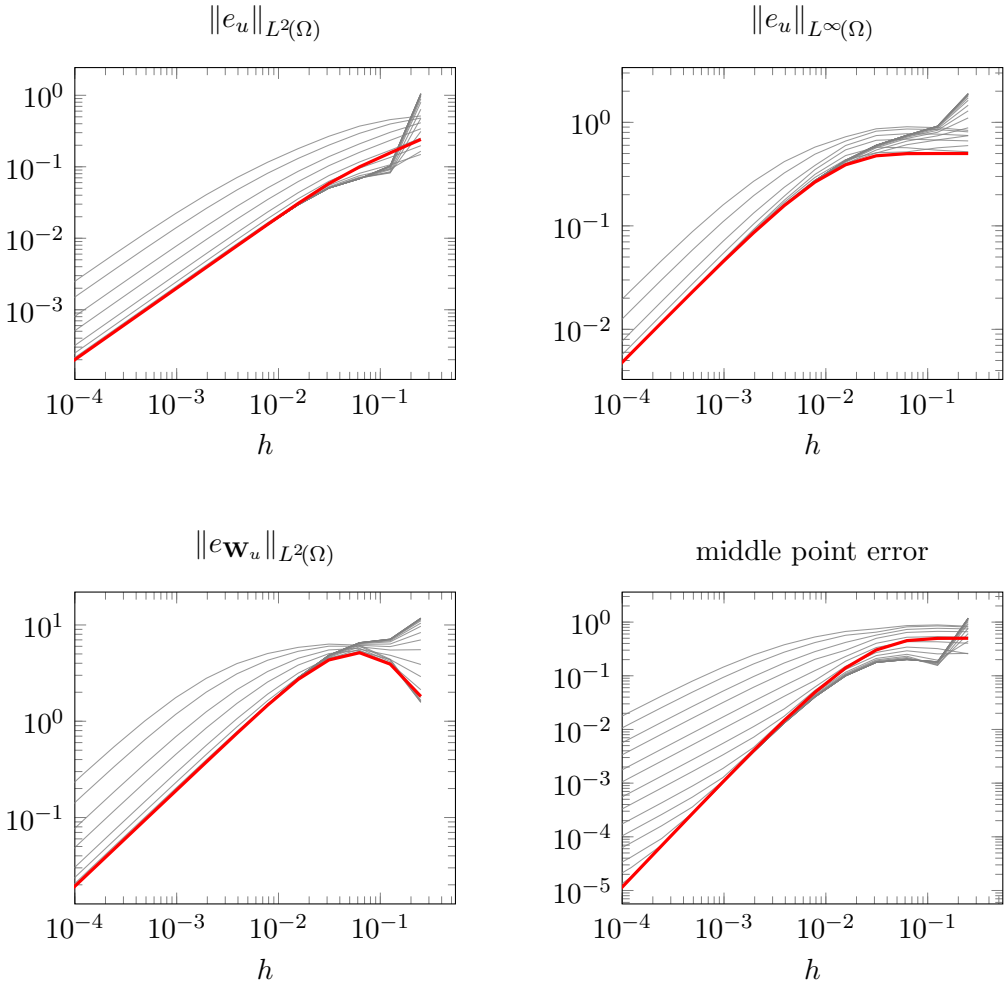


Figure 3.3: The error of the HDG$_0$ method as a function of the cell size h for a fixed value of the parameters α and β ($\alpha = 1$, $\beta = 100$) and for $f = 0$. The red line is the result using $\tau = \tau_0$.

Finally, in Figure 3.3 we compare the convergence of the HDG_0 method with $\tau = \tau_0$ with a fixed choice of τ. We report as gray lines the convergence plots for fixed choices of τ, going from 10^{-2} up to 10^3. In red, instead, we show the error of the HDG_0 method that uses $\tau = \tau_0$.

For the solution u, the plots show that we have the same order of convergence for both the L^2 and the L^∞ norms of the error. Also $\mathbf{W}_u$ preserves the same order of convergence in L^2.

Let us define for every cell $T \in \mathcal{T}$ the point x_T as the center of the cell. We define the middle point error of u^h as

$$\max_{T\in\mathcal{T}} \left| u^h(x_T) - u(x_T) \right|$$

The last plot of Figure 3.3 shows the middle point error of the HDG_0 method as a function of h. In this case, we see that using $\tau = \tau_0$ the error decreases faster than with any other fixed choice of τ, presenting a different order of convergence.

3.5 Relationship between FVM and HDG_k

3.5.1 HDG_k and the Scharfetter–Gummel FVM scheme

At the end of section 3.4.2, we have rewritten the value of τ_0 as a function of the mesh Péclet number $\mathcal{P}$. On a uniform grid, this result can be generalized to polynomials of higher order. In particular, we can show that there exists a family of constants τ_k such that, according to the previous section, the HDG_k scheme is dual equivalent to the finite volume scheme. Due to Proposition 1, we will focus on the formulation (3.13) and show the following main result.

Theorem 1. *For every degree $k > 0$, there exists a unique value τ_k such that the HDG_k method with the stabilization parameter $\tau = \tau_k$ is dual equivalent to the Scharfetter–Gummel scheme.*

The idea of the proof is to investigate the linear system for $\hat{u}^h$ which can be obtained by the static condensation. To this end, for each cell $T^i := (x_{i-1}, x_i) \in \mathcal{T}$, denote $\{\phi^i_j\}_{j=0}^k$ the set of shape functions in $\mathbb{V}^k(T^i)$. For the approximation u^h in T^i, we set $u^h \overset{\text{def}}{=} \sum_{j=0}^k u^i_j \phi^i_j$ with the coefficient vector $\mathbf{u} := (u^i_0, \dots, u^i_k)^\top$. Similarly, we set the approximation of the current $\mathbf{J}^h_u := \sum_{j=0}^k J^i_j \phi^i_j$ for some coefficient vector $\mathbf{J} := (J^i_0, \dots, J^i_k)^\top$.

We also set $\delta \overset{\text{def}}{=} \frac{\tau h}{\alpha}$ and we let $\Lambda = \{c_{i,j}\}_{i,j=0}^N$ denote the system matrix for $\hat{u}^h$ and $\mathbf{r}$ the right hand side vector. The static condensation indicates that on each cell T, $\mathbf{u}$ and $\mathbf{J}$ are functions of the boundary values $\hat{u}^h$. According to transmission condition (the third equation in (3.13)), we obtain that for $|i-j| > 1$, $c_{i,j} = 0$. The following lemma shows that for $|i-j| \leq 1$, $c_{i,j}$ is a function of δ and the mesh Péclet number $\mathcal{P}$.

Lemma 5. *The matrix Λ is a tridiagonal Toeplitz matrix*

$$\Lambda = \frac{\alpha}{h}\begin{pmatrix} c_2 & c_3 & & & \\ c_1 & c_2 & c_3 & & \\ & c_1 & \ddots & \ddots & \\ & & \ddots & \ddots & c_3 \\ & & & c_1 & c_2 \end{pmatrix} \tag{3.24}$$

whose coefficients c_1, c_2 and c_3 depend only on δ and $\mathcal{P}$. Moreover, there exists a coefficient $r = r(\delta, \mathcal{P})$ such that the right hand side vector $\mathbf{r}$ is a constant vector whose entries are all equal to

$$(\mathbf{r})_i = hfr.$$

Proof. Based on (3.13), for $m = 1, \ldots, N$, we write the local discrete system on $T := T^m \in \mathcal{T}$ by $\mathbf{A}_T\mathbf{X}_T = \mathbf{b}_T$. Here

$$\mathbf{X}_T = \begin{pmatrix} \mathbf{J} \\ \mathbf{u} \end{pmatrix}, \quad \mathbf{b}_T = \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{pmatrix} \text{ and } \quad \mathbf{A}_T = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix} \tag{3.25}$$

where

- $\mathbf{A}_{11} = \{\left(\phi_j^m, \phi_i^m\right)\}_{i,j=0}^k$. So the elements of this matrix scale linearly with h.
- $\mathbf{A}_{12} = \{-\left(\phi_j^m, \nabla\cdot(\alpha\phi_i^m)\right) - \left(\phi_j^m\beta, \phi_i^m\right)\}_{i,j=0}^k$. So the terms that appear inside this matrix are of the form $-k_1\alpha - k_2\beta h$, with $k_1, k_2 \in \mathbb{R}$ not depending on any parameter of the problem (τ, α, β or h).
- $\mathbf{A}_{21} = \{-\left(\phi_j^m, \nabla\phi_i^m\right) + \left\langle \phi_j^m\cdot\nu, \phi_i^m\right\rangle\}_{i,j=0}^k$. These elements do not depend on α, β or h.
- $\mathbf{A}_{22} = \{\left\langle \tau\phi_j^m, \phi_i^m\right\rangle\}_{i,j=0}^k$. In fact, $(\mathbf{A}_{22})_{00} = (\mathbf{A}_{22})_{kk} = \tau$ and all the other entries are zero.
- $\mathbf{b}_1 := \mathbf{b}_1^m = \{-\left\langle \hat{u}^h, \alpha\phi_i^m\cdot\nu\right\rangle\}_{i=0}^k$.
- $\mathbf{b}_2 := \mathbf{b}_2^m = \{\left\langle \tau\hat{u}^h, \phi_i^m\right\rangle + (f, \phi_i^m)\}_{i=0}^k$.

Now we proceed with the following change of variables

$$\mathbf{v} \overset{\text{def}}{=} \alpha\mathbf{u}, \qquad \hat{v}^h \overset{\text{def}}{=} \alpha\hat{u}^h, \qquad \mathbf{q} \overset{\text{def}}{=} h\mathbf{J}, \tag{3.26}$$

and set $\widetilde{\mathbf{X}}_T = (\mathbf{q}, \mathbf{v})^\top$ so that (3.25) becomes

$$\begin{pmatrix} \mathbf{A}_{11}/h & \mathbf{A}_{12}/\alpha \\ \mathbf{A}_{21} & h\mathbf{A}_{22}/\alpha \end{pmatrix}\widetilde{\mathbf{X}}_T = \begin{pmatrix} \mathbf{b}_1 \\ h\mathbf{b}_2 \end{pmatrix} \tag{3.27}$$

We denote the left hand side matrix above to be $\widetilde{\mathbf{A}}$ with block $\widetilde{\mathbf{A}}_{ts} \in \mathbb{R}^{(k+1)\times(k+1)}$ for $t, s = 1, 2$. So the matrix $\widetilde{\mathbf{A}}_{11}$ loses its dependency on h. The elements of $\widetilde{\mathbf{A}}_{12}$, instead, become linear functions respect of $\mathcal{P}$. The matrix $\widetilde{\mathbf{A}}_{21} = \mathbf{A}_{21}$ while the matrix $\widetilde{\mathbf{A}}_{22}$ is a matrix function of δ.

Now we want to apply the static condensation by combining the rescaled system (3.27) and the third equation from (3.13), i.e. the transmission condition. For the unknown trace value $\hat{u}_i$ defined between the two adjacent cells T^i and T^{i+1}, we write

$$(\tau u_k^i + J_k^i) + (\tau u_0^{i+1} - J_0^{i+1}) - 2\tau\hat{u}_i = 0.$$

Using the change of variables in (3.26), we obtain that

$$(\delta v_k^i + q_k^i) + (\delta v_0^{i+1} - q_0^{i+1}) - 2\delta\hat{v}_i = 0. \tag{3.28}$$

Let

$$\mathbf{p}_k \overset{\text{def}}{=} (\mathbf{e}_k, \delta\mathbf{e}_k)^\top, \qquad \text{and} \qquad \mathbf{p}_0 \overset{\text{def}}{=} (-\mathbf{e}_0, \delta\mathbf{e}_0)^\top, \tag{3.29}$$

with $\mathbf{e}_j$ for $j = 0, \ldots, k$ denoting the canonical vector in $\mathbb{R}^{k+1}$. Based on the rescaled system (3.27), we rewrite the terms in parentheses in (3.28) as the matrix form, i.e.

$$\delta v_k^i + q_k^i = \mathbf{p}_k^\top \widetilde{\mathbf{X}}_T = \mathbf{p}_k^\top \widetilde{\mathbf{A}}^{-1} \begin{pmatrix} \mathbf{b}_1^i \\ h\mathbf{b}_2^i \end{pmatrix} \tag{3.30}$$

and

$$\delta v_0^{i+1} - q_0^{i+1} = \mathbf{p}_0^\top \widetilde{\mathbf{X}}_T = \mathbf{p}_0^\top \widetilde{\mathbf{A}}^{-1} \begin{pmatrix} \mathbf{b}_1^{i+1} \\ h\mathbf{b}_2^{i+1} \end{pmatrix}. \tag{3.31}$$

Next we want to investigate the dependency of δ and $\mathcal{P}$ for the vector $\mathbf{b_T}$. Define the vector $\boldsymbol{\varphi} \in \mathbb{R}^{k+1}$ such that for $j = 0, \ldots, k$,

$$\boldsymbol{\varphi}_j \overset{\text{def}}{=} \frac{1}{h} \int_{T^i} \phi_j^i \, \mathrm{d}x.$$

According to the definition of $\mathbf{b}_T$ in (3.25), we can derive that

$$\mathbf{b}_1^i = \hat{v}_{i-1}\mathbf{e}_0 - \hat{v}_i\mathbf{e}_k, \qquad \mathbf{b}_1^{i+1} = \hat{v}_i\mathbf{e}_0 - \hat{v}_{i+1}\mathbf{e}_k, \tag{3.32}$$

$$\mathbf{b}_2^i = hf\boldsymbol{\varphi} + \frac{\delta}{h}\hat{v}_{i-1}\mathbf{e}_0 + \frac{\delta}{h}\hat{v}_i\mathbf{e}_k, \qquad \mathbf{b}_2^{i+1} = hf\boldsymbol{\varphi} + \frac{\delta}{h}\hat{v}_i\mathbf{e}_0 + \frac{\delta}{h}\hat{v}_{i+1}\mathbf{e}_k. \tag{3.33}$$

We combine the two equations above by setting

$$\mathbf{b}_0 \overset{\text{def}}{=} \begin{pmatrix} \mathbf{e}_0 \\ \delta\mathbf{e}_0 \end{pmatrix} \quad \text{and} \quad \mathbf{b}_k \overset{\text{def}}{=} \begin{pmatrix} -\mathbf{e}_k \\ \delta\mathbf{e}_k \end{pmatrix} \tag{3.34}$$

so that

$$\begin{pmatrix} \mathbf{b}_1^i \\ h\mathbf{b}_2^i \end{pmatrix} = h^2 f \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\varphi} \end{pmatrix} + \hat{v}_{i-1}\mathbf{b}_0 + \hat{v}_i\mathbf{b}_k, \tag{3.35}$$

$$\begin{pmatrix} \mathbf{b}_1^{i+1} \\ h\mathbf{b}_2^{i+1} \end{pmatrix} = h^2 f \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\varphi} \end{pmatrix} + \hat{v}_i\mathbf{b}_0 + \hat{v}_{i+1}\mathbf{b}_k. \tag{3.36}$$

Inserting (3.35) and (3.36) into (3.31) and (3.30) to write

$$\delta v_k^i + q_k^i = h^2 f \mathbf{p}_k^\top \widetilde{\mathbf{A}}^{-1} \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\varphi} \end{pmatrix} + \hat{v}_{i-1} \mathbf{p}_k^\top \widetilde{\mathbf{A}}^{-1} \mathbf{b}_0 + \hat{v}_i \mathbf{p}_k^\top \widetilde{\mathbf{A}}^{-1} \mathbf{b}_k. \tag{3.37}$$

and

$$\delta v_0^{i+1} - q_0^{i+1} = h^2 f \mathbf{p}_0^\top \widetilde{\mathbf{A}}^{-1} \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\varphi} \end{pmatrix} + \hat{v}_i \mathbf{p}_0^\top \widetilde{\mathbf{A}}^{-1} \mathbf{b}_0 + \hat{v}_{i+1} \mathbf{p}_0^\top \widetilde{\mathbf{A}}^{-1} \mathbf{b}_k \tag{3.38}$$

Finally, we apply the above two equations in (3.28) and combing coefficients with respect to $\{\hat{v}_i\}$. Define these coefficients by

$$r = r(\delta, \mathcal{P}) \overset{\text{def}}{=} -(\mathbf{p}_0 + \mathbf{p}_k)^\top \widetilde{\mathbf{A}}^{-1} \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\varphi} \end{pmatrix} \tag{3.39}$$

$$\begin{aligned} c_1 &= c_1(\delta, \mathcal{P}) \overset{\text{def}}{=} \mathbf{p}_k^\top \widetilde{\mathbf{A}}^{-1} \mathbf{b}_0, \\ c_2 &= c_2(\delta, \mathcal{P}) \overset{\text{def}}{=} \mathbf{p}_0^\top \widetilde{\mathbf{A}}^{-1} \mathbf{b}_0 + \mathbf{p}_k^\top \widetilde{\mathbf{A}}^{-1} \mathbf{b}_k - 2\delta, \\ c_3 &= c_3(\delta, \mathcal{P}) \overset{\text{def}}{=} \mathbf{p}_0^\top \widetilde{\mathbf{A}}^{-1} \mathbf{b}_k, \end{aligned} \tag{3.40}$$

and write

$$c_1 \hat{v}_{i-1} + c_2 \hat{v}_i + c_3 \hat{v}_{i+1} = h^2 f r, \tag{3.41}$$

The previous equation can be rewritten respect to $\hat{u}_i$:

$$\alpha \left(c_1 \hat{u}_{i-1} + c_2 \hat{u}_i + c_3 \hat{u}_{i+1} \right) = h^2 f r. \tag{3.42}$$

Noting that the previous equation has been manipulated by multiplying the original equation with a factor h, we thus obtain the system matrix (3.24) whose entries c_1, c_2 and c_3 functions of δ and $\mathcal{P}$. The proof is complete. □

We know from equation (3.18) that, for $k = 0$,

$$r = -1.$$

The following lemma shows that this result holds true also for higher degrees.

Lemma 6. *There holds that $r = -1$ in (3.39) for all polynomial degree k.*

Proof. Note that the right hand side of (3.39) is an algebraic form that does not depend on the data f. Though r is obtained from the discrete transmission condition (3.13) between two adjacent cells (i.e. static condensation), we can actually reproduce such algebraic form in a simpler mesh setting.

Consider the model problem (3) on $\Omega = (0, h)$ associated with the homogeneous Dirichlet boundary condition. We shall approximate the solution using the HDG scheme (3.13) with the mesh $\mathcal{T}$ that contains only one cell, namely $\mathcal{T} = \{\Omega\}$. Letting the data $f = 1/h^2$, we can approximate $\mathbf{J}_u^h$ and u^h by directly solving the local problem $\mathbf{A}_T \mathbf{X}_T = \mathbf{b}_T$ introduced by (3.25). In this proof we shall use the notations from the proof

in Lemma 5 but we remove the superscripts for the cell indices. Recalling that $\mathbf{J}$ and $\mathbf{u}$ are the corresponding finite element coefficient vectors with dimension $k+1$, we follow from the change of variables in (3.26) as well as the rescaled local system (3.27) to get

$$\begin{pmatrix} h\mathbf{J} \\ \alpha\mathbf{u} \end{pmatrix} = \begin{pmatrix} \mathbf{q} \\ \mathbf{v} \end{pmatrix} = \widetilde{\mathbf{A}}^{-1} \begin{pmatrix} \mathbf{b}_1 \\ h\mathbf{b}_2 \end{pmatrix} = \widetilde{\mathbf{A}}^{-1} \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\varphi} \end{pmatrix}, \tag{3.43}$$

where for the last equality we used the fact that $\mathbf{b}_1 = 0$ due to the zero boundary condition and $\mathbf{b}_2 = hf\boldsymbol{\varphi} = \frac{1}{h}\boldsymbol{\varphi}$. This leads to

$$\begin{aligned} r &= -\left(\mathbf{p}_0 + \mathbf{p}_k\right)^\top \widetilde{\mathbf{A}}^{-1} \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\varphi} \end{pmatrix} \\ &= -\left[\begin{pmatrix} -\mathbf{e}_0 \\ \delta\mathbf{e}_0 \end{pmatrix}^\top + \begin{pmatrix} \mathbf{e}_k \\ \delta\mathbf{e}_k \end{pmatrix}^\top \right] \begin{pmatrix} h\mathbf{J} \\ \alpha\mathbf{u} \end{pmatrix} = -h(\tau u_k + J_k + \tau u_0 - J_0). \end{aligned} \tag{3.44}$$

On the other hand, we choose the test function $v = 1$ in the second equation of (3.13) to get

$$\left\langle \mathbf{J}_u^h \cdot \nu + \tau\left(u^h - \hat{u}^h\right), 1 \right\rangle_{\partial\Omega} = (f, 1)_\Omega .$$

We use coefficient vectors as well as $f = 1/h^2$ to rewrite the above equation as

$$-J_0 + \tau u_0 + J_k + \tau u_k = \frac{1}{h}.$$

Combing (3.44) with the above equation immediately implies that $r = -1$. □

The next lemma provides some properties of the coefficients c_i defined in Lemma 5 for $i = 1, 2, 3$.

Lemma 7. *Let c_1, c_2 and c_3 be as defined in Lemma 5. There hold*

1. $c_1 + c_2 + c_3 = 0$;
2. $c_3 - c_1 = -\mathcal{P}$;
3. $c_3(\delta, \mathcal{P}) = c_1(\delta, -\mathcal{P})$;
4. $c_2(\delta, \mathcal{P}) = c_2(\delta, -\mathcal{P})$;
5. $c_1 = -\frac{1}{2}\left(c_2 - \mathcal{P}\right)$ *and* $c_3 = -\frac{1}{2}\left(c_2 + \mathcal{P}\right)$.

Proof. The idea to show (1) and (2) is to choose an exact solution u in the finite element space $\mathbb{V}^k(\mathcal{T})$ so that $u^h = u$ by uniqueness of the numerical scheme (3.13). In particular, f is a constant if u is linear. This allows as to apply the previous lemma to discover the properties of c_i for $i = 1, 2, 3$.

It is worth noting that the space $\mathbb{V}^0(\mathcal{T})$ does not contain the linear functions and, therefore, the argument for (2) below will not hold for $k = 0$. In this case, we can explicitly compute the matrix $\widetilde{\mathbf{A}}$ and its inverse. So,

$$\widetilde{\mathbf{A}} = \begin{pmatrix} 1 & -\mathcal{P} \\ 0 & 2\delta \end{pmatrix} \qquad \text{and} \qquad \widetilde{\mathbf{A}}^{-1} = \frac{1}{2\delta}\begin{pmatrix} 2\delta & \mathcal{P} \\ 0 & 1 \end{pmatrix}.$$

According to (3.40), for $k = 0$, we write

$$c_1 = \frac{1}{2}\delta + \frac{1}{2}\mathcal{P} + 1, \qquad c_2 = -\delta - 2, \qquad \text{and} \qquad c_3 = \frac{1}{2}\delta - \frac{1}{2}\mathcal{P} + 1,$$

and, therefore, all the assertions hold for $k = 0$.

Let $k > 0$. To prove (1), we set $u \equiv 1$. The assertion follows by plugging in the the discrete solutions $u^h = 1$, $\mathbf{J}_u^h = \beta$, $\hat{u} = 1$ as well as the data $f = 0$ into (3.13).

For (2), we consider $\Omega = [-h, h]$ made of two cells of the same size, i.e. $\mathcal{T} = \{(-h, 0), (0, h)\}$. We set $u = \frac{x}{h}$ so that $u^h = u$ and $f \overset{\text{def}}{=} \frac{\beta}{h}$ is a constant. Applying Lemma 6 into (3.42) to get

$$c_1\hat{u}^h(-h) + c_2\hat{u}^h(0) + c_3\hat{u}^h(h) = -\frac{h\beta}{\alpha}. \tag{3.45}$$

The assertion then follows by inserting the values of $\hat{u}$ into the above equation.

To simplify our proof for (3), we set the domain to be $(0, 1)$ and, as we did in proof of Lemma 6, we suppose that the domain is made of only one cell $T = \Omega$. Thus, the function $\hat{u}^h$ is directly given by the Dirichlet boundary conditions. We note that, for every $Q \in \mathbb{V}^k(T)$, also the function $Q(1 - x)$ is a function in $\mathbb{V}^k(T)$. Moreover, if $Q_0, \dots, Q_k$ is a lagrangian basis for $\mathbb{V}^k(T)$, then also $Q_0(1 - x), \dots, Q_k(1 - x)$ is a lagrangian basis for the same space (with symmetrical nodes). Let us suppose that $\left(u^h, J^h, \hat{u}^h\right)$ is a solution to our discrete problem (3.13) for some parameters α and β with Dirichlet boundary conditions $\hat{u}^h(0) = k_0$ and $\hat{u}^h(h) = k_1$. Then $\left(u^h(1 - x), -J^h(1 - x), \hat{u}^h(1 - x)\right)$ is a solution of the problem with parameters α and $-\beta$ with Dirichlet boundary conditions $\hat{u}^h(0) = k_1$ and $\hat{u}^h(h) = k_0$. This can be proven by inserting this solution inside system (3.13) and replacing each test function with its symmetrical one.

Now let us set $\alpha = 1$, $f = 0$, $k_0 = 0$, and $k_1 = 1$. Since $h = 1$, we have in (3.38) for $i = 0$ that

$$c_3(\delta, \mathcal{P}) = \mathbf{p}_0^\top \widetilde{\mathbf{A}}^{-1}(\mathcal{P})\mathbf{b}_k = \delta u^h(0) - J^h(0). \tag{3.46}$$

We shall use an analogous strategy to show $c_1(\delta, -\mathcal{P})$ is equal the right hand side above. Indeed, let us consider the problem with opposite Dirichlet boundary conditions (i.e., $k_0 = 1$ and $k_1 = 0$) and with $\alpha = 1$ and opposite peclet number respect to the previous problem (in other words, we swap the sign of the drift coefficient). We know that a solution of this problem is $(u^h(1 - x), J^h(1 - x), \hat{u}^h(1 - x))$. Therefore, applying (3.37) to get

$$c_1(\delta, -\mathcal{P}) = \mathbf{p}_k^\top \widetilde{\mathbf{A}}^{-1}(-\mathcal{P})\mathbf{b}_0 = \delta u^h(1 - 1) + (-J^h(1 - 1)) = \delta u^h(0) - J^h(0).$$

The proof of point (5) is obtained by algebraical composition of points (1) and (2). Point (4) follows from (5) and (3). The proof is complete. □

Before exposing the last lemma that we need for the proof of the main result of this section, we introduce a property that we will use later while proving Lemma 9.

Proposition 8. *Let $M \in \mathbb{R}^{n\times n}$ be a square matrix made of three blocks as follows*

$$M = \begin{pmatrix} A & B \\ C & 0 \end{pmatrix}$$

with A, B and C square matrices. If C is singular, then M is singular.

Proof. If C is singular, there exists a vector $\mathbf{v}$ such that

$$\mathbf{v}^\top C = 0.$$

We define the vector $\mathbf{w} \in \mathbb{R}^n$ as

$$\mathbf{w} := \begin{pmatrix} \mathbf{0} \\ \mathbf{v} \end{pmatrix},$$

so that

$$\mathbf{w}^\top M = 0.$$

Therefore, M is singular. □

The next lemma shows that the coefficient c_i for $i = 1, 2, 3$ are all rational functions for δ and $\mathcal{P}$ satisfying that the degrees of numerators and denominators are no more than 1 with respect to δ. To this end, we denote $\mathbb{P}_\delta^k$ the set of polynomials with respect to δ with degree less equal to k.

Lemma 9. *The coefficients c_1, c_2 and c_3 are rational functions for δ and $\mathcal{P}$. Indeed, for each index $i \in \{1, 2, 3\}$, there exist two polynomials $p_i(\delta, \mathcal{P})$ and $d_i(\delta, \mathcal{P})$ in $\mathbb{P}_\delta^1$ such that*

$$c_i = \frac{p_i}{d_i}.$$

Proof. Here we only provide a proof for c_1 and the proof for c_2 and c_3 follows by Part (3) and (5) in Lemma 7. Using the structure of $\mathbf{p}_k$ and $\mathbf{b}_0$, according to (3.40), we write

$$\begin{aligned} c_1 &= \mathbf{p}_k^\top \widetilde{\mathbf{A}}^{-1}\mathbf{b}_0 \\ &= (\widetilde{\mathbf{A}}^{-1}\mathbf{b}_0)_k + \delta(\widetilde{\mathbf{A}}^{-1}\mathbf{b}_0)_{2k} = \frac{\det(\widetilde{\mathbf{A}}_k) + \delta\det(\widetilde{\mathbf{A}}_{2k})}{\det(\widetilde{\mathbf{A}})}. \end{aligned} \tag{3.47}$$

Here we used the Cramer's rule for the last equality above and denote the matrix $\widetilde{\mathbf{A}}_k$ generated from $\widetilde{\mathbf{A}}$ but replacing its k-th column with $\mathbf{b}_0$.

Let us first focus on $\det(\widetilde{\mathbf{A}})$. Clearly, since δ only appears in only two entries in $\widetilde{\mathbf{A}}_{22}$, we have $\det(\widetilde{\mathbf{A}}) \in \mathbb{P}_\delta^2$. Furthermore, if $\delta = 0$, i.e. the stabilization parameter $\tau = 0$, we have $\widetilde{\mathbf{A}}_{22} = 0$. We note that the matrix $\widetilde{\mathbf{A}}_{21}$, whose elements are

$$\{-\left(\phi_j^m, \nabla\phi_i^m\right) + \left\langle \phi_j^m \cdot \nu, \phi_i^m \right\rangle\}_{i,j=0}^k = \{\left(\nabla \cdot \phi_j^m, \phi_i^m\right)\},$$

is singular. Indeed, let $p(x)$ be a polynomial of degree smaller than k on the cell T and let $\mathbf{p}$ be the vector of its coefficients respect to the basis $\phi_0^m, \ldots, \phi_k^m$. Then $\widetilde{\mathbf{A}}_{21}\mathbf{p}$ is a vector whose i-th entry is $(\nabla \cdot p, \phi_i^m)$. From this we get that the vector $(1, 1, \ldots, 1)$ (which correspond to the polynomial $p = 1$) is in the kernel of the matrix. From Proposition 8 we get that the matrix $\widetilde{\mathbf{A}}$ is also singular, namely $\det(\widetilde{\mathbf{A}}) = 0$ if $\delta = 0$. This allows us to write $\det(\widetilde{\mathbf{A}}) = \delta d$ with $d \in \mathbb{P}^1_\delta$. For $\widetilde{\mathbf{A}}_k$, we note that there are three entries linearly depend on δ: $(\widetilde{\mathbf{A}}_k)_{k+1,k}$, $(\widetilde{\mathbf{A}}_k)_{k+1,k+1}$ and $(\widetilde{\mathbf{A}}_k)_{2k,2k}$. Since two of them are in the $k+1$-th row, we still obtain that $\det(\widetilde{\mathbf{A}}_k) \in \mathbb{P}^2_\delta$. When $\delta = 0$, we notice that the last column in the block matrix $(\widetilde{\mathbf{A}}_k)_{(k+1):2k,0:k}$ (i.e. $\widetilde{\mathbf{A}}_{21}$ in $\widetilde{\mathbf{A}}$) are all zero and hence it is singular. Whence, by Proposition 8, $\widetilde{\mathbf{A}}_k$ is singular when $\delta = 0$ and we can write its determinant as $\det(\widetilde{\mathbf{A}}_k) = \delta m_k$ for some $m_k \in \mathbb{P}^1_\delta$.

In terms of the polynomial degree for $\det(\widetilde{\mathbf{A}}_{2k})$ with respect to δ, we first note that $\widetilde{\mathbf{A}}_{2k}$ has only two entries containing δ: $(\widetilde{\mathbf{A}}_{2k})_{k+1,k+1}$ and $(\widetilde{\mathbf{A}}_{2k})_{k+1,2k}$. Since they are in the same row, we have $\det(\widetilde{\mathbf{A}}_{2k}) \in \mathbb{P}^1_\delta$. Following the argument for $\det(\widetilde{\mathbf{A}})$ we can conclude that $\det(\widetilde{\mathbf{A}}_{2k}) = \delta m_{2k}$ with m_{2k} not depending on δ.

Gathering the above results for the determinants in (3.47), we conclude that

$$c_1 = \frac{m_k + \delta m_{2k}}{d}$$

as desired. □

Now we are in a position to show the main results in Theorem 1.

Proof of Theorem 1. From Lemma 9 we write c_2 as

$$c_2 = \frac{l}{q},$$

for some $l(\delta, \mathcal{P}), q(\delta, \mathcal{P}) \in \mathbb{P}^1_\delta$ satisfying that l and q are coprime. By Part (4) of Lemma 7, we have that

$$\frac{l(\delta, \mathcal{P})}{q(\delta, \mathcal{P})} = \frac{l(\delta, -\mathcal{P})}{q(\delta, -\mathcal{P})}.$$

This implies that $q(\delta, \mathcal{P})$ divides $l(\delta, \mathcal{P})q(\delta, -\mathcal{P})$. Since l and q are coprime, we have that $q(\delta, \mathcal{P})$ divides $q(\delta, -\mathcal{P})$ and, by an analogous argument, $q(\delta, -\mathcal{P})$ divides $q(\delta, \mathcal{P})$. Therefore, there exist a constant $t \in \mathbb{R}$ such that

$$q(\delta, \mathcal{P}) = tq(\delta, -\mathcal{P}).$$

On the other hand, the absolute values of coefficients of $q(\delta, \mathcal{P})$ and $q(\delta, -\mathcal{P})$ are the same. So, $t = -1$ or 1, namely, q can be an even or odd polynomial (respect to $\mathcal{P}$). Here we claim that q is even with respect to $\mathcal{P}$. If q is odd, l should be odd in order of keeping the fact that c_2 is even. However, q and l contradicts the coprimality assumption due to the extra factor $\mathcal{P}$.

Let $\mathbb{P}_{\mathcal{P}}$ be polynomial space for $\mathcal{P}$. Since $q \in \mathbb{P}^1_\delta$, there exist two *even* polynomials $q_0, q_1 \in \mathbb{P}_{\mathcal{P}}$ such that

$$q = q_1\delta + q_0$$

Similarly, there exist two even polynomials $s_0, s_1 \in \mathbb{P}_{\mathcal{P}}$ such that

$$l = -s_1\delta - s_0$$

Using Part (5) of Lemma 7, we can write c_1 as

$$c_1 = \frac{(s_1 + q_1\mathcal{P})\delta + (s_0 + q_0\mathcal{P})}{2(q_1\delta + q_0)}. \tag{3.48}$$

To guarantee that HDG_k is dual equvialant with the SG scheme (3.21), we now relate δ with $\mathcal{P}$ by setting

$$c_1(\delta_k, \mathcal{P}) = \frac{\mathcal{P}\mathrm{e}^{\mathcal{P}}}{\mathrm{e}^{\mathcal{P}} - 1} = B\left(-\mathcal{P}\right). \tag{3.49}$$

Solving the above equation for δ we get

$$\delta_k \overset{\text{def}}{=} -\frac{\mathrm{e}^{\mathcal{P}}(s_0 - q_0\mathcal{P}) - (s_0 + q_0\mathcal{P})}{\mathrm{e}^{\mathcal{P}}(s_1 - q_1\mathcal{P}) - (s_1 + q_1\mathcal{P})}, \tag{3.50}$$

Using the same δ_k together with Part 3 and 1 of Lemma 7, we can similarly derive the other two coefficients c_2 and c_3 that coincide with those in (3.21). The proof is complete. □

To obtain some explicit values for δ_k, we implemented a symbolic routine that computes the values of $\mathbf{A}$ using SageMath [46], a computer algebra system (CAS) with features covering many aspects of mathematics. Our implemented software also inverts the matrix and computes explicitly c_1 using eq. (3.40). Finally, the values for δ_k can be obtained by decomposing c_1 as seen in eq. (3.48) and inserting the coefficients in eq. (3.50). Table 3.1 shows these values up to degree 4. In Figure 3.4, we also plot $\delta_k(\mathcal{P})$ as a function of $\mathcal{P}$ for $k \in \{0, 1, \ldots, 5\}$.

3.5.2 Numerical considerations for HDG_k

In this section, we want to replicate what we have done in Section 3.4.3 for $k > 0$. Indeed, Figure 3.4 shows that, from a qualitative point of view, the values of δ_k is similar for different values of k. Therefore, it is natural to wonder if this is true also for the overall behavior of the HDG_k methods.

Figure 3.5, indeed, compares the solutions generated by the HDG_k methods for different values of k and it shows that the plots obtained are qualitatively alike.

In Figure 3.6 and Figure 3.7, we realize the same plot of Figure 3.2, only for degree 1 and 2. We see that the situation is similar to the case of degree 0. In particular, we see that there is always only one point where the error $e_{\hat{u}}$ is zero, according to Theorem 1.

Finally, in Figure 3.8 and Figure 3.9 we see the order of convergence of HDG_1 and HDG_2, exactly as we did for HDG_0 in Figure 3.3. In these plots, we see that, for $k > 0$,

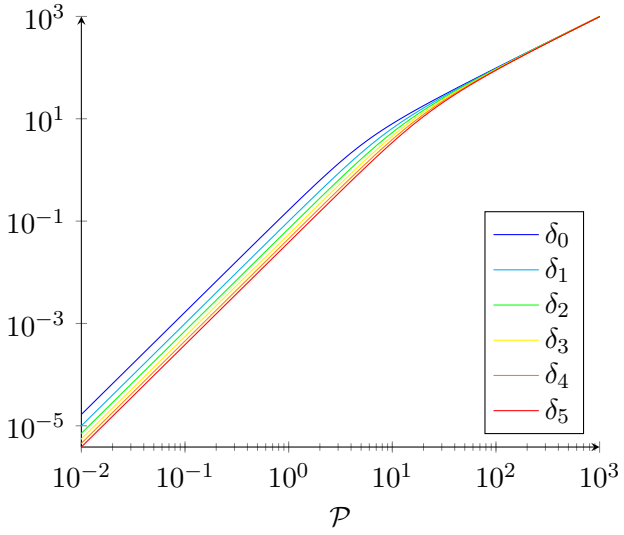


Figure 3.4: The values of δ_k for $k \in \{0, 1, \ldots, 5\}$.

δ_0	$-\dfrac{\mathrm{e}^{\mathcal{P}}\left(-\mathcal{P}+2\right)-\left(\mathcal{P}+2\right)}{\mathrm{e}^{\mathcal{P}}-1}$
δ_1	$-\dfrac{\mathrm{e}^{\mathcal{P}}\left(\mathcal{P}^2-6\,\mathcal{P}+12\right)-\left(\mathcal{P}^2+6\,\mathcal{P}+12\right)}{\mathrm{e}^{\mathcal{P}}\left(-\mathcal{P}+2\right)-\left(\mathcal{P}+2\right)}$
δ_2	$-\dfrac{\mathrm{e}^{\mathcal{P}}\left(-\mathcal{P}^3+12\,\mathcal{P}^2-60\,\mathcal{P}+120\right)-\left(\mathcal{P}^3+12\,\mathcal{P}^2+60\,\mathcal{P}+120\right)}{\mathrm{e}^{\mathcal{P}}\left(\mathcal{P}^2-6\,\mathcal{P}+12\right)-\left(\mathcal{P}^2+6\,\mathcal{P}+12\right)}$
δ_3	$-\dfrac{\mathrm{e}^{\mathcal{P}}\left(\mathcal{P}^4-20\,\mathcal{P}^3+180\,\mathcal{P}^2-840\,\mathcal{P}+1680\right)-\left(\mathcal{P}^4+20\,\mathcal{P}^3+180\,\mathcal{P}^2+840\,\mathcal{P}+1680\right)}{\mathrm{e}^{\mathcal{P}}\left(-\mathcal{P}^3+12\,\mathcal{P}^2-60\,\mathcal{P}+120\right)-\left(\mathcal{P}^3+12\,\mathcal{P}^2+60\,\mathcal{P}+120\right)}$
δ_4	$-\dfrac{\mathrm{e}^{\mathcal{P}}\left(-\mathcal{P}^5+30\,\mathcal{P}^4-420\,\mathcal{P}^3+3360\,\mathcal{P}^2-15120\,\mathcal{P}+30240\right)-\left(\mathcal{P}^5+30\,\mathcal{P}^4+420\,\mathcal{P}^3+3360\,\mathcal{P}^2+15120\,\mathcal{P}+30240\right)}{\mathrm{e}^{\mathcal{P}}\left(\mathcal{P}^4-20\,\mathcal{P}^3+180\,\mathcal{P}^2-840\,\mathcal{P}+1680\right)-\left(\mathcal{P}^4+20\,\mathcal{P}^3+180\,\mathcal{P}^2+840\,\mathcal{P}+1680\right)}$

Table 3.1: Values of δ_k for $k \in \{0,\ldots,4\}$. τ_k is defined as $\frac{\alpha}{h}\delta_k$

the error computed at the middle point for the solution obtained using τ_k has the same order of convergence of any other solution obtained with a fixed k even if the error, after a few refinements, become smaller than with any other solution. Moreover, as for degree 0, the error $\|e_{\mathbf{W}_u}\|_{L^2\Omega}$ is smaller than with any other solution.

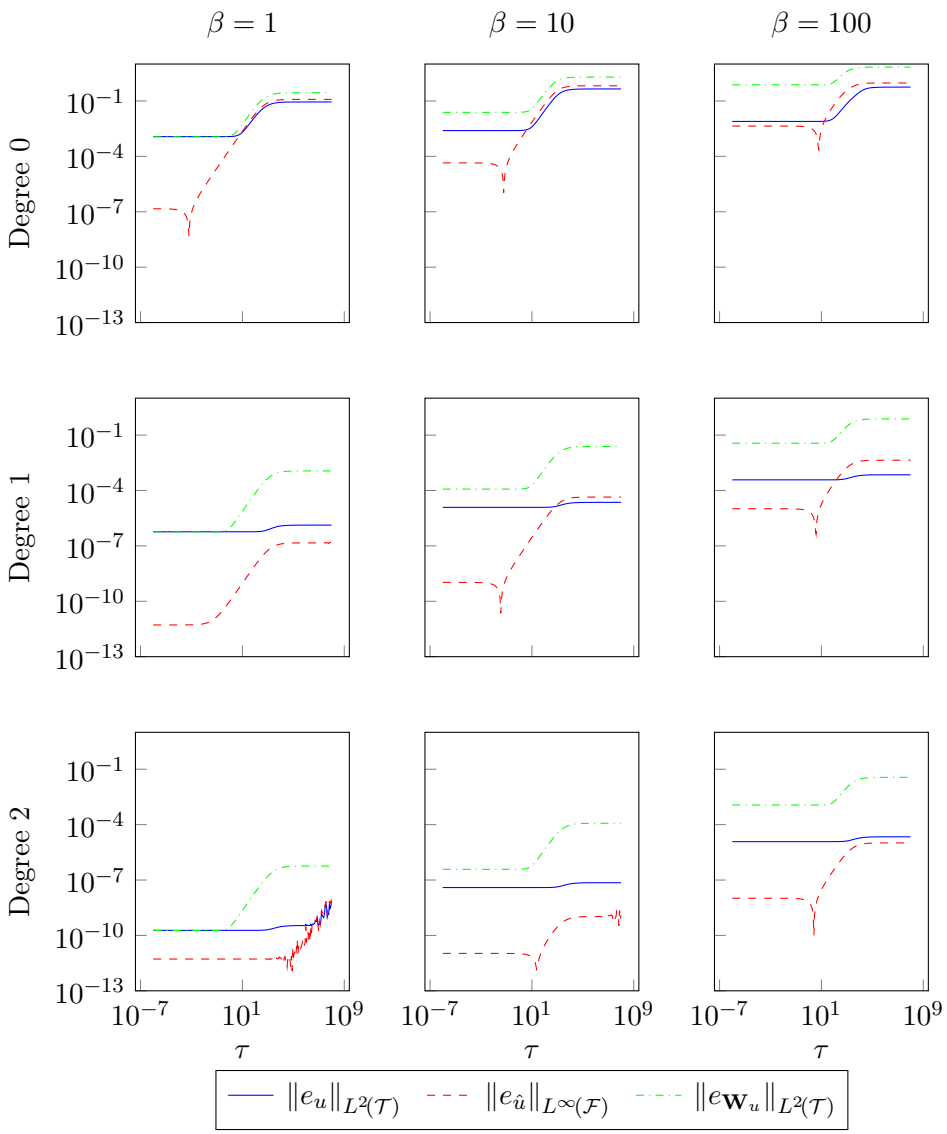


Figure 3.5: Errors obtained solving $\nabla\cdot(\boldsymbol{\beta}u - \nabla u) = 0$ on domain $\Omega = (0,1)$ with $\hat{u}(0) = 0$ and $\hat{u}(1) = 1$ for $h = \frac{1}{256}$.

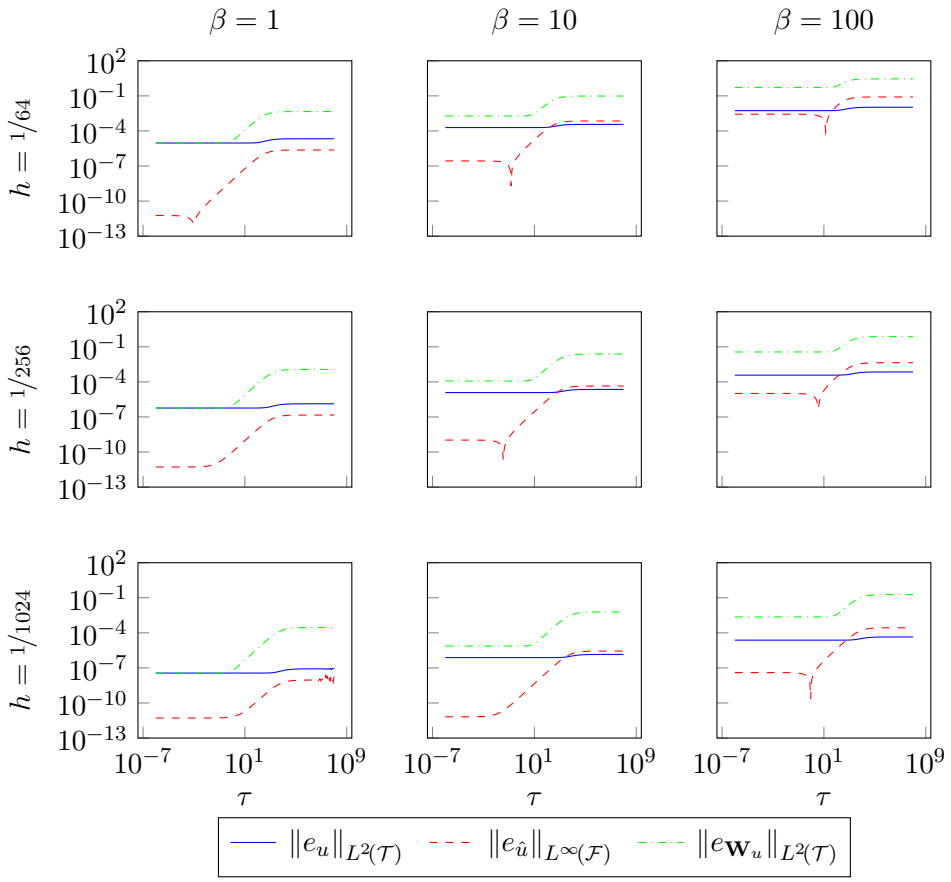


Figure 3.6: Errors obtained solving $\nabla\cdot(\boldsymbol{\beta}u - \nabla u) = 0$ on domain $\Omega = (0,1)$ with $\hat{u}(0) = 0$ and $\hat{u}(1) = 1$ using HDG$_1$.

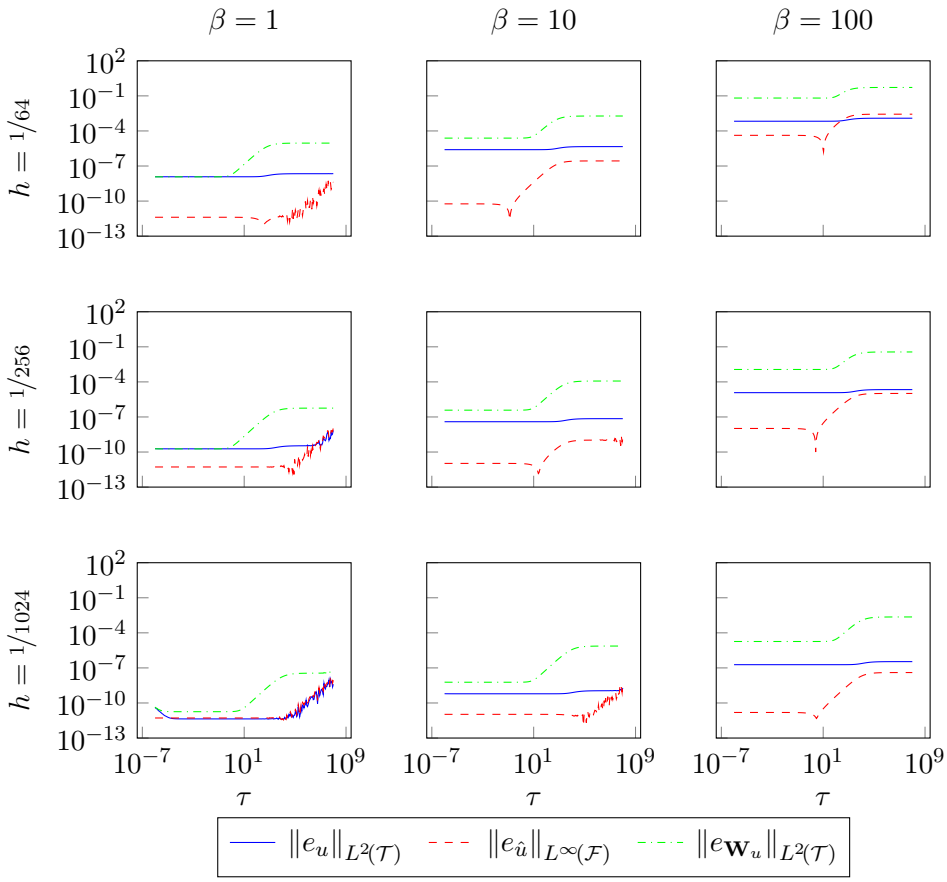


Figure 3.7: Errors obtained solving $\nabla \cdot (\boldsymbol{\beta} u - \nabla u) = 0$ on domain $\Omega = (0, 1)$ with $\hat{u}(0) = 0$ and $\hat{u}(1) = 1$ using HDG_2.

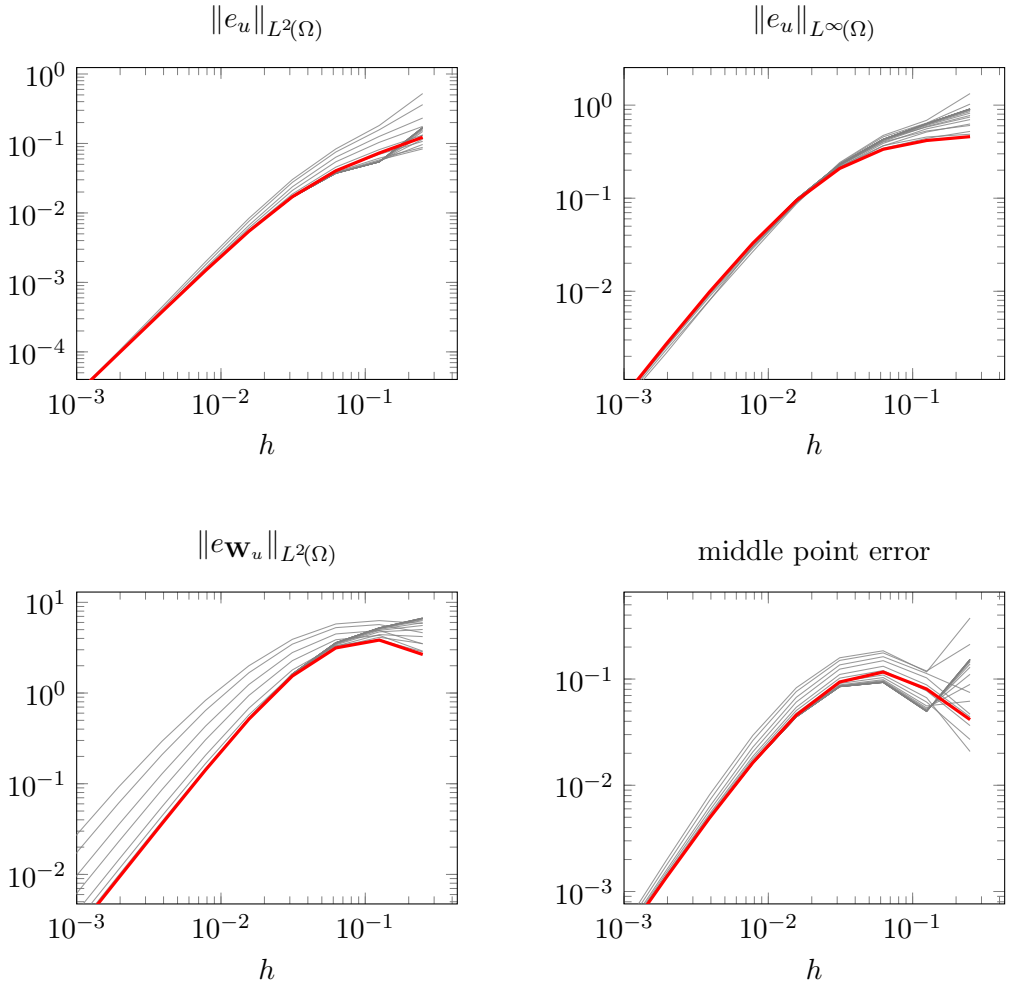


Figure 3.8: The error of the HDG$_1$ method as a function of the cell size h for a fixed value of the parameters α and β ($\alpha = 1$, $\beta = 100$) and for $f = 0$. The red line is the result using $\tau = \tau_1 = \frac{\alpha}{h}\delta_1$.

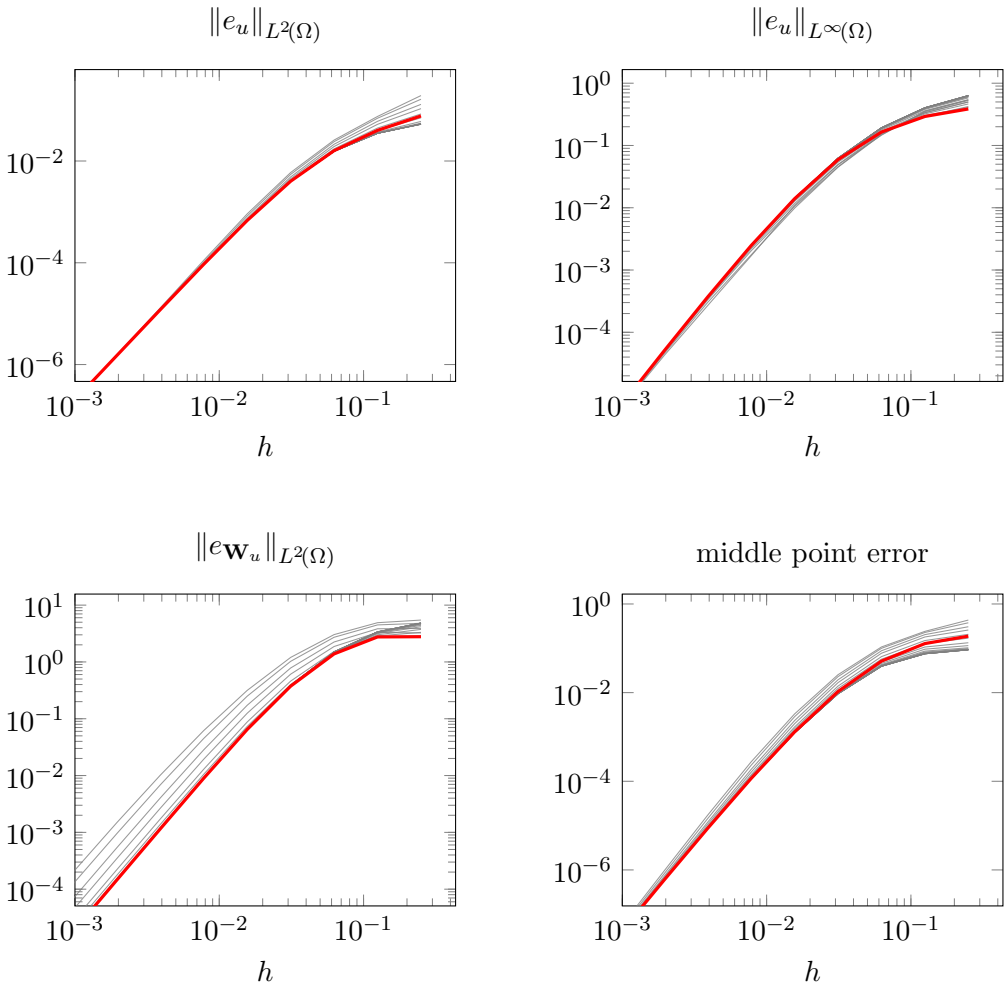


Figure 3.9: The error of the HDG_2 method as a function of the cell size h for a fixed value of the parameters α and β ($\alpha = 1$, $\beta = 100$) and for $f = 0$. The red line is the result using $\tau = \tau_2 = \frac{\alpha}{h}\delta_2$.

Chapter 4

Weighted HDG methods

4.1 Introduction

In the previous chapter we have shown a relationship between the HDG methods and the Scharfetter–Gummel scheme. Exploiting such relationship does not require to modify the original HDG scheme in any way beside choosing the stabilization parameter in a specific way; unfortunately, this result holds only on uniform one-dimensional grids and there is little to no hope to extend it to a more general setup without tailoring the formulation of the HDG methods.

In this chapter we introduce a modified version of the HDG methods, that we called *Weighted HDG* (W-HDG) that generalizes the Scharfetter–Gummel scheme in a slightly different way. We have previously shown that, for standard HDG methods, when $\tau \to 0$ the numerical solution on the trace converges to the one obtained by the finite volume method with a classical flux. When applying W-HDG methods with polynomial degree zero, instead, the convergence is towards the solution of the Scharfetter–Gummel scheme.

In the first part of this chapter we introduce the method and its relationship with the so called *Slootboom change of variables*, which essentially transforms a convection dominated problem into a pure diffusive one.

In Sections 4.4.5 to 4.4.8 we show that the proposed numerical method is well-posed and we compare its solutions to the one generated by the Scharfetter–Gummel scheme, justifying the interpretation of the W-HDG method as a high order generalization of the Scherfetter-Gummel stabilization.

In the last part of the chapter we validate numerically that W-HDG methods have the same properties of classical HDG methods, including optimal convergence and superconvergence of postprocessed solutions.

The content of this chapter has been submitted for publication as a standalone paper. To maintain consistency with the submitted manuscript, we maintained the original notations of the paper and therefore some notations may result slightly inconsistent with respect to the rest of the thesis.

4.2 Description of the problem

Given a function f defined on a bounded convex domain $\Omega \subset \mathbb{R}^d$, boundary data g_D and g_N respectively defined on Γ_D and Γ_N, where $\Gamma_D \cup \Gamma_N = \Gamma := \partial\Omega$, and $\Gamma_D \cap \Gamma_N = \emptyset$, we

want to find the pair $(u, \mathbf{j})$ satisfying

$$\mathbf{j} + \alpha \nabla u - \boldsymbol{\beta} u = 0, \qquad \text{in } \Omega, \tag{4.1a}$$
$$\nabla \cdot \mathbf{j} = f, \qquad \text{in } \Omega, \tag{4.1b}$$
$$u = g_D, \qquad \text{on } \Gamma_D, \tag{4.1c}$$
$$\mathbf{j} \cdot \mathbf{n} = g_N, \qquad \text{on } \Gamma_N. \tag{4.1d}$$

Here $\mathbf{n}$ denotes the outward-pointing normal vector of $\partial\Omega$, α is assumed to be a positive constant, and the vector field $\boldsymbol{\beta}$ is assumed to be in $[L^\infty(\Omega)]^d$.

When $\mathbf{j}$ is kept as an independent variable as in system (4.1), the problem is in its mixed form (see, e.g., [10]). By substituting $\mathbf{j}$ in equation (4.1b) with its definition in equation (4.1a) one obtains the classical primal formulation of the drift diffusion problem in divergence form.

If in addition the vector field $\boldsymbol{\beta}$ is the gradient of a potential ψ, i.e., $\boldsymbol{\beta} = -\nabla\psi$, one can rewrite both the primal and mixed formulation of the system using a change of variable

$$\mathbf{j} + \alpha \exp\left(-\frac{\psi}{\alpha}\right) \nabla s = 0, \qquad \text{in } \Omega, \tag{4.2a}$$
$$\nabla \cdot \mathbf{j} = f, \qquad \text{in } \Omega, \tag{4.2b}$$
$$s = g_D \exp\left(\frac{\psi}{\alpha}\right), \qquad \text{on } \Gamma_D, \tag{4.2c}$$
$$\mathbf{j} \cdot \mathbf{n} = g_N, \qquad \text{on } \Gamma_N. \tag{4.2d}$$

where the relation between u and s is given by

$$s = u \exp\left(\frac{\psi}{\alpha}\right), \tag{4.3}$$

and it is generally indicated as the Slotboom change of variables.

In system (4.1) the fluxes are split in drift and diffusive contributions, depending, respectively, on $\boldsymbol{\beta}$ and α. Rewriting system (4.1) using the set of variables $(s, \mathbf{j})$ in place of $(u, \mathbf{j})$, has the effect of transforming the problem into a purely diffusive one, with a highly nonlinear coefficient $\alpha \exp\left(-\frac{\psi}{\alpha}\right)$.

The analysis of such problems flourished in the context of semiconductor devices, where a combination of nonlinear systems with a similar nature of system (4.1) describes the behavior of the electrostatic potential and charge transport. The most famous version of such problems – the van Roosbroeck system – involves three different variables: the electrostatic potential (with the same nature of ψ), the electron density, and the hole density (both with the same nature of u).

The rewriting of the van Roosbroeck system as a purely diffusive problem using the Slotboom change of variable (4.3) enabled the first analytical result with zero right-hand side in semiconductor theory [36], and was also used more generally in [35]. From the numerical point of view in [11] this set of variables was used to generalize the exponential fitting method to two-dimensional problems.

The Slotboom formulation has mostly analytical advantages, since it allows to write the system in terms of quasi-linear partial differential equations, and one can apply classical analytical results that can be found for example in [45] and used in the context of semiconductors coupled with circuits in [2] and [3].

One of the main difficulties in using directly the Slotboom formulation is the fact that the nonlinear diffusion coefficients remains treatable only when the ratio $\frac{\psi}{\alpha}$ is of order one. In all other cases it rapidly blows up or goes to zero, rendering the numerical solution of the system difficult to deal with (see, e.g., [11]).

The idea behind our Weighted HDG formulation is to exploit a *local perturbation around the equilibruim* of the Slotboom transformation to eliminate the drift term from the system *element-wise*, as in equation (4.2a).

In order to have a mesh independent solution, we do not use directly the Slotboom variable s, but instead exploit weighted integrals to obtain the same effect of the Slotboom transformation, and we maintain the variable u as our system unknown.

Such formulation is possible by defining a locally shifted version of ψ which guarantees that $\exp(\frac{\psi}{\alpha})$ is of order one in every element of the discretization, and using it as a weight in the integrals that define the local HDG systems (see Section 4.4 for the details).

4.3 Notation

In this section, we introduce some notations used to define our numerical scheme for the solution of problem (4.1).

4.3.1 Subdivisions

Let us assume that the domain $\Omega \subset \mathbb{R}^d$ is a polytope. Let $\mathcal{T} := \mathcal{T}(\Omega)$ be a subdivision of Ω made of simplices; we call the elements of $\mathcal{T}$ "cells" and we call the intersection of two adjacent cells (or the intersection between a cell and $\partial\Omega$) with positive $(d-1)$-measure a "face". We denote with $\mathcal{E}^\circ$ and $\mathcal{E}^\partial$ the collection of these two kinds of faces and set $\mathcal{E} := \mathcal{E}^\circ \cup \mathcal{E}^\partial$. In what follows, we assume that $\mathcal{T}$ is shape-regular in the sense of [19], i.e., there exists a positive constant c_1 so that for each cell $K \in \mathcal{T}$,

$$h_K \leq c_1 \rho_K,$$

where h_K is the size of K and ρ_K is the diameter of the largest ball inscribed in K.

4.3.2 Approximation spaces and jump operators

For $K \in \mathcal{T}$, we define $\mathcal{P}_k(K)$ to be the space of polynomials defined on K with degree at most k. The vector space $\boldsymbol{\mathcal{Q}}(K) = [\mathcal{P}_k(K)]^d$ is used to approximate the flux $\mathbf{j}$, while the space $\mathcal{W}(K) := \mathcal{P}_k(K)$ is used for approximating the scalar solution u in each element K. The corresponding global finite element spaces $\boldsymbol{\mathcal{Q}}(\mathcal{T})$ and $\mathcal{W}(\mathcal{T})$ are respectively

$$\boldsymbol{\mathcal{Q}}(\mathcal{T}) := \{\mathbf{Q} \in [L^2(\Omega)]^d : \mathbf{Q}|_K \in \boldsymbol{\mathcal{Q}}(K)\}$$

and

$$\mathcal{W}(\mathcal{T}) := \{W \in L^2(\Omega) : W|_K \in \mathcal{W}(K)\}.$$

We note that the discontinuous functions $\mathbf{Q} \in \boldsymbol{\mathcal{Q}}(\mathcal{T})$ and $w \in \mathcal{W}(K)$ are double valued when they are restricted to each interior face $e \in \mathcal{E}^\circ$. For two cells K^+, K^- such that $e = K^+ \cap K^-$, we define the branches $W_{K^\pm}(\mathbf{x}) := \lim_{\epsilon \to 0} W(\mathbf{x} - \epsilon \mathbf{n}_{K^\pm})$ for all $\mathbf{x} \in e$, where $\mathbf{n}_{K^\pm}$ denotes the outward normal vectors of $K^\pm$. The notation for the vector field $\mathbf{Q}_{K^\pm}$ on e is similar. We also define the jump on $e \in \mathcal{E}^\circ$,

$$[\![W]\!]_e := W_{K^-}\mathbf{n}_{K^-} + W_{K^+}\mathbf{n}_{K^+} \quad \text{and} \quad [\![\mathbf{Q}]\!]_e := \mathbf{Q}_{K^-} \cdot \mathbf{n}_{K^-} + \mathbf{Q}_{K^+} \cdot \mathbf{n}_{K^+}.$$

By convention, for $e \in \mathcal{E}^\partial \cap K$ for some cell K, we define $[\![W]\!]_e = W_K \mathbf{n}_K$ and $[\![\mathbf{Q}]\!]_e = \mathbf{Q}_K \cdot \mathbf{n}_K$.

For each face $e \in \mathcal{E}$, we similarly define $\mathcal{M}(e) = \mathcal{P}_k(e)$ the polynomial space on e and

$$\mathcal{M}(\partial K) := \{\Lambda \in L^2(\mathcal{E}) : \Lambda|_e \in \mathcal{P}_k(e) \text{ for each face of } K\}.$$

The definition of the space $\mathcal{M}(\mathcal{E})$ follows the same idea. Globally, we also define

$$\begin{aligned} \mathcal{M}_D(\mathcal{E}) := \{\Lambda \in L^2(\mathcal{E}) : \Lambda|_e \in \mathcal{P}_k(e) \\ \text{for each } e \in \mathcal{E}\backslash\Gamma_D \text{ and } \Lambda|_e = G_D, \text{ for each } e \in \mathcal{E} \cap \Gamma_D\}, \end{aligned} \tag{4.4}$$

where $G_D \in \mathcal{M}(\Gamma_D)$ is an approximation of g_D on Γ_D. In particular, we use the space $\mathcal{M}_0(\mathcal{E})$ for $\mathcal{M}_D(\mathcal{E})$ if $G_D = 0$.

4.3.3 Inner products and norms

For $K \in \mathcal{T}$, $e \in \mathcal{E}_K := \mathcal{E} \cap \partial K$, and for a positive weight function μ_K defined on K, we set

$$(u, v)_{K,\mu_K} := \int_K \mu_K uv \, \mathrm{d}\mathbf{x} \quad \text{and} \quad \langle u, v\rangle_{e,\mu_K} := \int_e \mu_K uv \, \mathrm{d}s_{\mathbf{x}},$$

and define the norm $\|.\|_{K,\mu_K} = \sqrt{(.,.)_{K,\mu_K}}$. We also set

$$\langle u, v\rangle_{\partial K,\mu_K} := \sum_{e \in \mathcal{E}_K} \int_e \mu_K uv \, \mathrm{d}s_{\mathbf{x}}.$$

Globally, if μ is defined as a cell-wise function so that $\mu|_K = \mu_K$, we introduce the weighted inner-product

$$(u, v)_{\mathcal{T},\mu} := \sum_{K \in \mathcal{T}} (u, v)_{K,\mu}.$$

We set the norm $\|.\|_\mu := \sqrt{(.,.)_{\mathcal{T},\mu}}$. If the weight function $\mu \equiv 1$ in Ω, we omit the subscripts μ or μ_K.

4.4 Weighted HDG methods

In the section, we introduce our HDG method in the spirit of [17] assuming that α is a positive constant, and that $\boldsymbol{\beta}$ is a piecewise constant function subordinate to a subdivision $\mathcal{T}$ of Ω. Then there exists a function $\psi \in W^{1,\infty}(\mathcal{T})$ such that

$$\boldsymbol{\beta} = -\nabla\psi \qquad \text{locally on each } K. \tag{4.5}$$

Notice that, in general, the function ψ is not continuous across elements of $\mathcal{T}$.

4.4.1 Discrete formulation of the local problem

Let us start by considering problem (4.1) in a cell $K \in \mathcal{T}$. We multiply equation (4.1a) with a weight function

$$\mu_K(\mathbf{x}) := \exp\left(\frac{\psi(\mathbf{x})}{\alpha}\right) \tag{4.6}$$

satisfying that $\nabla\mu_K = \frac{\nabla\psi}{\alpha}\mu_K = -\frac{\boldsymbol{\beta}}{\alpha}\mu_K$. Whence,

$$\mu_K\mathbf{j} + \alpha\nabla(\mu_K u) = 0, \quad \text{in } K \tag{4.7}$$

Remark 10. *We note that the above equation would make sense globally (i.e., on the entire Ω) when $\psi \in W^{1,\infty}(\Omega)$ and not just on each element separately. This assumption is satisfied, for example, by the Poisson equation for the electrostatic potential ψ in the van Roosbroeck system. According to [35, Theorem 3.2.1], we can assume that the right hand side of the Poisson problem is in $L^p(\Omega)$ if the doping concentration is assumed to be an $L^p(\Omega)$ function with $p > d$. Whence by elliptic regularity $\psi \in W^{2,p}(\Omega)$ (cf. [24, Theorem 4.3.2.2]) and we guarantee that $\nabla\psi$ is Lipschitz by Sobolev embedding theorem.*

Next, we test (4.7) with a vector-valued function $\mathbf{Q} \in \boldsymbol{\mathcal{Q}}(K)$ and multiply (4.1b) with μW where $W \in \mathcal{W}(K)$. Using integration by parts for the term containing $\nabla(\mu_K u)$ and $\nabla\cdot\mathbf{j}$, our discrete weak formulation becomes: given a local Dirichlet boundary condition $\widehat{U} \in \mathcal{M}(\partial K)$ and $f \in L^2(K)$, find $(\mathbf{J}, U) \in \boldsymbol{\mathcal{Q}}(K) \times \mathcal{W}(K)$ satisfying that for all $\mathbf{Q} \in \boldsymbol{\mathcal{Q}}(K)$ and $W \in \mathcal{W}(K)$,

$$(\mathbf{J}, \mathbf{Q})_{K,\mu_K} - (\alpha U, \nabla\cdot\mathbf{Q})_{K,\mu_K} + \langle\alpha\widehat{U}, \mathbf{Q}\cdot\mathbf{n}\rangle_{\partial K,\mu_K} = 0, \tag{4.8a}$$

$$-(\mathbf{J}, \nabla W - \tfrac{\boldsymbol{\beta}}{\alpha}W)_{K,\mu_K} + \langle\widehat{\mathbf{J}}\cdot\mathbf{n}, W\rangle_{\partial K,\mu_K} = (f, W)_{K,\mu_K}. \tag{4.8b}$$

Here the numerical flux $\widehat{\mathbf{J}}$ is given by

$$\widehat{\mathbf{J}} = \mathbf{J} + \tau_K(U - \widehat{U})\mathbf{n}, \tag{4.9}$$

with $\tau_K \geq 0$ denoting a piecewise constant function (independent of h_K) defined on ∂K.

Remark 11 (Constant $\boldsymbol{\beta}$)**.** *In particular, if $\boldsymbol{\beta} = -\nabla\psi$ is a constant vector, we shall use the following weight function*

$$\mu_K := \exp\left(-\frac{\boldsymbol{\beta}\cdot(\mathbf{x} - \mathbf{x}_K)}{\alpha}\right), \tag{4.10}$$

where $\{\mathbf{x}_K\}_{K\in\mathcal{T}}$ *is a set of points in* $\mathbb{R}^d$*. As a consequence, the local problem* (4.8) *is rescaled by the factor* $\exp(-\boldsymbol{\beta}\cdot\mathbf{x}_K/\alpha)$*. We will use the weight function* (4.10) *both for the analysis and for computational purposes by choosing different sets of* $\{\mathbf{x}_K\}$*; see Section 4.4.6 and Section 4.6 for some particular choices.*

The following proposition shows that the local problem admits a unique solution if the stabilization parameter is strictly positive on at least one face of K.

Proposition 12 (Well-posedness of the local problem)**.** *The local problem* (4.8) *is uniquely solvable if* $\tau_K \geq 0$ *and it is strictly positive on at least one face of* K*.*

Proof. This proof follows closely [17, Proposition 3.2]. We report it here for completeness. It is sufficient to show that when $\widehat{U}=0$ on ∂K and $f=0$ in K, both $\mathbf{J}$ and $U=0$ are trivial. To this end, we first apply integration by parts for the first term on the left hand side of (4.8b) and insert (4.9). Whence,

$$(\nabla\cdot\mathbf{J},W)_{K,\mu_K}+\langle\tau_K(U-\widehat{U}),W\rangle_{\partial K,\mu_K}=0. \tag{4.11}$$

We choose $\mathbf{Q}=\mathbf{J}/\alpha$ in (4.8a) and $W=U$ in (4.11) and sum up these two equations to get

$$\frac{1}{\alpha}(\mathbf{J},\mathbf{J})_{K,\mu_K}+\langle\tau_K U,U\rangle_{\partial K,\mu_K}=0. \tag{4.12}$$

Since $\tau_K\geq 0$, we immediately get $\mathbf{J}_\Lambda=0$ and $\sqrt{\tau_K}U_\Lambda=0$. So $\tau_K U_\Lambda=0$. By assumption, τ_K is strictly positive at some face e of K. Then $U=0$ on e. If the polynomial degree $k=0$, we get $U=0$ and the proof is complete. Otherwise, if $k>0$, using the barycentric coordinate system, we can write $U=l_e p_{k-1}$ with $p_{k-1}\in\mathcal{P}_{k-1}(K)$ and where l_e is the corresponding barycentric coordinate function that vanishes on e. Since we have derived that $\mathbf{J}=0$, according to (4.8a) with $\mathbf{Q}$ satisfying $\nabla\cdot\mathbf{Q}=p_{k-1}$ (this is because the divergence operator mapping from $[\mathcal{P}_k(K)]^d$ to $\mathcal{P}_{k-1}(K)$ is surjective), we have

$$0=(\alpha U,p_{k-1})_{K,\mu_K}=\int_K \alpha\mu_K l_e p_{k-1}^2.$$

So $p_{k-1}=0$ and we conclude that $U_\Lambda=0$. □

Remark 13. *Proposition 12 implies that there exists a linear operator* T *mapping the data pair* $(\widehat{U},f)$ *to the solution pair* $(\mathbf{J},U)$*.*

4.4.2 Global system

At the continuous level, the global system (4.1) can be understood by combining the local system (4.7) and (4.1b) in each cell K and glue them together by the following transmission condition

$$[\![\mathbf{j}]\!]_e=0,\qquad \forall e\in\mathcal{E}^\circ. \tag{4.13}$$

The boundary conditions follows from (4.1c) and (4.1d). Now for each face $e\in\mathcal{E}$, we shall impose (4.13), (4.1c) and (4.1d) weakly using the $L^2(e)$ inner product as well as the

discrete counterpart $(\widehat{\mathbf{J}}, \widehat{U})$. More precisely speaking, given the numerical flux $\widehat{\mathbf{J}}$ defined by (4.9), for each face $e \in \mathcal{E}^\circ$,

$$\langle [\![\widehat{\mathbf{J}}]\!], \xi \rangle = 0, \quad \forall \xi \in \mathcal{M}(e) \tag{4.14}$$

and for each $e \in \mathcal{E}^\partial$ and $\xi \in \mathcal{M}(e)$,

$$\langle \widehat{U}, \xi \rangle_e = \langle g_D, \xi \rangle_e, \text{ if } e \in \Gamma_D \quad \text{and} \quad \langle \widehat{\mathbf{J}} \cdot \mathbf{n}, \xi \rangle_e = \langle g_N, \xi \rangle_e \text{ if } e \in \Gamma_N. \tag{4.15}$$

Noting that $\widehat{U} = G_D := \pi g_D$ with π denoting the piecewise L^2 projection on $\Gamma_D(\mathcal{E})$. We also observe that since $[\![\widehat{\mathbf{J}}]\!] \in \mathcal{M}(e)$ and the formulation (4.14) immediately implies that $[\![\widehat{\mathbf{J}}]\!] = 0$ on $\mathcal{E}^\circ$. Whence $\widehat{\mathbf{J}}$ is single-valued on $\mathcal{E}$.

Our discrete formulation reads: find $(\mathbf{J}, U, \widehat{U}) \in \boldsymbol{\mathcal{Q}}(\mathcal{T}) \times \mathcal{W}(K) \times \mathcal{M}_D$ such that for all $\mathbf{Q} \in \boldsymbol{\mathcal{Q}}(\mathcal{T})$, $W \in \mathcal{W}(\mathcal{T})$ and $\xi \in \mathcal{M}_0(\mathcal{E})$,

$$\begin{aligned} (\mathbf{J}, \mathbf{Q})_{\mathcal{T},\mu} - (\alpha U, \nabla \cdot \mathbf{Q})_{\mathcal{T},\mu} + \sum_{K \in \mathcal{T}} \langle \alpha \widehat{U}, \mathbf{Q} \cdot \mathbf{n} \rangle_{\partial K,\mu} &= 0, \\ (\nabla \cdot \mathbf{J}, W)_{\mathcal{T},\mu} + \sum_{K \in \mathcal{T}} \langle \tau_K (U - \widehat{U}), W \rangle_{\partial K,\mu} &= (f, W)_{\mathcal{T},\mu}, \\ \sum_{K \in \mathcal{T}} \langle \mathbf{J} \cdot \mathbf{n} + \tau_K (U - \widehat{U}), \xi \rangle_{\partial K} &= \sum_{e \in \partial \Gamma_N} \langle g_N, \xi \rangle_e. \end{aligned} \tag{4.16}$$

4.4.3 Characterization of $\widehat{U}$

According to Remark 13, $\mathbf{J}$ and U can be written as an operator acting on $\widehat{U}$ and f. So when solving the discrete system (4.16), we can first eliminate the unknowns $\mathbf{J}$ and U to rewrite the system only for Λ. In this subsection, we shall derive such system for $\widehat{U}$ in order to investigate the well-posedness of (4.16). Using the notations from Remark 13, we denote the solution triplets

$$(\mathbf{J}_0, U_0, \widehat{\mathbf{J}}_0) \quad \text{and} \quad (\mathbf{J}_f, U_f, \widehat{\mathbf{J}}_f) \tag{4.17}$$

when f and g_D vanish respectively. More precisely, based on the discrete transmission condition (4.14) as well as the Neumann boundary condition defined in Equation (4.15), the solution $\widehat{U} \in \mathcal{M}_D$ satisfies

$$a(\widehat{U}, \xi) = b(\xi), \quad \forall \xi \in \mathcal{M}_0, \tag{4.18}$$

where

$$\begin{aligned} a(\widehat{U}, \xi) &= -\langle [\![\widehat{\mathbf{J}}_0]\!], \xi \rangle_{\mathcal{E}} \\ b(\xi) &= \langle [\![\widehat{\mathbf{J}}_f]\!], \xi \rangle_{\mathcal{E}} - \langle g_N, \xi \rangle_{\Gamma_N} \end{aligned}$$

4.4.4 A decomposition of the bilinear form $a(.,.)$

Let μ be a positive function defined in Ω so that $\mu|_K = \mu_K$ for every $K \in \mathcal{T}$. Then we decompose the bilinear form $a(.,.)$ by

$$a(\widehat{U}, \xi) = \widetilde{a}(\widehat{U}, \xi) + E(\widehat{U}, \xi), \tag{4.19}$$

where

$$\widetilde{a}(\widehat{U}, \xi) = -\langle [\![\mu \widehat{\mathbf{J}}_0]\!], \xi \rangle_{\mathcal{E}} \quad \text{and} \quad E(\widehat{U}, \xi) = -\big(\langle [\![\widehat{\mathbf{J}}_0]\!], \xi \rangle_{\mathcal{E}} + \langle [\![\mu \widehat{\mathbf{J}}_0]\!], \xi \rangle_{\mathcal{E}}\big).$$

The following lemma provides a characterization of the bilinear form $\widetilde{a}(.,.)$ assuming zero Dirichlet boundary condition.

Lemma 14 (characterization of $\widetilde{a}$)**.** *The bilinear form $\widetilde{a}(.,.)$ is symmetric semidefinite on $\mathcal{M}_0 \times \mathcal{M}_0$. Moreover, given any $\Lambda \in \mathcal{M}_0$, we denote $(\mathbf{J}_\Lambda, U_\Lambda)$ the corresponding cell solutions by solving the local problem* (4.8) *with $f = 0$ and recall* (4.17)*. Then there holds that for $\xi \in \mathcal{M}_0$,*

$$\widetilde{a}(\Lambda, \xi) = \frac{1}{\alpha}(\mathbf{J}_\xi, \mathbf{J}_\Lambda)_{\mathcal{T},\mu} + \langle [\![\mu\tau(U_\Lambda - \Lambda)(U_\xi - \xi)]\!], 1\rangle_{\mathcal{E}} \tag{4.20}$$

where τ is a function defined on $\mathcal{E}$ so that $\tau|_{\partial K} = \tau_K$.

Proof. Before deriving (4.20), we first provide some essential identities. By summing up the local flux problem (4.8a) with $\widehat{U} = \lambda$ for all $K \in \mathcal{T}$, we have

$$\frac{1}{\alpha}(\mathbf{J}_\Lambda, \mathbf{Q})_{\mathcal{T},\mu} - (U_\Lambda, \nabla \cdot \mathbf{Q})_{\mathcal{T},\mu} = -\langle \Lambda, [\![\mu \mathbf{Q}]\!] \rangle_{\mathcal{E}} \tag{4.21}$$

We use integration by parts for the first term on the left hand side of (4.8b) and sum up the equations for all $K \in \mathcal{T}$ to obtain that (note that $f = 0$)

$$(\nabla \cdot \mathbf{J}_\Lambda, W)_{\mathcal{T},\mu} = \langle [\![\mu(\mathbf{J}_\Lambda - \widehat{\mathbf{J}}_\Lambda)W]\!], 1\rangle_{\mathcal{E}}. \tag{4.22}$$

Using the above relations, we arrive at

$$\begin{aligned}
\widetilde{a}(\Lambda, \xi) &= -\langle [\![\mu \mathbf{J}_\Lambda]\!], \xi \rangle_{\mathcal{E}} - \langle [\![\mu(\widehat{\mathbf{J}}_\Lambda - \mathbf{J}_\Lambda)]\!], \xi \rangle_{\mathcal{E}} \\
&= \frac{1}{\alpha}(\mathbf{J}_\xi, \mathbf{J}_\Lambda)_{\mathcal{T},\mu} - (U_\xi, \nabla \cdot \mathbf{J}_\Lambda)_{\mathcal{T},\mu} \\
&\quad - \langle [\![\mu(\widehat{\mathbf{J}}_\Lambda - \mathbf{J}_\Lambda)]\!], \xi \rangle_{\mathcal{E}} && \text{by (4.21) with } \Lambda = \xi, \mathbf{Q} = \mathbf{J}_\Lambda \\
&= \frac{1}{\alpha}(\mathbf{J}_\xi, \mathbf{J}_\Lambda)_{\mathcal{T},\mu} + \langle [\![\mu(\widehat{\mathbf{J}}_\Lambda - \mathbf{J}_\Lambda)(U_\xi - \xi)]\!], 1\rangle_{\mathcal{E}} && \text{by (4.22) with } W = U_\xi.
\end{aligned}$$

The proof in complete by inserting the definition of the numerical flux (4.9). □

Remark 15. *The above theorem also shows that the local problems* (4.8) *for all $K \in \mathcal{T}$ together with the weak transmission condition*

$$\langle [\![\mu \widehat{\mathbf{J}}]\!], \xi \rangle_{\mathcal{E}} = 0, \quad \forall \xi \in \mathcal{M}_0.$$

form a symmetric discrete system.

Remark 16. *For $\Lambda \in \mathcal{M}_0$, we recall that*

$$0 \le \widetilde{a}(\Lambda, \Lambda) = \sum_{K\in\mathcal{T}} -\langle \widehat{\mathbf{J}}_\Lambda, \Lambda\rangle_{\partial K,\mu} =: \sum_{K\in\mathcal{T}} I_K.$$

We can also show the above by proving the nonnegativity of I_K. Indeed, let $\widehat{U} = \Lambda$, $\mathbf{Q} = \mathbf{J}$ and $W = U$ in (4.8) *and sum up the equations with $f = 0$ to derive that*

$$I_K = \frac{1}{\alpha}(\mathbf{J}_\Lambda, \mathbf{J}_\Lambda)_{K,\mu} + \langle \tau_K(U_\Lambda - \Lambda), U_\Lambda - \Lambda\rangle_{\partial K,\mu} \ge 0.$$

4.4.5 Well-posedness of the global problem (4.18) for constant $\boldsymbol{\beta}$

Given a constant $\boldsymbol{\beta}$, we choose $\mathbf{x}_K$ to be the origin for all $K \in \mathcal{T}$. So the weight function μ defined by (4.10) is continuous. Using this information, we can show that (4.18) is well-posed in the following theorem.

Theorem 2. *Assume that i) ψ is continuous and that $\boldsymbol{\beta} = -\nabla\psi$ is piecewise constant on $\mathcal{T}$, ii) τ_K is positive on ∂K for all $K \in \mathcal{T}$. Then problem* (4.18) *is uniquely solvable.*

Proof. It suffices to show that given vanishing f, g_D, g_N such that $a(\Lambda, \xi) = 0$ for all $\xi \in \mathcal{M}_0$, we can only obtain trivial solutions. Due to the continuity of the weight function μ by choosing $\mathbf{x}_K$ to be the origin and the fact that $\widehat{\mathbf{J}}_\Lambda$ is single-valued, according to (4.19), we have $E(\Lambda, \xi) = 0$ and hence $0 = a(\Lambda, \xi) = \widetilde{a}(\Lambda, \xi)$. Invoking Lemma 14 as well as Remark 16, we immediately obtain $\mathbf{J}_\Lambda = 0$ in K and $\tau_K(U_\Lambda - \Lambda) = 0$ on ∂K. By assumption for τ_K, $\Lambda = U_\Lambda$ on ∂K. We next want to show that $U_\Lambda = 0$. Applying integration by parts in (4.8a) implies that

$$(\nabla(\mu U_\Lambda), \mathbf{Q})_K - \langle \mu(U_\Lambda - \Lambda), \mathbf{Q}\cdot\mathbf{n}\rangle_{\partial K} = 0. \tag{4.23}$$

We again apply $U_\Lambda = \Lambda$ on ∂K into (4.23) to get

$$(\nabla(\mu U_\Lambda), \mathbf{Q})_K = 0, \quad \forall \mathbf{Q} \in \boldsymbol{\mathcal{Q}}(K). \tag{4.24}$$

Letting $\mathbf{Q} = \nabla U_\Lambda - \frac{\boldsymbol{\beta}}{\alpha} U_\Lambda \in \boldsymbol{\mathcal{Q}}(K)$ implies that

$$0 = (\nabla(\mu U_\Lambda), \mathbf{Q})_K = (\nabla(\mu U_\Lambda), \nabla(\mu U_\Lambda))_{K,1/\mu}.$$

Hence, we deduce that $U_\Lambda = C_K/\mu$ for some constant C_K in each K and $\Lambda = U_\Lambda$ on ∂K. If $\boldsymbol{\beta} = 0$, we derive that U_Λ is a constant and hence $U_\Lambda = 0$ due to the vanishing Dirichlet boundary condition. If $\boldsymbol{\beta} \neq 0$, as $U_\Lambda \in \mathcal{W}(K)$, we conclude that $U_\Lambda = 0$ and hence $\Lambda = 0$. The proof is complete. □

Remark 17. *We can extend the above argument to show the well-posedness of the global system for the case when $\boldsymbol{\beta} = -\nabla\psi$ with $\psi \in C^0(\Omega) \cap \mathcal{P}_1(\mathcal{T})$. So $\boldsymbol{\beta}$ is a piecewise constant function and μ is continuous. This will be used to develop the numerical scheme for drift-diffusion system when the electric potential ψ is approximated by continuous piecewise linear functions subordinate to $\mathcal{T}$.*

4.4.6 Well-posedness for the one-dimensional case with $\boldsymbol{\beta} \in L^\infty(\Omega)$

If $\Omega = [a, b] \subset \mathbb{R}$, we can further assume that $\boldsymbol{\beta}$ is a piecewise constant function even though ψ may not be continuous.

Corollary 18. *Assuming that $\boldsymbol{\beta}$ is a piecewise constant function with $\boldsymbol{\beta}_K := \boldsymbol{\beta}|_K$. We also assume that τ_K is positive on $\mathcal{E}$. Then problem* (4.18) *is uniquely solvable.*

Proof. Our goal is to construct a continuous weight function μ so that $E(U_\Lambda, \xi) = 0$ in the proof of Theorem 2. We set $\{\mathbf{x}_i\}_{i=0}^N$ are the grid points so that $a = \mathbf{x}_0 < \ldots < \mathbf{x}_N = b$ and set $K_i = [\mathbf{x}_{i-1}, \mathbf{x}_i]$. For each $i = 1, \ldots, N-1$, we define μ_{K_i} and $\mu_{K_{i+1}}$ so that μ is continuous at $\mathbf{x}_i$. To achieve this, we set

$$\frac{\boldsymbol{\beta}_{K_i}}{\alpha}(\mathbf{x}_i - \mathbf{x}_{K_i}) = \frac{\boldsymbol{\beta}_{K_{i+1}}}{\alpha}(\mathbf{x}_i - \mathbf{x}_{K_{i+1}}). \tag{4.25}$$

Given a fixed $\mathbf{x}_{K_1} \in \mathbb{R}$, we can solve for $\mathbf{x}_{K_i}$ for $i = 1, \ldots, N$ based on (4.25) and hence the resulting μ is continuous. The proof is complete according to the proof of Theorem 2. □

4.4.7 Well-posedness for a general case

In this section we want to extend the assumption that there exists a function $\psi \in W^{1,\infty}(\Omega)$ so that $\boldsymbol{\beta} = -\nabla\psi \in [L^\infty(\Omega)]^d$. We then choose the weight function (4.6). To show the well-posedness of the global problem (4.18), we shall use the Poincaré inequality: for $w \in H_D^1(\Omega)$, there exists a constant $C_p \sim C\,\mathrm{diam}(\Omega)$ satisfying

$$\|w\|_{L^2(\Omega)} \le C_p \|\nabla w\|_{L^2(\Omega)}.$$

Theorem 3. *Suppose that $\psi \in W^{1,\infty}(\Omega)$. We further assume that $\|\beta\|_{L^\infty(\Omega)}$ is small enough such that the Poincaré constant satisfies*

$$\|\boldsymbol{\beta}\|_{L^\infty(\Omega)} < \frac{2\alpha}{C_p} \tag{4.26}$$

Under the assumption that τ_K is positive on ∂K for all $K \in \mathcal{T}$, the discrete problem (4.18) *is then uniquely solvable.*

Proof. Following the proof of Theorem 2, let us start our proof at (4.24). Under the current setting, $\mathbf{Q} = \nabla U_\Lambda - \frac{\boldsymbol{\beta}}{\alpha} U_\Lambda \notin \boldsymbol{\mathcal{Q}}(K)$. Here we choose $\mathbf{Q} = \nabla U_\Lambda$. We rewrite (4.24) to get

$$0 = (\nabla(\mu U_\Lambda), \nabla U)_K = \|\nabla\left(\mu^{1/2} U_\Lambda\right)\|_{L^2(K)}^2 - \left\|\frac{|\boldsymbol{\beta}|}{2\alpha}\mu^{1/2} U_\Lambda\right\|_{L^2(K)}^2.$$

Now we sum up the above equation for all $K \in \mathcal{T}$. Note that $U_\Lambda = \Lambda$ implies that $U_\Lambda \in H_D^1(\Omega)$. So $\mu^{1/2} U_\Lambda \in H_D^1(\Omega)$. By Poincaré inequality, we have

$$\begin{aligned} 0 &= \|\nabla\left(\mu^{1/2} U_\Lambda\right)\|_{L^2(\Omega)}^2 - \left\|\frac{|\boldsymbol{\beta}|}{2\alpha}\mu^{1/2} U_\Lambda\right\|_{L^2(\Omega)}^2 \\ &\ge \left(\frac{1}{C_p^2} - \frac{\|\boldsymbol{\beta}\|_{L^\infty(\Omega)}^2}{4\alpha^2}\right)\|\mu^{1/2} U_\Lambda\|_{L^2(\Omega)}^2. \end{aligned}$$

Assumption (4.26) shows that the constant on the right hand side above is strictly positive. Hence $U_\Lambda = 0$ and the proof is complete. □

4.4.8 A limiting case in the one-dimensional setting

Let us consider the numerical scheme (4.16) in 1d with $k = 0$, i.e., using piecewise constant functions. Set $\Omega = (a, b)$ for some real numbers $a < b$ and Ω can be subdivided with $a = \mathbf{x}_0 < \ldots < \mathbf{x}_N = b$. For $i = 0, \ldots, N-1$, denote $h_i = \mathbf{x}_{i+1} - \mathbf{x}_i$. We also denote $\{\Lambda_i\}_{i=0}^N$ the trace values on $\mathbf{x}_i$ and $\{(\mathbf{J}_i, U_i)\}_{i=1}^N$ the cell values on the interval $(\mathbf{x}_{i-1}, \mathbf{x}_i)$.

Using the constant test functions in (4.16), we can write $\mathbf{J}_i$ as the function of $\{\Lambda_i\}$. That is for $i = 1, \ldots, N$,

$$\mathbf{J}_i = \frac{\alpha}{h_i}\left(B\left(\frac{\boldsymbol{\beta} h_i}{\alpha}\right)\Lambda_{i-1} - B\left(-\frac{\boldsymbol{\beta} h_i}{\alpha}\right)\Lambda_i\right), \tag{4.27}$$

where $B(.)$ is the Bernoulli function:

$$B(t) := \frac{t}{e^t - 1}.$$

The above flux is usually referred to as the Scherfetter-Gummel flux. For the second equation in (4.16), we write for $i = 1, \ldots, N$,

$$U_i = \frac{e_i \Lambda_{i-1} + e_{-i}\Lambda_i}{e_h + e_{-h}} + \frac{F_i}{\tau_i(e_i + e_{-i})} \tag{4.28}$$

with $e_{\pm i} := \exp(\pm\frac{\boldsymbol{\beta} h_i}{2\alpha})$ and $F_i = \int_{\mathbf{x}_{i-1}}^{\mathbf{x}_i} f\mu$. In terms of the discrete transmission condition, here we simply assume that the stabilization parameter τ is a positive constant on $\mathcal{E}$. Under the piecewise constant setting, we write for $i = 1, \ldots, N-1$,

$$\mathbf{J}_i - \mathbf{J}_{i+1} + \tau(U_i + U_{i+1} - 2\Lambda_i) = 0. \tag{4.29}$$

Inserting (4.27) and (4.28) into (4.29) and letting τ tend to zero to write the global problem for Λ by

$$\begin{aligned} &\frac{\alpha}{h_i} B\left(\frac{\boldsymbol{\beta} h_i}{\alpha}\right)\Lambda_{i-1} \\ &\quad - \left(\frac{\alpha}{h_i} B\left(-\frac{\boldsymbol{\beta} h_i}{\alpha}\right) + \frac{\alpha}{h_{i+1}} B\left(\frac{\boldsymbol{\beta} h_{i+1}}{\alpha}\right)\right)\Lambda_i \\ &\quad + \frac{\alpha}{h_{i+1}} B\left(-\frac{\boldsymbol{\beta} h_{i+1}}{\alpha}\right)\Lambda_{i+1} = F_i. \end{aligned} \tag{4.30}$$

The above system is identical to the system built using the finite volume method together with the Scherfetter-Gummel flux, justifying the interpretation of the W-HDG method as a high order generalization of the Scherfetter-Gummel stabilization.

4.5 Postprocessing

We propose two postprocessing procedures in an element-by-element fashion to obtain the density approximations of u using $\mathcal{P}_{k+1}$ elements so that these approximations converges with order of $O(h^{k+2})$ in both $L^2(\Omega)$ and $L^\infty(\Omega)$ norms.

4.5.1 Postprocessing by $L^2(K)$ minimization

We consider a density approximation $U_* \in \mathcal{P}_{k+1}(\mathcal{T})$ so that for each $K \in \mathcal{T}$, U_* minimize the functional

$$\int_K |\alpha \nabla U_* - \beta U + \mathbf{J}|^2 \, \mathrm{d}\mathbf{x}$$

under the constraint $\int_K (U_* - U)\, \mathrm{d}\mathbf{x} = 0$. We note that the constraint is necessary since $U_* + C$ is also a minimizer of the above functional with C being any constant. The approximation U_* can be obtained by solving the following local problem:

$$\begin{aligned} (1, U_*)_K &= (1, U)_K, \\ (\alpha \nabla U_*, \nabla W) &= (\boldsymbol{\beta} U - \mathbf{J}, \nabla W), \quad \forall W \in \mathcal{P}_{k+1}(K). \end{aligned} \tag{4.31}$$

We note that U_* has a superconvergence property with rate $O(h^{k+2})$ in the $L^2(\Omega)$-sense since both U and $\mathbf{J}$ converge in optimal rate $O(h^{k+1})$ and the local average of U, i.e., $\frac{1}{|K|}\int_K U$ super converges at rate $O(h^{k+2})$ (cf. [18]). The numerical simulation provided in Section 4.6 also suggests that if u is smooth enough, the postprocess approximation also converges in $L^\infty(\Omega)$ with rate $O(h^{k+2})$. We also need to point out that this postprocessing results does not satisfy the local problem (4.8) in any sense.

4.5.2 Postprocessing based on the local problem (4.8)

We next construct an postprocessing approximation based on the local problem (4.8) following the idea from [38]. The procedure needs more computations compared to the $L^2(K)$-minimization.

Reconstruction for the flux

We shall first reconstruct the flux approximation $\mathbf{J}$ in the space $H(\mathrm{div}, \Omega)$, the vector space in $[L^2(\Omega)]^d$ so that its divergence belongs to $L^2(\Omega)$. To achieve this, we consider the piecewise Raviart-Thomas-Nédélec space $\mathcal{RTN}_k(\mathcal{T}) = \{\mathbf{Q} \in [L^2(\Omega)]^d : \mathbf{Q}|_K \in \mathcal{RTN}_k(K)\}$, where $\mathcal{RTN}_k(K) := [\mathcal{P}_k(K)]^d + \mathbf{x}\mathcal{P}(K)$. Using the cell approximation $\mathbf{J}$ as well as the trace approximation $\widehat{\mathbf{J}}$, we define the reconstruction $\mathbf{J}_{div} \in \mathcal{RTN}_k(\mathcal{T})$ as

$$\begin{aligned} \langle \mathbf{J}_{div} \cdot \mathbf{n}, \mathbf{Q} \rangle_e &= \langle \mathbf{J} \cdot \mathbf{n}, \xi \rangle_{\partial K}, \quad \forall \xi \in \mathcal{P}_k(e), \\ (\mathbf{J}_{div}, \mathbf{Q})_K &= (\widehat{\mathbf{J}}, \mathbf{Q})_K, \qquad \forall \mathbf{Q} \in [\mathcal{P}_{k-1}(K)]^d \text{ if } k \geq 1. \end{aligned} \tag{4.32}$$

Here we have to point out some properties of $\mathbf{J}_{div}$: (1) since $\widehat{\mathbf{J}}$ is single valued on $\mathcal{E}$, the normal component of $\mathbf{J}_{div}$ is continuous across interior faces. So $\mathbf{J}_{div} \in H(\mathrm{div}, \Omega)$; (2) [16] shows that both $\mathbf{J}_{div}$ and $\nabla \cdot \mathbf{J}_{div}$ converge in optimal rate $O(h^{k+1})$, which is crucial to the postprocessing of u in the next step.

Postprocessing of U

Our reconstruction of U is based on the problem (4.1) defined on each element K together with a Neumann boundary condition, i.e., at the continuous level, we want to find the pair $(\mathbf{j}^*, u^*)$ satisfying

$$\begin{aligned} \mathbf{j}^* + \alpha\nabla u^* - \boldsymbol{\beta} u^* &= 0, && \text{in } K,\\ \nabla\cdot\mathbf{j}^* &= \nabla\cdot\mathbf{J}_{div}, && \text{in } K,\\ \mathbf{j}^*\cdot\mathbf{n} &= \mathbf{J}_{div}\cdot\mathbf{n}, && \text{on } \partial K. \end{aligned}$$

Similar to the $L^2(K)$ minimization scheme, we also need the constraint $\int_K (u^* - U) = 0$ to guarantee that the local reconstruction is unique.

Now we are ready to define an approximation of $(\mathbf{j}^*, u^*)$ by modifying the discrete local problem (4.8): letting $\boldsymbol{\mathcal{Q}}^*(K)$, $W^*(K)$ and $\mathcal{M}^*(\partial K)$ be the polynomial spaces like $\boldsymbol{\mathcal{Q}}(K)$, $W(K)$ and $\mathcal{M}(\partial K)$ but with the degree $k+1$, we want to find $(\mathbf{J}^*, U^*, \widehat{U}^*) \in \boldsymbol{\mathcal{Q}}^*(K)\times W^*(K)\times\mathcal{M}^*(\partial K)$ such that $\int_K (U^* - U) = 0$ and for all $\mathbf{Q} \in \boldsymbol{\mathcal{Q}}^*(K)$, $W \in \mathcal{W}^*(K)$ and $\xi \in \mathcal{M}^*(K)$,

$$\begin{aligned} (\mathbf{J}^*, \mathbf{Q})_{K,\mu_K} - (\alpha U^*, \nabla\cdot\mathbf{Q})_{K,\mu_K} + \langle \alpha\widehat{U}^*, \mathbf{Q}\cdot\mathbf{n}\rangle_{\partial K,\mu_K} &= 0,\\ -(\mathbf{J}^*, \nabla W - \tfrac{\boldsymbol{\beta}}{\alpha} W)_{K,\mu_K} + \langle \widehat{\mathbf{J}}^*\cdot\mathbf{n}, W\rangle_{\partial K,\mu_K} &= (\nabla\cdot\mathbf{J}_{div}, W)_{K,\mu_K},\\ \langle \widehat{\mathbf{J}}^*\cdot\mathbf{n}, \xi\rangle_{\partial K} &= \langle \mathbf{J}_{div}\cdot\mathbf{n}, \xi\rangle_{\partial K}, \end{aligned} \tag{4.33}$$

with the numerical flux $\widehat{\mathbf{J}}^*$ given by

$$\widehat{\mathbf{J}}^* = \mathbf{J}^* + \tau_K(U^* - \widehat{U}^*)\mathbf{n}.$$

Proposition 12 shows that the above problem admits a unique solution. Thanks to the $O(h^{k+1})$ rate of convergence for both $\mathbf{J}_{div}$ and $\nabla\cdot\mathbf{J}_{div}$, and to the superconvergence property for the local average of U, in [15, 18] the authors show that, for $k \geq 1$, the postprocessing approximation U^* converges with rate $O(h^{k+2})$ in the $L^2(\Omega)$ sense. This convergence behavior is confirmed numerically also for W-HDG in Section 4.6.

4.6 Numerical illustration

In this section, we will present some numerical experiments to verify the stability and convergence of our proposed numerical scheme. Before presenting the examples, we introduce a Gaussian quadrature rule to compute the inner-products $(v, w)_{K,\mu_K}$ and $\langle v, w\rangle_{\partial K,\mu_K}$ exactly when the element $K = \Pi_{i=1}^d (a_i, b_i)$ is a hyperrectangle. The finite element space are defined using bi-polynomials, and $\boldsymbol{\beta}$ is a piecewise constant function. The weight function μ is defined in accordance with (4.10) with $\mathbf{x}_K$ denoting the center of $K \in \mathcal{T}$.

4.6.1 Quadrature schemes on rectangle elements with constant $\boldsymbol{\beta}$

We first consider the one-dimensional case $(v,w)_{\widehat{K},\mu}$ where $\widehat{K} = (0,1)$ and $\mu(\hat{x}) = \exp(-\hat{\beta}(\hat{x} - \hat{x}_{\widehat{K}}))$ for some constant $\hat{\beta}$. We generate a Gaussian quadrature scheme $\widehat{Q}(\widehat{\beta}, \widehat{K})$ based on orthogonal polynomials up to degree k_0 with respect to the inner-product $(v,w)_{\widehat{K},\mu}$. Our integral computation is exact when vw is a polynomial with degree at most $2k_0 - 1$. For any interval $K = (a,b)$, we can compute the inner-product by the change of variable $x = a + h_K\hat{x}$ with $h_K = b - a$. Whence,

$$\int_K e^{-\beta(x-x_K)} v(x)w(x)\,\mathrm{d}x = h_K \int_{\widehat{K}} e^{-\beta h_K(\hat{x}-\hat{x}_{\widehat{K}})}\hat{u}\hat{v}\,\mathrm{d}\hat{x}.$$

The integral on the right hand side is computed exactly by the quadrature $\widehat{Q}(h_K\beta, \widehat{K})$. For the multidimensional case with a constant $\boldsymbol{\beta}$, we apply the tensor product quadrature rule to compute $(v,w)_{K,\mu}$ based on the 1d quadrature scheme. Denoting the corresponding quadrature by $Q_d(\boldsymbol{\beta}, K)$ and given a face e of the element K, we note that we can compute similarly $(v,w)_{e,\mu}$ by $Q_{d-1}(\boldsymbol{\beta}, e)\exp(-\boldsymbol{\beta}\cdot(\mathbf{x}_e - \mathbf{x}_K))$, with $\mathbf{x}_e$ denoting the center of the face.

4.6.2 2D convergence tests on uniform Cartesian grids

We test our proposed numerical scheme (4.16) to approximate the problem (4.1) on the unit square $\Omega = (0,1)^2$. Given a constant vector $\boldsymbol{\beta} = (\beta_1, \beta_2)^{\mathtt{T}}$, it is easy to verify that

$$u(x_1,x_2) \stackrel{\text{def}}{=} x_1x_2\frac{(1-\exp((x_1-1)\beta_1))(1-\exp((x_2-1)\beta_2))}{(1-\exp(-\beta_1))(1-\exp(-\beta_2))}$$

is an analytic solution of (4.1) that vanishes at the boundary and produces boundary layers along $x_1 = 1$ and $x_2 = 1$ when β_1 and β_2 are large enough, respectively.

Convergence for $(\mathbf{J}, U)$

Our numerical implementation relies on the `deal.II` finite element library [4] using bi-polynomial elements $\mathcal{Q}_k$ under a sequence of quadrilateral subdivisions. Here we set $\{\mathcal{T}_j\}_{j=1}^6$ to be sequence of uniform Cartesian grids with the mesh size $h_j = 2^{-j-1}\sqrt{2}$. In Table 4.1, we report the $L^2(\Omega)$-errors between $(\mathbf{j}, u)$ and $(\mathbf{J}, U)$ as well as the $L^\infty(\Omega)$ errors for U using different degrees of bi-polynomials. We note that the stabilization parameter τ is fixed to be 1 on all edges. The convergence rate showing in the table is computed with

$$\text{rate}_{j+1} := \log(e_{j+1}/e_j)/\log(\#\text{DoF}_{j+1}/\#\text{DoF}_j), \tag{4.34}$$

From Table 4.1, we observed optimal rate of convergence $O(h^{k+1})$ in both $L^2(\Omega)$ and $L^\infty(\Omega)$ norms. Here we note that the $L^\infty(\Omega)$-error is approximated by comparing the values on the 6th-order Gaussian quadrature points on each cell. We also report the error behavior of the cell-wise average of U. That is $\max_{K\in\mathcal{T}}(1, u-U)_K$. We observe

deg	# cells	# DoFs	$\|\mathbf{j}-\mathbf{J}\|_{L^2(\Omega)}$		$\|u-U\|_{L^2(\Omega)}$		$\|u-U\|_{L^\infty(\Omega)}$		avg	
			error	rate	error	rate	error	rate	error	rate
0	16	48	8.186e+00	-	2.583e-01	-	6.846e-01	-	4.024e-01	-
	64	192	5.679e+00	0.53	3.371e-01	-0.38	1.035e+00	-0.60	7.088e-01	-0.82
	256	768	3.402e+00	0.74	2.763e-01	0.29	8.435e-01	0.30	6.070e-01	0.22
	1024	3072	2.115e+00	0.69	1.653e-01	0.74	5.761e-01	0.55	3.961e-01	0.62
	4096	12288	1.363e+00	0.63	8.714e-02	0.92	3.479e-01	0.73	2.414e-01	0.71
	16384	49152	8.366e-01	0.70	4.436e-02	0.97	1.965e-01	0.82	1.384e-01	0.80
	65536	196608	4.775e-01	0.81	2.237e-02	0.99	1.053e-01	0.90	7.503e-02	0.88
1	16	192	2.218e+00	-	9.313e-02	-	5.659e-01	-	1.706e-01	-
	64	768	1.107e+00	1.00	4.298e-02	1.12	2.630e-01	1.11	8.488e-02	1.01
	256	3072	3.637e-01	1.61	1.403e-02	1.62	1.163e-01	1.18	1.860e-02	2.19
	1024	12288	9.950e-02	1.87	3.842e-03	1.87	3.887e-02	1.58	3.460e-03	2.43
	4096	49152	2.557e-02	1.96	9.880e-04	1.96	1.120e-02	1.80	5.357e-04	2.69
	16384	196608	6.448e-03	1.99	2.493e-04	1.99	2.999e-03	1.90	7.450e-05	2.85
	65536	786432	1.617e-03	2.00	6.252e-05	2.00	7.756e-04	1.95	9.821e-06	2.92
2	16	432	1.419e+00	-	5.574e-02	-	2.518e-01	-	7.054e-02	-
	64	1728	3.199e-01	2.15	1.234e-02	2.18	5.277e-02	2.25	9.511e-03	2.89
	256	6912	4.996e-02	2.68	1.929e-03	2.68	8.822e-03	2.58	5.021e-04	4.24
	1024	27648	6.630e-03	2.91	2.562e-04	2.91	1.239e-03	2.83	2.195e-05	4.52
	4096	110592	8.354e-04	2.99	3.229e-05	2.99	1.765e-04	2.81	8.334e-07	4.72
	16384	442368	1.042e-04	3.00	4.028e-06	3.00	2.395e-05	2.88	2.874e-08	4.86
	65536	1769472	1.299e-05	3.00	5.022e-07	3.00	3.119e-06	2.94	9.433e-10	4.93

Table 4.1: Errors and observed convergence rates between the fluxes $\mathbf{j}$ and $\mathbf{J}$ in $L^2(\Omega)$ norm and between the densities u and U in both $L^2(\Omega)$ and $L^\infty(\Omega)$ norms using bi-polynomials $\mathcal{Q}_k$ with $k = 0, 1, 2$ under a sequence of uniform quadrilateral subdivisions. Optimal rates of convergences are observed. The last column shows the error decay for the cell-wise average of U, i.e., avg $:= \max_{K\in\mathcal{T}} |(u-U, 1)_K|$.

that this quantity superconverges with rate $O(h^{2k+1})$ when $k \geq 1$. This well help us the obtain higher order rate of convergence in the procedure of the postprocessing using bi-polynomials $\mathcal{Q}_{k+1}$.

Convergence for the local postprocessing

We next examine the convergence of the local postprocessing U_* according to (4.31), i.e. reconstruction by the local L^2 minimization. Table 4.2 reports the error decays for U_* in both $L^2(\Omega)$ and $L^\infty(\Omega)$ norms. Both errors decay with rate $O(h^{k+2})$ for $k \geq 1$.

deg	# cells	# dofs	$\|u - U_*\|_{L^2(\Omega)}$		$\|u - U_*\|_{L^\infty(\Omega)}$	
			error	rate	error	rate
1	16	192	7.868e-02	-	5.873e-01	-
	64	768	2.309e-02	1.77	2.566e-01	1.19
	256	3072	4.557e-03	2.34	6.750e-02	1.93
	1024	12288	7.049e-04	2.69	1.255e-02	2.43
	4096	49152	9.673e-05	2.87	2.092e-03	2.58
	16384	196608	1.260e-05	2.94	3.007e-04	2.80
	65536	786432	1.606e-06	2.97	4.035e-05	2.90
2	16	432	3.211e-02	-	2.485e-01	-
	64	1728	3.985e-03	3.01	5.269e-02	2.24
	256	6912	3.647e-04	3.45	7.371e-03	2.84
	1024	27648	2.624e-05	3.80	5.929e-04	3.64
	4096	110592	1.712e-06	3.94	3.964e-05	3.90
	16384	442368	1.085e-07	3.98	2.511e-06	3.98
	65536	1769472	6.813e-09	3.99	1.574e-07	4.00

Table 4.2: Error between the postprocess solution U_* (L^2-minimization) defined by (4.31) and the exact solution u in both $L^2(\Omega)$ and $L^\infty(\Omega)$ norms using bi-polynomials with degree 1 and 2. Optimal rates of convergence are observed.

Table 4.3 provides the convergence results for the flux reconstruction $\mathbf{J}_{div}$ defined by (4.32) as well as the local postprocessing U^* based on $\mathbf{J}_{div}$; see (4.33). We confirm numerically that both $\mathbf{J}_{div}$ and $\nabla \cdot \mathbf{J}_{div}$ converge with rate $O(h^{k+1})$. Such results, together with the superconvergence of the cell-wise average of U (see the last column of Table 4.1), guarantee that $\|u - U^*\|_{L^2(\Omega)}$ decays with rate $O(h^{k+2})$ and that the errors are greater than $\|u - U_*\|_{L^2(\Omega)}$. We also observe optimal convergence of U^* in $L^\infty(\Omega)$, with an error larger than the $L^\infty(\Omega)$ error for U_*.

Figure 4.1 reports the approximate solution U and its postprocessing results U_* and U^* on the uniform grid $\mathcal{T}_6$ (16384 cells). We observe that the error of U mainly concentrates on the two boundary layer but not on the top right corner of the domain. On the other hand, both U_* and U^* have large errors around the top-right corner. Meanwhile, cells near the two boundary layers contains both positive and negative errors, which behave differently compared to the error of U.

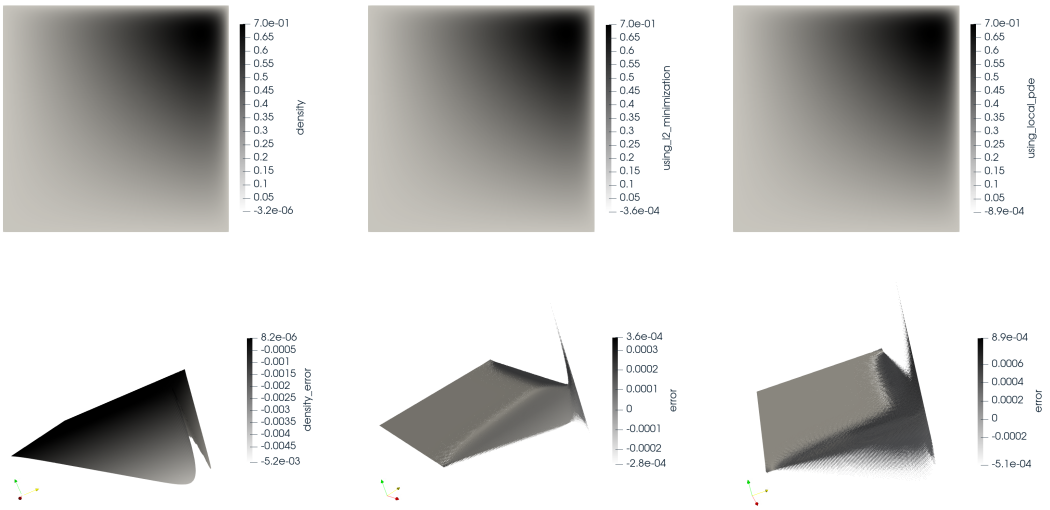


Figure 4.1: From left to right: approximate solutions U, U_* and U^* (top) defined on the grid $\mathcal{T}_6$ using bi-linear elements and the corresponding error plots (bottom). The range of the plots are $[-5.2e-3, 8.2e-6]$, $[-2.8e-4, 3.6e-4]$ and $[-5.1e-4, 8.9e-4]$. Here we rescaled magnitudes with factors 100, 1500, and 1000 respectively.

deg	# cells	# DoFs	$\|\mathbf{j}-\mathbf{J}_{div}\|_{L^2(\Omega)}$		$\|\nabla\cdot\mathbf{j}-\nabla\cdot\mathbf{J}_{div}\|_{L^2(\Omega)}$		$\|u-U^*\|_{L^2(\Omega)}$		$\|u-U^*\|_{L^\infty(\Omega)}$	
			error	rate	error	rate	error	rate	error	rate
1	16	192	2.259e+00	-	4.4415e+00	-	8.8866e-02	-	6.1846e-01	-
	64	768	1.108e+00	1.03	2.2154e+00	1.00	3.0418e-02	1.55	3.8536e-01	0.68
	256	3072	3.635e-01	1.61	6.9941e-01	1.66	6.2003e-03	2.29	1.6328e-01	1.24
	1024	12288	9.942e-02	1.87	1.8539e-01	1.92	9.2468e-04	2.75	3.3066e-02	2.30
	4096	49152	2.555e-02	1.96	4.6504e-02	2.00	1.2232e-04	2.92	5.0770e-03	2.70
	16384	196608	6.443e-03	1.99	1.1557e-02	2.01	1.5584e-05	2.97	6.9794e-04	2.86
	65536	786432	1.616e-03	2.00	2.8750e-03	2.01	1.9622e-06	2.99	9.1424e-05	2.93
2	16	432	1.419e+00	-	3.5109e+00	-	4.6873e-02	-	3.9884e-01	-
	64	1728	3.197e-01	2.15	8.2793e-01	2.08	7.1029e-03	2.72	2.3068e-01	0.79
	256	6912	4.992e-02	2.68	1.3070e-01	2.66	5.9918e-04	3.57	2.7253e-02	3.08
	1024	27648	6.624e-03	2.91	1.7086e-02	2.94	4.0934e-05	3.87	2.1080e-03	3.69
	4096	110592	8.347e-04	2.99	2.1204e-03	3.01	2.6252e-06	3.96	1.4343e-04	3.88
	16384	442368	1.041e-04	3.00	2.6186e-04	3.02	1.6532e-07	3.99	9.3157e-06	3.94
	65536	1769472	1.298e-05	3.00	3.2583e-05	3.01	1.1081e-08	3.90	5.9344e-07	3.97

Table 4.3: $L^2(\Omega)$-error decay for the flux reconstruction $\mathbf{J}_{div}$ (defined by (4.32)) as well as its divergence. The $L^2(\Omega)$ and $L^\infty(\Omega)$ errors for the postprocessing U^* defined by (4.33) (based on flux reconstruction) are report in this table as well.

4.6.3 A one dimensional benchmark test

In section 4.4.8, we have pointed out that the Scharfetter–Gummel scheme (4.30) can be obtained as a certain limit of our proposed method. Here we perform a comparison test using these two numerical approaches as well as the standard HDG scheme to solve the stationary van Roosbroeck model in a one-dimensional domain $\Omega = [0, \ell]$.

The van Roosbroeck system

The van Roosbroeck system is one of the most common models used to describe currents of electric charge carriers inside a semiconductor device. It consists of three nonlinear differential equations for the unknown electrostatic potential $\psi(x)$, the density of electrons n and the density of holes p. A complete description of this model can be found in [35]. In the following numerical test, we compute the so-called thermodynamic equilibrium, a physical state defined by vanishing currents. In equilibrium, it is possible to decouple the above mentioned three equations, i.e., solving a nonlinear Poisson equation to obtain $\psi(x)$ and using this function to compute the other two unknown densities (n, p).

Here the one dimensional Poisson equation for $\psi(x)$ is given by

$$-\frac{\mathrm{d}}{\mathrm{d}x}\left(\varepsilon_s \frac{\mathrm{d}}{\mathrm{d}x}\psi\right) = q\left(N_v \exp\left(\frac{E_v - q\psi}{k_B T}\right) - N_c \exp\left(\frac{q\psi - E_c}{k_B T}\right) + C\right), \tag{4.35}$$

where q is the elementary charge, k_B is the Boltzmann constant, T represents the temperature of the device (which we will suppose to be constant on all Ω), and ε_s, N_v, E_v, N_c, E_c, and C are constants or piecewise constant functions that describe some physical properties of the material; we refer to Table 4.4 for all these values. Dirichlet boundary conditions for (4.35) can be obtained under the assumption that the right hand side datum vanishes at the boundary (cf. [35, Section 2.3]).

Given the electrostatic potential ψ, we shall solve n and p governed by a drift-diffusion system. Here we can solve them separately in thermal equilibrium and we only focus on the hole density. For the hole density, there holds

$$\begin{cases} -q\dfrac{\mathrm{d}}{\mathrm{d}x}\left(\mu_p\left(U_T\dfrac{\mathrm{d}p}{\mathrm{d}x} + p\dfrac{\mathrm{d}\psi}{\mathrm{d}x}\right)\right) = 0, \\ p(x_0) = N_v \exp\left(\dfrac{E_v - q\psi\,(x_0)}{k_B T}\right) \qquad \text{for } x_0 \in \{0, \ell\}. \end{cases} \tag{4.36}$$

where $U_T := \frac{k_B T}{q}$ and the mobility constant μ_p defines how easy is for the holes to travel through the device. Comparing the above equation with (4.1), we have

$$\alpha := U_T, \qquad \beta := -\frac{\mathrm{d}\psi}{\mathrm{d}x}, \qquad f := 0.$$

Numerical settings

We will test our numerical method for (4.36) using the electrostatic potential given by (4.35). Here we consider our physical domain as a p-i-n device, which contains a p- and

an n- doped regions as well as an intrinsic layer in between. The doping concentration C in (4.35) is assumed to be a piecewise constant function

$$C = \begin{cases} N_D, & \text{if } 0 \leq x < \frac{\ell}{3}, \\ 0, & \text{if } \frac{\ell}{3} \leq x < \frac{2\ell}{3}, \\ -N_A, & \text{if } x \geq \frac{2\ell}{3}, \end{cases} \tag{4.37}$$

where N_D and N_A are positive constants defined in Table 4.4. A graphical representation of the solutions to this problem is shown in Figure 4.2.

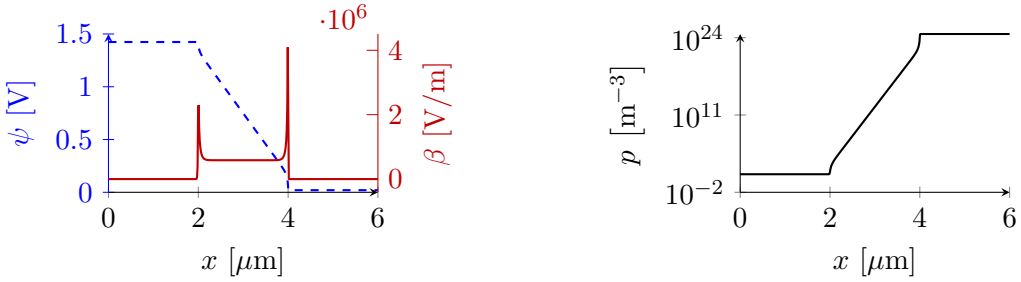


Figure 4.2: (Left) electrostatic potential ψ (dashed blue line) for (4.35) and the electric field $\beta = -\mathrm{d}\psi/\mathrm{d}x$ (solid red line) for a model that uses the constants described in Table 4.4. (Right) the hole concentration for (4.36). Here we obtain the results using linear elements on a highly refined grid with 18432 cells and 92161 degrees of freedom (counting the degrees of freedom for the solution, its gradient and the solution on the trace).

We construct a sequence of hierarchical grids $\{\mathcal{T}_i\}_i$ of Ω for i in $\{1, \ldots, 7\}$. The coarsest grid $\mathcal{T}_1$ is a refinement of the uniform partition of Ω with intervals $I_j := (\frac{(j-1)\ell}{6}, \frac{j\ell}{6})$ for $j = 1, \ldots, 6$, such that the two junctions $\{\frac{\ell}{3}, \frac{2\ell}{3}\}$ are grid points of $\mathcal{T}_i$. Then we build $\mathcal{T}_1$ as follows:

1. We set $I_1, I_6 \in \mathcal{T}_1$.
2. Bisect the rest of the intervals with the following graded strategy: for each I_j with $j = 2, 3, 4, 5$, we consider an affine transformation $\mathcal{F}_j : I_j \to (0, 1)$ such the junction point is at 0. Then we subdivide $(0, 1)$ uniformly with four intervals and rescale them by taking the square of the position, i.e., for $i = 1$ the partition is given by $\mathcal{P} := \{(\frac{(k-1)^2}{2^{i+2}}, \frac{k^2}{2^{i+2}})\}_{k=1}^{2^{i+1}}$.
3. $\mathcal{T}_1$ consists of I_1, I_6, and $\mathcal{F}_j^{-1}(\mathcal{P})$ for $j = 2, 3, 4, 5$.

To generate $\mathcal{T}_i$ for $i > 1$, we refine globally $\{I_1, I_6\}$ with i times, and refine the other I_j intervals following Step 2. Examples of the grids in $\{\mathcal{T}_i\}_{i=1}^3$ are shown in Figure 4.3.

We approximate the solutions to (4.36) on $\mathcal{T}_j$ with three different methods: the Scharfetter–Gummel scheme (4.30), the standard HDG method (H-LDG) [17] and our

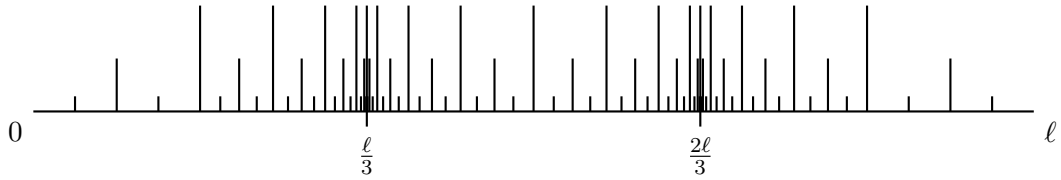


Figure 4.3: The triangulation $\mathcal{T}_j$ over Ω with 18, 36 and 72 cells.

proposed one in (4.16). Denote the approximation of the hole density with P_j. We obtain a numerical reference solution P_r by solving the problem with the standard HDG method on a grid $\mathcal{T}_r$ which is globally refined from $\mathcal{T}_7$ four times. The coefficient $\beta = -\frac{\mathrm{d}\psi}{\mathrm{d}x}$ is generated by solving the Poisson problem (4.35) on $\mathcal{T}_r$ with the standard HDG and project (in the L^2 sense) the resulting β onto $\boldsymbol{\mathcal{Q}}(\mathcal{T})$ using piecewise constant functions.

Results

In the top-left plot of Figure 4.4, we report the errors $\|P_j - P_r\|_{L^2(\Omega)}$ versus the numbers of degrees of freedom, where, to maintain the comparison fair with the finite volume scheme, we use $\mathcal{P}_0$ functions for the HDG and W-HDG approaches. It turns out that all three numerical methods converge to first order, however, the numerical solution computed using the standard HDG violates the maximum principle; see the top-right plot for the profile of P_4. On the other hand, the numerical results using (4.16) behaves similarly to the state-of-the-art solution obtained with Scharfetter-Gummel stabilized finite volumes.

It is common to postprocess the solution generated by the Scharfetter-Gummel stabilized finite volume scheme by connecting the mid-points of all cells with straight lines, obtaining a piecewise linear and continuous solution. Therefore, in the bottom part of Figure 4.4, we perform an analogous comparison after implementing this strategy for all three methods. In particular, for the HDG and W-HDG scheme, this algorithm has been implemented not by connecting the mid-points of the cells but the points on the trace obtained from $\hat{u}$. The bottom-left logarithmic plot shows that the resulting linear approximations, denoted by P_j^*, converge to P_r with second order in all cases. On the other side, the quality of the solution on the points where the charge carrier density is low (i.e., between 0 and $\frac{2}{3}\ell$) is quite different between the FVM and W-HDG method and the standard HDG; indeed, the contribute of these regions on the global L^2 error is negligible. On the bottom-right plot we can see the profile for P_3^*. The approximations from the FVM method and the W-HDG both satisfy the discrete maximum principle; instead, the result for the standard HDG method suffers the fact that some values of $\hat{u}$ are negative (up to $-2.284 \times 10^{13}\mathrm{m}^{-3}$); therefore, here we only show the part with values above $10^{-2}\mathrm{m}^{-3}$.

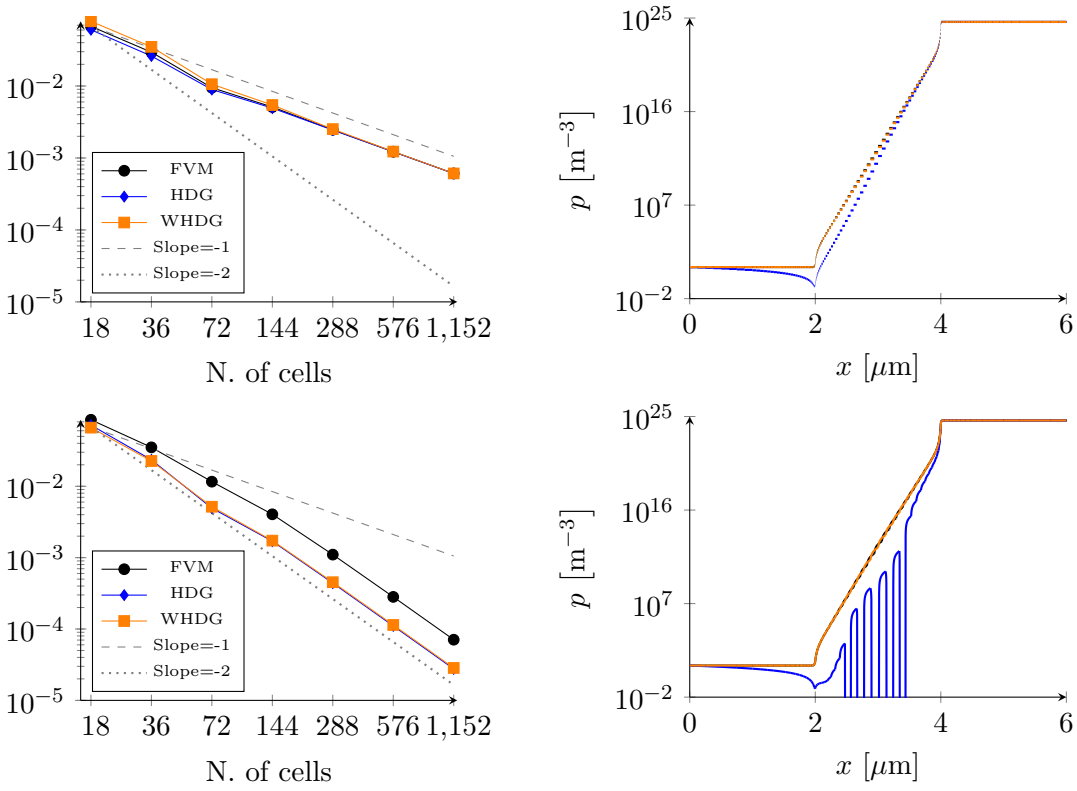


Figure 4.4: (Top-left) $L^2(\Omega)$-errors between P_j and P_r against the number cells using different numerical approaches. (Top-right) Numerical approximation of P_4. (Bottom-left) post-processing P_j^* of P_j using linear elements. (Bottom-right) Post-processing results for P_3^*. Here we have truncated the results P_3^* from the standard HDG for the values above 10^{-2}m^{-3}.

Physical Quantity	Symbol	Value	Units
Device length	ℓ	6.00	µm
Reference temperature	T	300.00	K
Absolute permittivity	ε_s	1.1422×10^{-10}	F m^{-1}
Valence band density	N_v	9.1396×10^{24}	m^{-3}
Valence band-edge energy	E_v	0	eV
Conduction band density	N_c	4.3520×10^{23}	m^{-3}
Conduction band-edge energy	E_c	1.4240	eV
Hole mobility	μ_p	4.0000×10^{-2}	$\text{m}^2\,\text{V}^{-1}\,\text{s}^{-1}$
Acceptor doping density	N_A	4.2042×10^{24}	m^{-3}
Donator doping density	N_D	4.3520×10^{23}	m^{-3}
Elementary charge	q	1.6022×10^{-19}	C
Boltzmann constant	k_B	1.3806×10^{-23}	J K^{-1}

Table 4.4: The physical constants used in Section 4.6.3.

Conclusions

This thesis presents three main contributions:

- a data–driven solution for non-destructive doping reconstruction in semiconductors [arXiv:2208.00742];
- the proof of equivalence of HDG methods and SG-FVM methods in one dimension [arXiv:2211.01648];
- the presentation of a new weighted class of HDG methods (W-HDG) for the solution of convection dominated problems.

These results are currently being evaluated for publications, and are available as preprints. They have been presented in the NUSOD conference of September 2022 with the talk "Data-driven doping reconstruction" and in the PRIMO workshop 2022 (September 08, 2022) with the talk "Solving inverse problems for doping reconstruction in semiconductors via machine learning".

The datasets that were generated for the publication [arXiv:2208.00742] are available at [41]. The generation of the dataset was possible thanks to computational hours on the CINECA cluster Marconi, and on the computational machines of University of Trieste, that we would like to gratefully acknowledge.

The ideas of the publication [arXiv:2208.00742] were developed during two visits in Berlin (from March 23, 2022 to April 08, 2022 and from April 25, 2022 to May 06, 2022) in the group of Dr. Farrell, and during a visit of Dr. Farrell to Trieste in the context of the exchange program "professori visitatori" of GNCS which is gratefully acknowledged.

In our study on the LPS problem, we have shown that least square models can offer a good compromise between computational cost and quality of the predictions. On the other hand, neural networks have been able to achieve better results, roughly halving the ℓ^∞ error in each test. It is worth noting that MLP and ResNets produce similar results (even if the MLP model proved to be more robust to noise); nevertheless, the smaller computational effort required to train the MLP model and its smaller number of hyperparameters make it a more promising method in this setup.

Future research may go into different directions. A better understanding of the theoretical problem could reduce the ambiguity we have seen in Figure 2.25. The numerical simulation used for the forward model served as a proof of principle and was limited to a 2D version of the photovoltage problem. For the 3D problem, efficient data generation strategies need to be designed. In such setup (with 2D signals instead of onedimensional ones), ResNets may exhibit a notable advantage over the MLP models.

In Chapter 3 we have studied the relationship between standard HDG methods and Finite Volume methods stabilized by Scharfetter–Gummel scheme. A future challenge

would be to extend this results to nonuniform grids; unfortunately, this is a nontrivial task that would require a much finer tuning of the numerical fluxes.

Finally, in Chapter 4 we presented a weighted HDG method for the solution of convection-diffusion problems. We proved well-posedness property of the method, validated its numerical stability as well as convergence properties, and provide a fair comparison with a state-of-the-art finite volume discretization using Scharfetter and Gummel stabilization, and with standard HDG methods, for a convection dominated problem with dramatic scale changes, derived from the modeling of a p-i-n semiconductor junction.

W-HDG is equivalent to the Scharfetter and Gummel stabilized finite volume for piecewise constant polynomial approximations, and generalizes it to arbitrary high order, while maintaining similar stability properties. An important path of investigation would be the construction of a positivity preserving post-processing solution for W-HDG, which is not guaranteed from the current algorithm.

Bibliography

[1] J. Adler and O. Öktem. Solving ill-posed inverse problems using iterative deep neural networks. *Inverse Problems*, 33(12):124007, 2017.

[2] G. Alì, A. Bartel, M. Günther, and C. Tischendorf. Elliptic partial differential-algebraic multiphysics models in electrical network design. *Mathematical Models and Methods in Applied Sciences*, 13(09):1261–1278, Sept. 2003.

[3] G. Alì and N. Rotundo. An existence result for elliptic partial differential–algebraic equations arising in semiconductor modeling. *Nonlinear Analysis: Theory, Methods & Applications*, 72(12):4666 – 4681, 2010.

[4] D. Arndt, W. Bangerth, B. Blais, M. Fehling, R. Gassmöller, T. Heister, L. Heltai, U. Köcher, M. Kronbichler, M. Maier, P. Munch, J.-P. Pelteret, S. Proell, K. Simon, B. Turcksin, D. Wells, and J. Zhang. The `deal.II` library, version 9.3. *Journal of Numerical Mathematics*, 29(3):171–186, 2021, accepted for publication.

[5] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini. Unified analysis of discontinuous galerkin methods for elliptic problems. *SIAM Journal on Numerical Analysis*, 39(5):1749–1779, Jan. 2002.

[6] S. Arridge, P. Maass, O. Öktem, and C.-B. Schönlieb. Solving inverse problems using data-driven models. *Acta Numerica*, 28:1–174, 2019.

[7] J. Bajaj, L. O. Bubulac, P. R. Newman, W. E. Tennant, and P. M. Raccah. Spatial mapping of electrically active defects in HgCdTe using laser beam-induced current. *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, 5(5):3186–3189, sep 1987.

[8] M. Bessemoulin-Chatard. A finite volume scheme for convection–diffusion equations with nonlinear diffusion derived from the scharfetter–gummel scheme. *Numerische Mathematik*, 121(4):637–670, Feb. 2012.

[9] D. Boffi, F. Brezzi, and M. Fortin. *Mixed Finite Element Methods and Applications*. Springer Berlin Heidelberg, 2013.

[10] F. Brezzi and M. Fortin. *Mixed and hybrid finite element methods*, volume 15. Springer Science & Business Media, 2012.

[11] F. Brezzi, L. D. Marini, and P. Pietra. Two-dimensional exponential fitting and applications to drift-diffusion models. *SIAM Journal on Numerical Analysis*, 26(6):1342–1355, 1989.

[12] M. Burger, H. W. Engl, A. Leitao, and P. A. Markowich. On inverse problems for semiconductor equations. *Milan Journal of Mathematics*, 72(1):273–313, 2004.

[13] S. Busenberg, W. Fang, and K. Ito. Modeling and analysis of laser-beam-induced current images in semiconductors. *SIAM Journal on Applied Mathematics*, 53(1):187–204, 1993.

[14] G. Chen, P. Monk, and Y. Zhang. An hdg method for the time-dependent drift–diffusion model of semiconductor devices. *Journal of Scientific Computing*, 80(1):420–443, 2019.

[15] B. Cockburn, B. Dong, and J. Guzmán. A superconvergent ldg-hybridizable galerkin method for second-order elliptic problems. *Mathematics of Computation*, 77(264):1887–1916, 2008.

[16] B. Cockburn, B. Dong, J. Guzmán, M. Restelli, and R. Sacco. A hybridizable discontinuous galerkin method for steady-state convection-diffusion-reaction problems. *SIAM Journal on Scientific Computing*, 31(5):3827–3846, 2009.

[17] B. Cockburn, J. Gopalakrishnan, and R. Lazarov. Unified hybridization of discontinuous galerkin, mixed, and continuous galerkin methods for second order elliptic problems. *SIAM Journal on Numerical Analysis*, 47(2):1319–1365, 2009.

[18] B. Cockburn, J. Guzmán, and H. Wang. Superconvergent discontinuous galerkin methods for second-order elliptic problems. *Mathematics of Computation*, 78(265):1–24, 2009.

[19] A. Ern and J.-L. Guermond. *Theory and practice of finite elements*, volume 159. Springer, 2004.

[20] P. Farrell, S. Kayser, and N. Rotundo. Modeling and simulation of the lateral photovoltage scanning method. *Computer and Mathematics with Applications*, 102:248–260, 2021.

[21] P. Farrell, N. Rotundo, D. Doan, M. Kantner, J. Fuhrmann, and T. Koprucki. Mathematical methods: Drift-diffusion models. In J. Piprek, editor, *Handbook of Optoelectronic Device Modeling and Simulation*, volume 2, chapter 50, pages 733–771. CRC Press, 2017.

[22] W. Fichtner, D. J. Rose, and R. E. Bank. Semiconductor device simulation. *SIAM Journal on Scientific and Statistical Computing*, 4(3):391–415, Sept. 1983.

[23] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[24] P. Grisvard. *Elliptic problems in nonsmooth domains*. SIAM, 2011.

[25] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[26] L. Jastrzebski, J. Lagowski, G. W. Cullen, and J. I. Pankove. Hydrogenation induced improvement in electronic properties of heteroepitaxial silicon-on-sapphire. *Applied Physics Letters*, 40(8):713–716, apr 1982.

[27] J. Kaipio and E. Somersalo. *Statistical and computational inverse problems*, volume 160. Springer, 2006.

[28] S. Kayser, P. Farrell, and N. Rotundo. Detecting striations via the lateral photovoltage scanning method without screening effect. *Optical and Quantum Electronics*, 53:288, 2021.

[29] A. Leitao, P. Markowich, and J. Zubelli. On inverse doping profile problems for the stationary voltage–current map. *Inverse Problems*, 22(3):1071–1088, 2006.

[30] H. Li, J. Schwab, S. Antholzer, and M. Haltmeier. Nett: Solving inverse problems with deep neural networks. *Inverse Problems*, 36(6):065005, 2020.

[31] L. Li, K. Jamieson, A. Rostamizadeh, E. Gonina, M. Hardt, B. Recht, and A. Talwalkar. A system for massively parallel hyperparameter tuning. *arXiv*, 2018.

[32] Y. Liu and C.-W. Shu. Error analysis of the semi-discrete local discontinuous galerkin method for semiconductor device simulation models. *Science China Mathematics*, 53(12):3255–3278, 2010.

[33] Y. Liu and C.-W. Shu. Analysis of the local discontinuous galerkin method for the drift-diffusion model of semiconductor devices. *Science China Mathematics*, 59(1):115–140, 2016.

[34] A. Lüdge and H. Riemann. Doping Inhomogeneities in Silicon Crystals Detected by the Lateral Photovoltage Scanning (LPS) Method. *Inst. Phys. Conf. Ser.*, 160:145–148, 1997.

[35] P. A. Markowich. *The Stationary Semiconductor Device Equations*. Springer Vienna, 1986.

[36] M. S. Mock. On equations describing steady-state carrier distributions in a semiconductor device. *Communications on Pure and Applied Mathematics*, 25(6):781–792, Nov. 1972.

[37] P. Moritz, R. Nishihara, S. Wang, A. Tumanov, R. Liaw, E. Liang, M. Elibol, Z. Yang, W. Paul, M. I. Jordan, and I. Stoica. Ray: A distributed framework for emerging ai applications, 2017.

[38] N. C. Nguyen, J. Peraire, and B. Cockburn. An implicit high-order hybridizable discontinuous galerkin method for linear convection–diffusion equations. *Journal of Computational Physics*, 228(9):3232–3254, 2009.

[39] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[40] D. Peschka, N. Rotundo, and M. Thomas. Doping optimization for optoelectronic devices. *Optical and Quantum Electronics*, 50(3):125, Feb 2018.

[41] S. Piani, W. Lei, N. Rotundo, P. Farrell, and L. Heltai. Software repository for data-driven reconstruction of doping profiles in semiconductors. `http://dx.doi.org/10.5281/zenodo.7294848`, Sept. 2022.

[42] A. Quarteroni. *Numerical Models for Differential Problems*. Springer International Publishing, 2017.

[43] D. Scharfetter and H. Gummel. Large-signal analysis of a silicon read diode oscillator. *IEEE Transactions on Electron Devices*, 16(1):64–77, 1969.

[44] S. Sze and K. K. Ng. *Physics of Semiconductor Devices*. John Wiley & Sons, Inc., Apr. 2006.

[45] M. E. Taylor. *Partial Differential Equations III*. Springer New York, 2011.

[46] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 9.6)*, 2022. `https://www.sagemath.org`.

[47] W. Van Roosbroeck. Theory of the flow of electrons and holes in germanium and other semiconductors. *Bell System Technical Journal*, 29(4):560–607, 1950.

[48] C. R. Vogel. *Computational methods for inverse problems*. SIAM, 2002.

[49] D. B. Wittry and D. F. Kyser. Measurement of diffusion lengths in direct-gap semiconductors by electron-beam excitation. *Journal of Applied Physics*, 38(1):375–382, 1967.