POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

On the Choice of the Entropy Function and Iterative Solver in Structure-Preserving Discontinuous Galerkin Methods for the Fisher-Kolmogorov-Petrovsky-Piskunov Equation

Tesi di Laurea Magistrale in
Mathematical Engineering - Ingegneria Matematica

Nicola Ciliberti, 10767457

Advisor:
Prof. Paola F. Antonietti

Co-advisors:
Prof. Ilaria Perugia
Dr. Mattia Corti

Academic year:
2025-2026

Abstract: The Fisher-Kolmogorov-Petrovsky-Piskunov equation is a partial differential equation, widely used in modelising phenomena that involve reaction-diffusion mechanisms. One of the possible applications is the modeling neurodegenerative diseases, such as Alzheimer's and Parkinson's disease, which are characterised by the propagation of misfolded proteins trough brain tissue. This thesis describes a structure-preserving Local Discontinuous Galerkin framework for the numerical approximation of the Fisher-Kolmogorov-Petrovsky-Piskunov equation, which relies on an entropy variable reformulation to guarantee that the discrete solution satisfies the suitable bounds at all times. Two aspects of the framework are investigated. First, we analyze the influence of the entropy transformation on the accuracy and robustness of the scheme, testing the Shannon–Boltzmann, Hellinger, and k-Fermi–Dirac functions as alternatives to the standard Fermi–Dirac transformation. Numerical experiments reveal that theoretical admissibility is not sufficient to guarantee practical performance. Indeed, the Hellinger transformation fails systematically on sharp-front solutions due to excessive numerical diffusion, while Shannon–Boltzmann entropic transformation converges with optimal rates but produces larger errors. The k-Fermi–Dirac family illustrates that performance depends jointly on the steepness of the transformation near $w = 0$ and on its asymptotic behavior. Second, we address the sensitivity of Newton's method to the choice of initial guess by proposing and testing a "Warm-Start" iterative approach, which precedes Newton iterations with a small number of Richardson steps and a relaxed transition phase. Numerical tests demonstrate that this method improves robustness significantly in challenging regimes, particularly for large time steps and low polynomial degrees at a modest additional computational cost, while offering no advantage in long-time simulations where error accumulation dominates.

Key-words: Fisher–Kolmogorov equation, structure-preserving methods, Local Discontinuous Galerkin, entropy transformation, iterative solvers, neurodegenerative diseases.

1. Introduction

Neurodegenerative diseases such as Alzeheimer's and Parkinson's are pathological processes that progressively compromise, interfere with and eventually block neuronal function. The mechanism by which both these ill-

nesses develop is very similar: healthy proteins misfold and accumulate inside the brain tissue. Parkinson's disease is associated with α-Synuclein [18] and Alzheimer's disease with the Amyloid-β and τ protein [20]. The tracing of these proteins is still an important matter of research and state-of-the-art methods have typically significant limitations, such as knowing the region of diffusion beforehand or using invasive procedures [12]. The development of mathematical models allows to simulate the diffusion of such proteins is a powerful tool to complement clinical data and investigate disease progression.
One of the most known mathematical models for protein propagation is the Fisher-Kolmogorov-Petrovsky-Piskunov (FKPP) equation, a nonlinear reaction–diffusion model, originally meant to model population dynamics [4], that has become widely used to simulate the propagation of misfolded proteins [16]. The FKPP equation combines diffusive transport with a logistic reaction term. This formulation naturally captures the characteristic traveling wavefront patterns observed experimentally in disease progression [9, 13]. Many different numerical methods have been proposed to compute approximate solutions of the FKPP equation in the context of brain neurodegeneration. Classical approaches include Finite Element Methods [19], Boundary Element Methods [5], and Discontinuous Galerkin methods [2]. However, due to the unstable nature of the healthy equilibrium, numerical solutions are prone to violations of the physical bounds, particularly in regions where the concentration approaches the boundary values, leading to unphysical solutions or numerical instabilities, particularly for solutions with sharp fronts characteristic of disease spreading patterns [21]. To address these challenges, structure-preserving numerical methods have been developed [3, 10].These methods aim to preserve at the discrete level fundamental properties of the continuous problem, such as positivity and boundedness.
Recently, Antonietti et al. [1] introduced a structure-preserving Local Discontinuous Galerkin (LDG) discretization for the FKPP equation based on entropy variable transformations. The key idea is to reformulate the problem in terms of an entropy variable w such that $c = u(w)$, where $u(w)$ is chosen to satisfy specific properties. This approach ensures that the discrete solution preserves the bounds $[0, 1]$ and maintains entropy stability. The LDG framework offers additional advantages including local conservativity, flexibility on general polygonal meshes, high order accuracy and optimal convergence rates, which makes it applicable to a wide range of realistic scenarios. The analysis conducted in [1] showed that it has a further computational advantage with respect to classical methods, such as Interior Penalty Discontinuous Galerkin methods (IPDG, see [17]), since the definition of the nonlinear operators does not contain surface integrals, which avoids performing numerous calculations. Moreover, the method seems to be very accurate, even for very low-order polynomial degrees. On the other hand, this method has some weaknesses. First of all, the added complexity in the code, which is more articulated than the one for classical methods. Secondly, the choice of the entropy function. Theoretically, any choice would be as fine, as long as it fulfills suitable theoretical properties. Numerically speaking, a different entropy function can show a completely different behaviour. Lastly, the choice of the iterative method. The most common choice for these problems is Newton's method, which combines fast convergence with good robustness. However, this approach is affected by a significant limitation, namely the selection of the initial guess. This issue is particularly critical in the presence of ill-conditioned problems, which commonly arise in practical applications. For this reason, several hybrid methods have been tested in order to enhance the robustness of the approach. Overall, the aim of this thesis is to further develop and extend the framework introduced in [1], addressing the aforementioned issues.This thesis is structured as follows. In Section 2 we present the model problem and the methodology behind the structure-preserving LDG scheme. In Section 3 we introduce a collection of possible entropy transformations. The standard choice in [1] is the Fermi-Dirac transformation, in this thesis it is tested over the Shannon-Boltzmann function and the Hellinger function. Both of these functions share the same characteristics as the Fermi-Dirac function. In Section 5 we conduct tests on several alternative hybrid methods with the aim to improve the performances of Newton's method. Finally, Section 6 contains conclusive observations on the discoveries made and proposals for new developments.

2. Equations

Let $\Omega \subset \mathbb{R}^d$ ($d \in \{2, 3\}$) be a polygonal/polyhedral space domain with Lipschitz boundary $\Gamma = \partial\Omega$ and outward-pointing normal unit vector $\mathbf{n}_\Omega$, and let $(0, T)$ be a time interval with $T > 0$. In the space–time cylinder $Q_T = \Omega \times (0, T)$, we consider the following initial and boundary value problem for the Fisher–Kolmogorov equation: find $c : Q_T \to \mathbb{R}$ such that

$$\partial_t c - \nabla \cdot (\mathbf{D}\nabla c) = \alpha\, c(1 - c) =: f(c) \qquad \text{in } Q_T, \tag{1a}$$

$$(\mathbf{D}\nabla c) \cdot \mathbf{n}_\Omega = 0 \qquad \text{on } \Gamma \times (0, T), \tag{1b}$$

$$c(\cdot, 0) = c_0 \qquad \text{in } \Omega. \tag{1c}$$

Here, the coefficient $\alpha = \alpha(\mathbf{x}) \in L^\infty(\Omega)$ is strictly positive, and the diffusion tensor $\mathbf{D} = \mathbf{D}(\mathbf{x}) \in L^\infty(\Omega)^{d\times d}$ is

assumed to be symmetric and uniformly positive definite, namely there exists a constant $D_0 > 0$ such that

$$\mathbf{z}^\top \mathbf{D}\,\mathbf{z} \geq D_0|\mathbf{z}|^2 \qquad \forall\, \mathbf{z} \in \mathbb{R}^d. \tag{2}$$

Additionally, we assume that $0 \leq c_0 \leq 1$ a.e. in Ω, so that

$$0 \leq c(\mathbf{x}, t) \leq 1 \qquad \text{in } Q_T;$$

From now on, given a domain $\mathcal{D}$, we use $(\cdot,\cdot)_{\mathcal{D}}$ to denote the $L^2(\mathcal{D})$ inner product, and by $|\mathcal{D}|$ its measure.

2.1. Entropy variable

The main idea of the entropy-variable formulations leading to structure preserving method, presented in [6, 10] relies on a change of variable

$$c = u(w), \quad w : Q_T \to \mathbb{R}$$

We next give the following definition:

Definition 1. *An entropy function $s : [0,1] \to \mathbb{R}$ is a function such that:*

- $s'(c) = u^{-1}(c)$,
- $s(c) > 0$,
- $s''(c) \geq \alpha_0 > 0$.

We will call entropy quantities:

- *$s(c)$ and its derivatives (up to third degree),*
- *$u(w)$ and its first derivative,*
- *$u^2(w)$ and its first derivative.*

2.2. The entropy formulation and its numerical discretisation

We introduce the following auxiliary variables in Q_T:

$$w \text{ s.t. } \quad c = u(w), \tag{3a}$$
$$\boldsymbol{z} = -\nabla w, \tag{3b}$$
$$Ds''(c)\boldsymbol{\sigma} = -Ds''(c)\nabla c = D\boldsymbol{z}, \tag{3c}$$
$$\boldsymbol{r} = D\boldsymbol{\sigma}. \tag{3d}$$
$$\nabla w = s''(c)\nabla c. \tag{3e}$$

The first identity in (3c) imposes $\boldsymbol{\sigma} = -\nabla c$, due to the invertibility of D and the fact that s'' is bounded away from zero. After applying the change of variables in (3a), the definition of the auxiliary variables follows the standard LDG approach, avoiding the presence of nonlinearities under differential operators. This is achieved by imposing the chain rule in equation (3e). Specifically, we define

$$\boldsymbol{r} = -D\nabla c \quad \text{(see (3d))},$$

writing $\boldsymbol{\sigma} = -\nabla c$ (first identity in (3c)). The second identity in (3c), together with (3b), enforces the chain rule in (3c), allow us to write $\nabla c = \nabla(u(w))$ in the formulation. The variable $\boldsymbol{r}$ thus represents the flux of c and will appear explicitly in the formulation (see (4) below).
The original problem is reformulated in terms of the variables defined in (3) as follows: Find

$$w : Q_T \to \mathbb{R}, \qquad \boldsymbol{r} : Q_T \to \mathbb{R}^d,$$

such that

$$\partial_t c + \nabla \cdot \boldsymbol{r} = \alpha c(1-c) = f(c) \quad \text{in } Q_T, \tag{4a}$$
$$\boldsymbol{r} \cdot \boldsymbol{n}_\Omega = 0 \quad \text{on } \Gamma \times (0,T), \tag{4b}$$
$$c(\cdot, 0) = c_0 \quad \text{in } \Omega \times \{0\}, \tag{4c}$$

where, in $Q_T = \Omega \times (0,T)$, $c = u(w)$.

2.3. Meshes

We partition the spatial domain Ω by a locally quasi-uniform polytopal mesh $\mathcal{T}_h$ with mesh size $h = \max_{K\in\mathcal{T}_h} h_K$, where h_K denotes the diameter of the element K. We also partition the time interval $(0,T)$ by a mesh $\mathcal{T}_\tau$ defined by the points

$$0 = t_0 < t_1 < \cdots < t_N = T.$$

For $n = 1,\dots,N$, we define the time interval

$$I_n = (t_{n-1}, t_n), \qquad \tau_n = t_n - t_{n-1}.$$

As usual in the DG setting, we denote by

$$\mathcal{F}_h = \mathcal{F}_h^I \cup \mathcal{F}_h^{\mathrm{N}},$$

the set of all mesh facets of $\mathcal{T}_h$, where $\mathcal{F}_h^I$ and $\mathcal{F}_h^{\mathrm{N}}$ are the sets of internal and (Neumann) boundary faces, respectively.
Additionally, we define the mesh-size function $\mathbf{h} \in L^\infty(\mathcal{F}_h^I)$ as

$$\mathbf{h}(\boldsymbol{x}) = \eta_F^{-1}\left[\frac{1}{2}\left(\left(\frac{|K_1|}{m_{K_1}|F|}\right)^\theta + \left(\frac{|K_2|}{m_{K_2}|F|}\right)^\theta\right)\right]^{1/\theta}, \tag{5}$$

if $\boldsymbol{x} \in F$ and $F \in \mathcal{F}_h^I$ is shared by $K_1, K_2 \in \mathcal{T}_h$. Here, $\eta_F > 0$ is a parameter independent of the mesh size, which is chosen in (6) below in terms of the diffusion tensor and the polynomial degree in the space discretization, θ is the power-mean exponent, and m_{K^*} denotes the number of facets of the polytopal element K^*.
In the numerical experiments presented, $\theta = -1$ is always used, which corresponds to the harmonic mean.
For each element $K \in \mathcal{T}_h$, let $\boldsymbol{n}_K$ denote the unit outward normal vector to ∂K. Given any piecewise smooth scalar function μ and any $\mathbb{R}^d$-valued function $\boldsymbol{\mu}$, we define the normal jumps and weighted averages as follows. For each facet $F \in \mathcal{F}_h^I$ shared by two elements K_1 and K_2, we set

$$\begin{aligned}
[\![\mu]\!]_{\boldsymbol{n}} &= \mu|_{K_1}\boldsymbol{n}_{K_1} + \mu|_{K_2}\boldsymbol{n}_{K_2}, \qquad & \{\!\{\mu\}\!\}_{\gamma_F} &= (1-\gamma_F)\mu|_{K_1} + \gamma_F\mu|_{K_2},\\
[\![\boldsymbol{\mu}]\!]_{\boldsymbol{n}} &= \boldsymbol{\mu}|_{K_1}\cdot\boldsymbol{n}_{K_1} + \boldsymbol{\mu}|_{K_2}\cdot\boldsymbol{n}_{K_2}, \qquad & \{\!\{\boldsymbol{\mu}\}\!\}_{\gamma_F} &= (1-\gamma_F)\boldsymbol{\mu}|_{K_1} + \gamma_F\boldsymbol{\mu}|_{K_2},
\end{aligned}$$

with $\gamma_F \in [0,1]$.
Finally, we define the parameters η_F and γ_F as

$$\eta_F = \eta_0\ell^2\frac{2\left(\boldsymbol{n}_{K_1}^\top D|_{K_1}\boldsymbol{n}_{K_1}\right)\left(\boldsymbol{n}_{K_2}^\top D|_{K_2}\boldsymbol{n}_{K_2}\right)}{\boldsymbol{n}_{K_1}^\top D|_{K_1}\boldsymbol{n}_{K_1} + \boldsymbol{n}_{K_2}^\top D|_{K_2}\boldsymbol{n}_{K_2}}, \qquad \gamma_F = \frac{\boldsymbol{n}_{K_1}^\top D|_{K_1}\boldsymbol{n}_{K_1}}{\boldsymbol{n}_{K_1}^\top D|_{K_1}\boldsymbol{n}_{K_1} + \boldsymbol{n}_{K_2}^\top D|_{K_2}\boldsymbol{n}_{K_2}}, \tag{6}$$

where ℓ is the polynomial approximation degree in the space discretization and $\eta_0 > 0$ is a constant independent of the problem coefficients and discretization parameters. The definitions of η_F and γ_F are guided by the literature on robust discontinuous Galerkin methods for diffusion problems with strongly heterogeneous coefficients, where the use of suitable weighted averages and penalty parameters ensures robustness.

2.4. Discretization spaces

Given a polynomial degree $\ell \in \mathbb{N}$ with $\ell \geq 1$, we define the spaces

$$\mathcal{W}_\ell(\mathcal{T}_h) = \prod_{K\in\mathcal{T}_h}\mathbb{P}_\ell(K), \qquad \boldsymbol{\mathcal{R}}_\ell(\mathcal{T}_h) = \prod_{K\in\mathcal{T}_h}\mathbb{P}_\ell(K)^d,$$

where $\mathbb{P}_\ell(K)$ denotes the space of scalar polynomials of degree at most $\ell \geq 1$ defined on K. Additionally, we denote by $\Pi_{\mathcal{W}}$ and $\Pi_{\boldsymbol{\mathcal{R}}}$ the $L^2(\Omega)$ - and $L^2(\Omega)^d$ - orthogonal projections onto $\mathcal{W}_\ell(\mathcal{T}_h)$ and $\boldsymbol{\mathcal{R}}_\ell(\mathcal{T}_h)$, respectively.
We introduce the discrete space gradient and divergence operators. The LDG gradient operator

$$\nabla_{\mathrm{LDG}} : \mathcal{W}_\ell(\mathcal{T}_h) \to \boldsymbol{\mathcal{R}}_\ell(\mathcal{T}_h),$$

is defined by

$$(\nabla_{\mathrm{LDG}} v_h, \boldsymbol{\phi}_h)_\Omega = (\nabla_h v_h - \mathcal{L}(v_h), \boldsymbol{\phi}_h)_\Omega \qquad \forall \boldsymbol{\phi}_h \in \boldsymbol{\mathcal{R}}_\ell(\mathcal{T}_h),$$

where ∇_h denotes the piecewise gradient operator and the jump lifting operator $\mathcal{L} : \mathcal{W}_\ell(\mathcal{T}_h) \to \boldsymbol{\mathcal{R}}_\ell(\mathcal{T}_h)$ is defined by

$$(\mathcal{L}(v_h), \boldsymbol{\phi}_h)_\Omega = \sum_{F\in\mathcal{F}_h^I}\left([\![v_h]\!]_{\boldsymbol{n}}, \{\!\{\boldsymbol{\phi}_h\}\!\}_{1-\gamma_F}\right)_F \qquad \forall \boldsymbol{\phi}_h \in \boldsymbol{\mathcal{R}}_\ell(\mathcal{T}_h).$$

The LDG divergence operator

$$\mathrm{div}_{\mathrm{LDG}} : \boldsymbol{\mathcal{R}}_\ell(\mathcal{T}_h) \to \mathcal{W}_\ell(\mathcal{T}_h),$$

is defined by

$$(\mathrm{div}_{\mathrm{LDG}}\, \boldsymbol{r}_h, \psi_h)_\Omega = -(\boldsymbol{r}_h, \nabla_{\mathrm{LDG}}\psi_h)_\Omega \qquad \forall \psi_h \in \mathcal{W}_\ell(\mathcal{T}_h).$$

2.5. BDF time integration

We discretize in time by the backward differentiation formula (BDF) with v steps. Since the BDF method is unstable for $\nu \geq 7$, we restrict to $\nu \leq 6$. For a non-autonomous ordinary differential equation $y' = g(y,t)$, the v-step BDF scheme, denoted by BDF_ν, reads

$$y^{(n+1)} - \nu \sum_{j=1}^{\nu} a_j \, y^{(n+1-j)} = \tau_{n+1} \beta \, g\big(y^{(n+1)}, t_{n+1}\big).$$

From now on, we omit the dependence on v and simply denote the coefficients by β and a_j. For uniform meshes, the coefficients for $\nu \leq 6$ can be found. In particular, for $\nu = 1$, the BDF method coincides with the backward Euler method, independently of the mesh.
Given a penalty parameter $\varepsilon > 0$, our BDF–LDG discretization of the reformulated FKPP problem4, after an initialization phase (e.g., using a one-step method), reads as follows. For $n = \nu - 1, \dots, N-1$, find

$$w_h^{(n+1)} \in \mathcal{W}_\ell(\mathcal{T}_h), \qquad \boldsymbol{z}_h^{(n+1)}, \, \boldsymbol{\sigma}_h^{(n+1)}, \, \boldsymbol{r}_h^{(n+1)} \in \boldsymbol{\mathcal{R}}_\ell(\mathcal{T}_h),$$

such that

$$\boldsymbol{z}_h^{(n+1)} = -\nabla_{\mathrm{LDG}} w_h^{(n+1)}, \tag{7a}$$

$$\big(\mathbf{D}s''(u(w_h^{(n+1)}))\boldsymbol{\sigma}_h^{(n+1)}, \boldsymbol{\phi}_h\big)_\Omega = \big(\mathbf{D}\boldsymbol{z}_h^{(n+1)}, \boldsymbol{\phi}_h\big)_\Omega \quad \forall \boldsymbol{\phi}_h \in \boldsymbol{\mathcal{R}}_\ell(\mathcal{T}_h), \tag{7b}$$

$$\boldsymbol{r}_h^{(n+1)} = \Pi_{\boldsymbol{\mathcal{R}}}\big(D\boldsymbol{\sigma}_h^{(n+1)}\big), \tag{7c}$$

$$\begin{aligned} \varepsilon(w_h^{(n+1)}, \psi_h)_{\mathrm{LDG}} + \frac{1}{\tau_{n+1}\beta}\Big(u(w_h^{(n+1)}) - \nu \sum_{j=1}^{\nu} a_j u_h^{(n+1-j)}, \psi_h\Big)_\Omega \\ + \big(\mathrm{div}_{\mathrm{LDG}}\, \boldsymbol{r}_h^{(n+1)}, \psi_h\big)_\Omega + \sum_{F \in \mathcal{F}_h^I} \big(\mathbf{h}^{-1}[\![w_h^{(n+1)}]\!]_{\boldsymbol{n}}, [\![\psi_h]\!]_{\boldsymbol{n}}\big)_F \\ = \big(f(u(w_h^{(n+1)})), \psi_h\big)_\Omega \quad \forall \psi_h \in \mathcal{W}_\ell(\mathcal{T}_h). \end{aligned} \tag{7d}$$

In (7d), the bilinear form $(\cdot,\cdot)_{\mathrm{LDG}}$ is defined as

$$(w,\psi)_{\mathrm{LDG}} = (\alpha w, \psi)_\Omega + (D\nabla_{\mathrm{LDG}} w, \nabla_{\mathrm{LDG}}\psi)_\Omega + \sum_{F \in \mathcal{F}_h^I} \big(\mathbf{h}^{-1}[\![w]\!]_{\boldsymbol{n}}, [\![\psi]\!]_{\boldsymbol{n}}\big)_F,$$

which is coercive with respect to the norm

$$\|w\|_{\mathrm{DG}}^2 = \|\alpha^{1/2} w\|_{L^2(\Omega)}^2 + \|D^{1/2}\nabla_h w\|_{L^2(\Omega)^d}^2 + \sum_{F \in \mathcal{F}_h^{\mathcal{I}}} \|\mathbf{h}^{-1/2}[\![w]\!]_{\boldsymbol{n}}\|_{L^2(F)^d}^2,$$

See e.g. [1]. The terms $u_h^{(\kappa)}$ appearing in the BDF summation are defined by

$$u_h^{(\kappa)} = \begin{cases} \Pi_{\mathcal{W}} c_0, & \kappa = 0, \\ \Pi_{\mathcal{W}} u(w_h^{(\kappa)}), & \kappa > 0, \end{cases}$$

that is, $u_h^{(0)}$ is the $L^2(\Omega)$-projection of the initial condition for the original variable c, while for $\kappa > 0$ the terms are the $L^2(\Omega)$-projections of the transformed variables computed at previous time steps. Finally, since $\mathbf{D}s''(u(w))$ is uniformly positive definite (see (2.8)), given $w_h^{(n+1)}$ and $\boldsymbol{z}_h^{(n+1)}$, equation (7b) determines $\boldsymbol{\sigma}_h^{(n+1)}$ uniquely.

2.6. Matrix form

In order to write the BDF–LDG method in Equation 4 in matrix form, we define the following bilinear forms and nonlinear functionals: for all $w_h, \psi_h \in \mathcal{V}^\ell(\mathcal{T}_h)$ and $\boldsymbol{z}_h, \boldsymbol{\sigma}_h, \boldsymbol{\phi}_h \in \boldsymbol{\mathcal{R}}^\ell(\mathcal{T}_h)$,

$$m_h(\boldsymbol{z}_h, \boldsymbol{\phi}_h) = (\boldsymbol{z}_h, \boldsymbol{\phi}_h)_\Omega, \tag{8a}$$

$$b_h(w_h, \boldsymbol{\phi}_h) = (\nabla_h w_h, \boldsymbol{\phi}_h)_\Omega - \sum_{F \in \mathcal{F}_h^i} \big(\llbracket w_h \rrbracket_{\mathbf{N}}, \{\!\{\boldsymbol{\phi}_h\}\!\}_{1-\gamma_F}\big)_F, \tag{8b}$$

$$j_h(w_h, \psi_h) = \sum_{F \in \mathcal{F}_h^i} \big(\mathsf{h}^{-1} \llbracket w_h \rrbracket_{\mathbf{N}}, \llbracket \psi_h \rrbracket_{\mathbf{N}}\big)_F, \tag{8c}$$

$$d_h(\boldsymbol{z}_h, \boldsymbol{\phi}_h) = \sum_{K \in \mathcal{T}_h} (\mathbf{D}\boldsymbol{z}_h, \boldsymbol{\phi}_h)_K, \tag{8d}$$

$$n_h(w_h; \boldsymbol{\sigma}_h, \boldsymbol{\phi}_h) = \sum_{K \in \mathcal{T}_h} \big(\mathbf{D}s''(u(w_h))\boldsymbol{\sigma}_h, \boldsymbol{\phi}_h\big)_K, \tag{8e}$$

$$u_h(w_h, \psi_h) = (u(w_h), \psi_h)_\Omega, \tag{8f}$$

$$f_h(w_{p,h}, w_{q,h}; \psi_h) = \big(f(u(w_h)), \psi_h\big)_\Omega. \tag{8g}$$

For fixed bases of $\mathcal{V}^\ell(\mathcal{T}_h)$ and $\boldsymbol{\mathcal{R}}^\ell(\mathcal{T}_h)$, respectively, we denote by $\mathbf{W}_h$, $\mathbf{Z}_h$, $\boldsymbol{\Sigma}_h$, and $\mathbf{R}_h$ the coefficient vectors expressing w_h, $\boldsymbol{z}_h$, $\boldsymbol{\sigma}_h$, and $\boldsymbol{r}_h$, in terms of those bases, respectively.
Moreover, we denote by $\mathbf{M}_I$, $\mathbf{B}$, $\mathbf{J}$, $\mathbf{M}_D$, and $\mathbf{A}_{\text{LDG}}$ the matrices associated with the bilinear forms $m_h(\cdot,\cdot)$, $b_h(\cdot,\cdot)$, $j_h(\cdot,\cdot)$, $d_h(\cdot,\cdot)$, and $(\cdot,\cdot)_{\text{LDG}}$, respectively, and by $\mathcal{N}$, $\mathcal{U}$, and $\mathcal{F}$ the operators associated with the nonlinear functionals $n_h(\cdot;\cdot,\cdot)$, $u_h(\cdot,\cdot)$, and $f_h(\cdot,\cdot;\cdot)$, respectively.
Since the nonlinear functional $n_h(\cdot;\cdot,\cdot)$ is linear with respect to its second argument, we write

$$\mathcal{N}\big(\mathbf{W}_h^{(n+1)}\big)\boldsymbol{\Sigma}_h^{(n+1)},$$

instead of

$$\mathcal{N}\big(\mathbf{W}_h^{(n+1)}; \boldsymbol{\Sigma}_h^{(n+1)}\big).$$

As the mass matrix $\mathbf{M}_I$ is symmetric, positive definite, and block diagonal, it can be inverted efficiently. Hence, we can write

$$\mathbf{Z}_h^{(n+1)} = -\mathbf{M}_I^{-1}\mathbf{B}\mathbf{W}_h^{(n+1)}, \tag{9}$$

$$\mathbf{R}_h^{(n+1)} = \mathbf{M}_I^{-1}\mathbf{M}_D\boldsymbol{\Sigma}_h^{(n+1)}. \tag{10}$$

Given $\mathbf{W}_h^{(n+1)}$, eq. (9) determines $\boldsymbol{\Sigma}_h^{(n+1)}$ in a unique way. This follows from the uniform positive definiteness of $\mathbf{D}s''(u(w))$; see (2). Consequently, we obtain

$$\boldsymbol{\Sigma}_h^{(n+1)} = -\mathcal{N}\big(\mathbf{W}_h^{(n+1)}\big)^{-1}\mathbf{M}_D\mathbf{M}_I^{-1}\mathbf{B}\mathbf{W}_h^{(n+1)}. \tag{11}$$

Using the above relations, the system of equations can be reformulated in terms of the sole unknown $\mathbf{W}_h^{(n+1)}$ as follows

$$\begin{aligned}\varepsilon\mathbf{A}_{\text{LDG}}\mathbf{W}_h^{(n+1)} + \frac{1}{\tau_{n+1}\beta}\mathcal{U}\big(\mathbf{W}_h^{(n+1)}\big) + \mathbf{B}^T\mathbf{M}_I^{-1}\mathbf{M}_D\mathcal{N}\big(\mathbf{W}_h^{(n+1)}\big)^{-1}\mathbf{M}_D\mathbf{M}_I^{-1}\mathbf{B}\mathbf{W}_h^{(n+1)} \\ + \mathbf{J}\,\mathbf{W}_h^{(n+1)} = \mathcal{F}\big(\mathbf{W}_h^{(n+1)}\big) + \frac{1}{\tau_{n+1}\beta}\sum_{j=1}^{\nu} a_j\,\mathcal{U}\big(\mathbf{W}_h^{(n+1-j)}\big).\end{aligned} \tag{12}$$

2.7. Newton's method

To complete the presentation of the method, the formulation stemming from Newton's iteration is derived. Let $C = M_D M_I^{-1} B$, we obtain

$$\begin{aligned}&\Big[\varepsilon A_{\text{LDG}} + \frac{1}{\tau^{n+1}\beta}\mathcal{D}_{\mathbf{W}}\big(\mathcal{U}(\mathbf{W}_h^k)\big) + C^T\big(\mathcal{A}(\mathbf{W}_h^k)\big)^{-1}C - \mathbb{M}\mathbf{W}_h^k + J - \mathcal{D}_{\mathbf{W}}\big(\mathcal{F}(\mathbf{W}_h^k)\big)\Big]\mathbf{W}_h^{k+1} \\ &= \Big[\frac{1}{\tau^{n+1}\beta}\mathcal{D}_{\mathbf{W}}\big(\mathcal{U}(\mathbf{W}_h^k)\big) - \mathcal{D}_{\mathbf{W}}\big(\mathcal{F}(\mathbf{W}_h^k)\big) - \mathbb{M}\mathbf{W}_h^k\Big]\mathbf{W}_h^k \\ &\quad - \frac{1}{\tau^{n+1}\beta}\left[\mathcal{U}(\mathbf{W}_h^k) - \nu\sum_{j=1}^{n}\mathcal{U}_h^{(n+1-j)}\right] + \mathcal{F}(\mathbf{W}_h^k),\end{aligned} \tag{13}$$

where $\mathbb{M}$ is the third-order tensor defined as

$$\mathbb{M} = C^T \left(\mathcal{A}(\mathbf{W}_h^k)\right)^{-1} \mathcal{D}_{\mathbf{W}} \left(\mathcal{A}(\mathbf{W}_h^k)\right) \left(\mathcal{A}(\mathbf{W}_h^k)\right)^{-1} C.$$

3. Entropy transformations

The first feature we wanted to test is the behavior of the scheme for different choices of the entropy function. The standard choice in [1] was the **Fermi-Dirac** function, i.e.

$$u(w) = \frac{e^w}{1+e^w}.$$

Recalling the definition of *entropy function* and its properties given in Definition 1, these are sufficient properties needed in order to guarantee the well-posedness of the problem and the convergence of the method, but they are **not** necessary. Therefore, we chose other possible entropy functions that possess the same properties
All the transformations are either the same as those shown in [7] or a slightly modified version to allow $u(w)$ to map $\mathbb{R} \to [0, 1]$.

We first introduce **Shannon-Boltzmann function**, given by

$$u(w) = e^w.$$

A notable feature of this transformation is its lack of upper bound. This typically does not cause any problems since in the original equation, $c = 1$ is a stable equilibrium of the stationary version of equation (1a). Next, we define the **Hellinger function**, given by

$$u(w) = \frac{1}{2}\left(1 + \frac{w}{\sqrt{1+w^2}}\right).$$

This function is similar in shape to the **Fermi-Dirac** function, but has different features from an asymptotic point of view. Lastly, we define k-**Fermi-Dirac**, defined as:

$$u(w) = \frac{k^w}{1+k^w}, \quad k > 0.$$

This function is a modified version of the original **Fermi-Dirac** transformation. All its *entropy quantities* differ from **Fermi-Dirac**'s *entropy quantities* by a multiplicative constant. In Figure 1 we plot both $u(w)$ and $s''(c)$ for each one of these transformations.

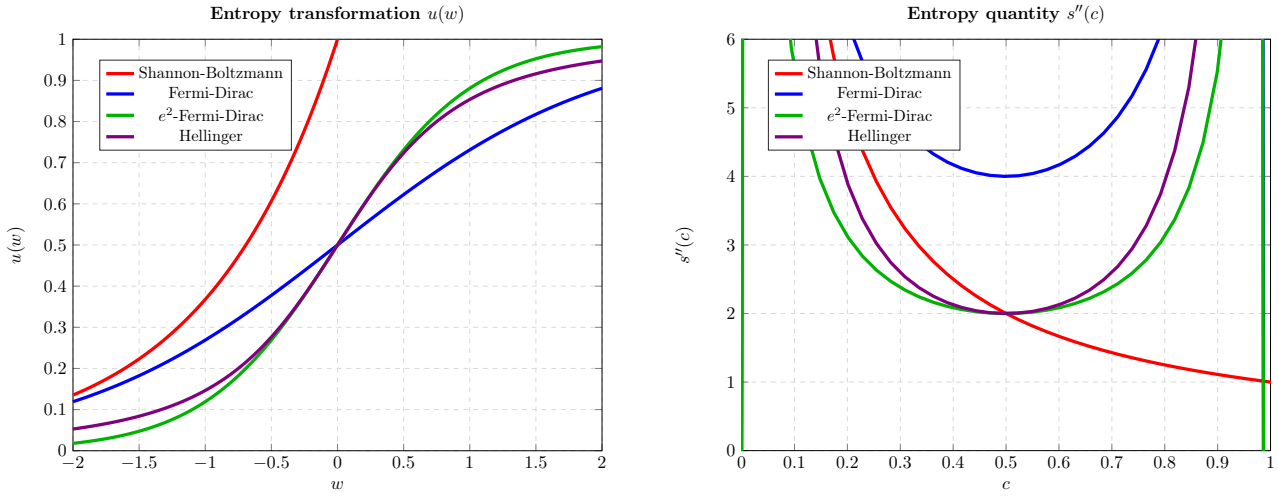


Figure 1: Entropy functions $u(w)$ (left) and entropy quantity $s''(c)$ (right) plots

4. Numerical Results

Most of the tests are the same as those performed in [1]. For all tests $\theta = -1$ is fixed and the stabilization parameter $\eta_0 = 1$ is set in the definition of η_F. For the Newton iterations, given a small tolerance tol, the

following stopping criterion is adopted:

$$\min\left\{\left\|w_h^{k+1}-w_h^k\right\|_{L^2(\Omega)},\ \left|\mathrm{res}^{k+1}\right|\right\}\leq \mathrm{tol}, \tag{3.1}$$

where res^{k+1} is the residual of the algebraic system (2.16) for the approximation at the $(k+1)$th Newton iteration of $w_h^{(n+1)}$.
In the convergence tests reported below, we measure the errors at the final time, i.e.

$$E_c=\left\|c(\cdot,T)-u\left(w_h^{(N)}\right)\right\|_{L^2(\Omega)}, \tag{14}$$

and

$$E_\sigma=\left\|\nabla c(\cdot,T)+\sigma_h^{(N)}\right\|_{L^2(\Omega)^d}. \tag{15}$$

4.1. Test Case 1

In the first test case we repeat the results obtained in [1] with a manufactured solution on a square domain $\Omega=(0,1)^2$ and known exact solution:

$$c_{ex}(x,t)=\frac{1}{4}\big(\cos(2\pi x)\cos(2\pi y)+2\big)(1-t),$$

All the other model's parameters are described in Table 1. The test is performed on progressively refined polytopal meshes and for increasing polynomial degree ℓ. The source term is chosen accordingly.

Parameter	Name	Value
Ω	Domain	$(0,1)^2$
$\mathbf{D}$	Diffusion tensor	$\mathbb{I}_2$
α	Reaction coefficient	1
`tol`	Newton stopping criterion	10^{-10}
N_{el}	Number of elements	$30, 100, 300, 1000$
ℓ	Polynomial degree	$1, 2, 3, 4$
c_{ex}	Exact solution	$\frac{1}{4}\big(\cos(2\pi x)\cos(2\pi y)+2\big)(1-t)$
τ	Time step	10^{-3}
T	Final time	$5\cdot 10^{-2}$
ν	BDF scheme order	1

Table 1: Setup of Test Case 1

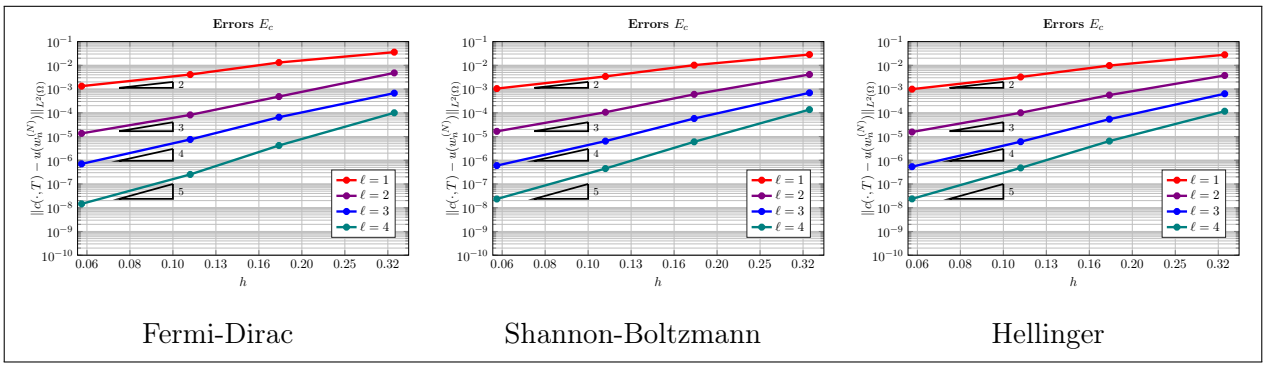


Figure 2: Computed error E_c (defined in 14) for Test Case 1

In Figures 2 and 3 is shown the E_c error and the E_σ errors, respectively. The errors are plotted with respect to the mesh size h and for each polynomial degree ℓ. The results demonstrate that for all tested variable transformations the error decay at the expected rate, for both the L^2 and the energy norm. In particular, all of the errors decrease according to the results shown in [1]. It is worth noting that the results for this test case are practically indistinguishable among the different entropy functions. There are slight variations of the error, but all the entropy functions perform equally well.

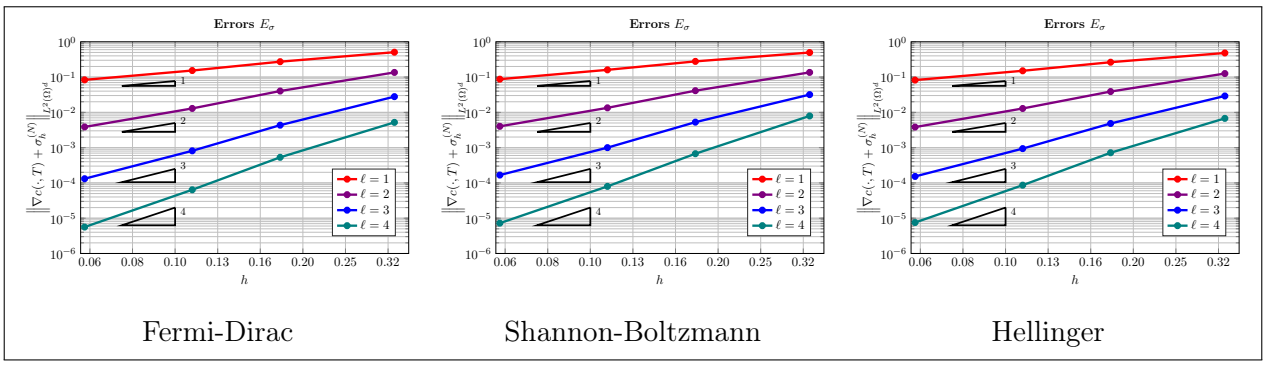


Figure 3: Computed error E_σ (defined in 15) for Test Case 1

4.2. Test Case 2

In the second test case, we consider a traveling wave solution on a rectangular $\Omega = (0, 3) \times (0, 1)$, with progressively fine meshes, for increasing orders of BDF, with polynomial degree $\ell = 2$ and comparing the output with the known exact solution:

$$c_{ex}(x, t) = \frac{1}{4}\left(1 + \tanh\left(8 - \sqrt{\frac{\alpha}{24\, d_{\text{ext}}}}\,\xi\right)\right)^2, \quad \xi = x - vt,$$

The setup is shown in Table 2. The results provided will not feature **Hellinger's** transformation since it failed to converge in every case in this test.

Parameter	Name	Value
Ω	Domain	$(0, 3) \times (0, 1)$
$\mathbf{D}$	Diffusion tensor	$10^{-3}\mathbb{I}_2$
α	Reaction coefficient	$\frac{6}{25 d_{\text{ext}}}\nu^2 = 1$
v	Wave speed	6.45×10^{-2}
`tol`	Newton stopping criterion	10^{-10}
N_{el}	Number of elements	$50, 100, 200, 400, 800, 1600$
c_{ex}	Exact solution	$\frac{1}{4}\left(1 + \tanh\left(8 - \sqrt{\frac{\alpha}{24\, d_{\text{ext}}}}\,\xi\right)\right)^2$
τ	Time step	10^{-1}
T	Final time	1
ν	BDF order	1,2,3,4,5

Table 2: Setup of Test Case 2

Starting from the case $\ell = 2$, in Figure 4 is shown the E_c error for both the **Fermi-Dirac** and **Shannon-Boltzmann** transformation. The errors are plotted with respect to the mesh size h and for increasing BDF order ν. This test reveals the first significant differences among the transformations. First, the **Shannon-Boltzmann** transformation takes longer to reach the plateau for **BDF1** and **BDF2**. Moreover, there is essentially no difference between **BDF3** and **BDF4**. Finally, even though the error decreases according to theoretical estimates, **Shannon-Boltzmann**'s transformation always provides a larger error, numerically speaking. This means that **Shannon-Boltzmann** is indeed a feasible entropy function for this problem. The estimate proves to be worse than the one with the classic **Fermi-Dirac** as a consequence of the unboundedness of the exponential. This does not seem to affect convergence since $c = 1$ is a stable equilibrium of equation (1a), but it does affect performance. The test results for **BDF5** are not shown as the difference with **BDF4** is negligible.

4.3. Test Case 3

In the third test case we perform a test which is identical to Section 4.2, but this time with a polynomial approximation degree of $\ell = 3$. The setup is described in Table 2. Again, in Figure 5 we are plotting the E_c error for both the **Fermi-Dirac** and **Shannon-Boltzmann** transformation. The errors are plotted with respect to the mesh size h and for increasing BDF order ν. By increasing the polynomial degree we encounter

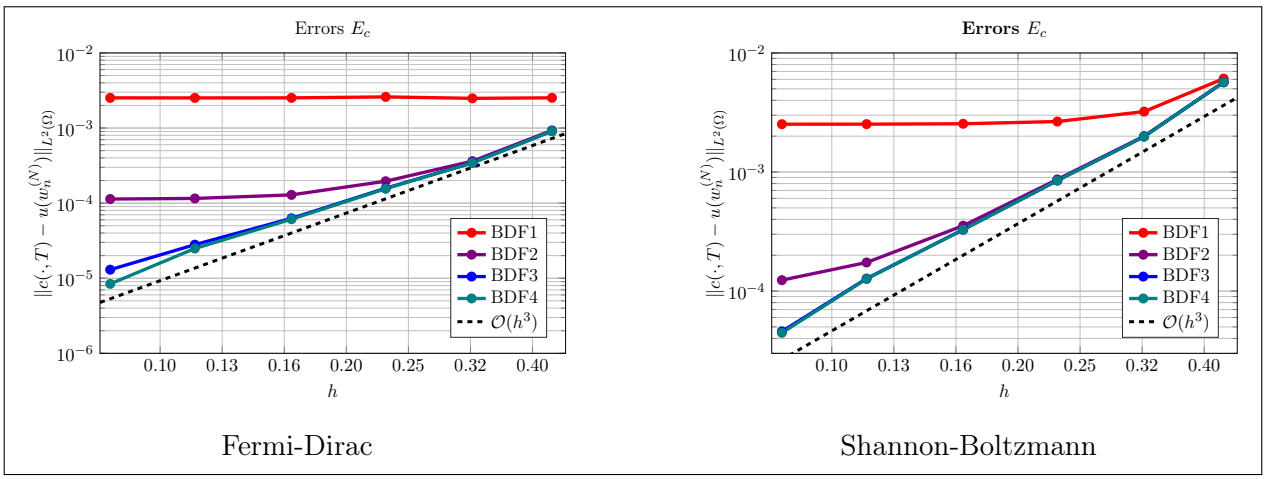


Figure 4: Computed error E_c for Fermi-Dirac transformation (left) and E_c for Shannon-Boltzmann transformation (right) for Test Case 2

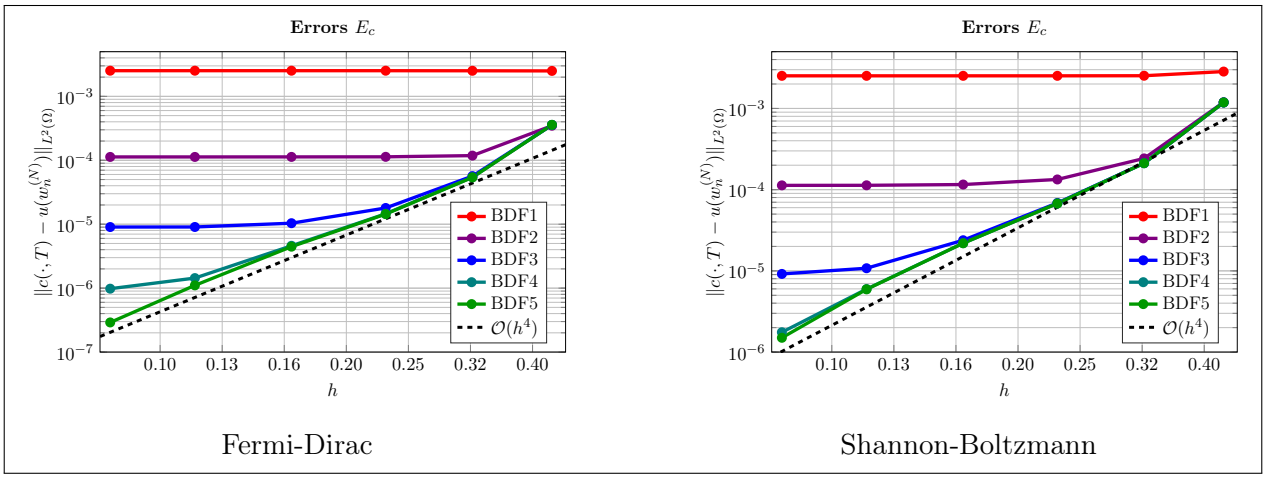


Figure 5: Computed error E_c for Fermi-Dirac transformation (left) and E_c for Shannon-Boltzmann transformation (right) for Test Case 3

issues analogous to those of the previous case. Coherently with the previous test, the **Shannon-Boltzmann** transformation takes longer to reach the plateau and the difference between **BDF4** and **BDF5** is negligible. Finally, **Shannon-Boltzmann**'s transformation again provides a larger error, despite the fact that it decreases according to theory.

4.4. Hellinger's failure

Hellinger's transformation proved itself to be theoretically a valid choice in Test Case 1. However, Test Case 2 yielded different results. When testing on a traveling wave solution, the failure is due to the behavior of the algorithm in the presence of a sharp front. More precisely, the method reconstructs the wave front to be less steep than it actually is, hence bringing the iterative method to fail over time due to error accumulation. The problem is not specific to the mesh, since it appears even for smaller values of h. The plots in Figures 6, 7, and 8 clearly show that this phenomenon consistently appears in the same region.

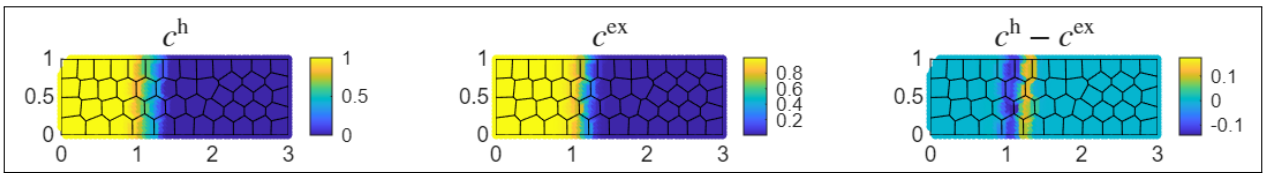


Figure 6: Test Case 3, Hellinger solution plot for $t = 0.2$, $N = 50$

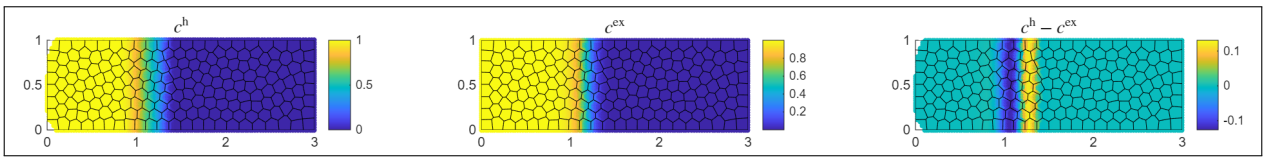


Figure 7: Test Case 3, Hellinger solution plot for $t = 0.2$, $N = 400$

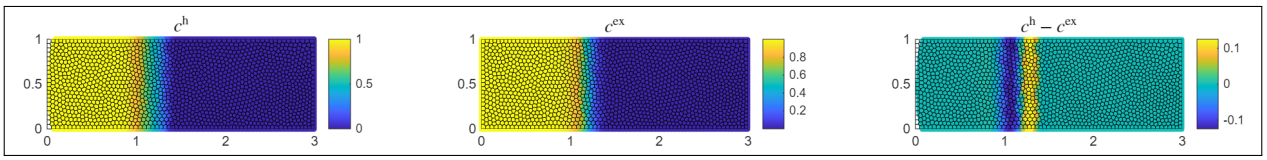


Figure 8: Test Case 3, Hellinger solution plot for $t = 0.2$, $N = 1600$

To test this hypothesis, I ran another test with a different initial datum. The setup is described in Table 3. The objective was to check whether the scheme can work with a solution of a different shape. That's why I chose a gaussian bell as a initial datum. As we can see in Figure 9, by comparing the results between **Fermi-Dirac** and **Hellinger**, we notice how the scheme tends to be very diffusive, meaning that it tends to flatten high slopes. We can easily tell that **Hellinger**'s solution tends to be more smoothed out in the domain. This test suggests furtherly that this change of variable is not useful from a practical point of view.

Parameter	Name	Value
Ω	Domain	$(0,3) \times (0,1)$
$\mathbf{D}$	Diffusion tensor	$10^{-3}\mathbb{I}_2$
α	Reaction coefficient	$\frac{6}{25 d_{\text{ext}}}\nu^2 = 1$
v	Wave speed	6.45×10^{-2}
`tol`	Newton stopping criterion	10^{-10}
N_{el}	Number of elements	1600
ℓ	Polynomial degree of approximation	3
c_0	Initial datum	$0.6 \exp\left(-\frac{(x-1.5)^2+(y-0.5)^2}{2 \cdot 0.1^2}\right)$
τ	Time step	10^{-1}
T	Final time	1
ν	BDF order	1

Table 3: Setup of Hellinger Gaussian Bell Test

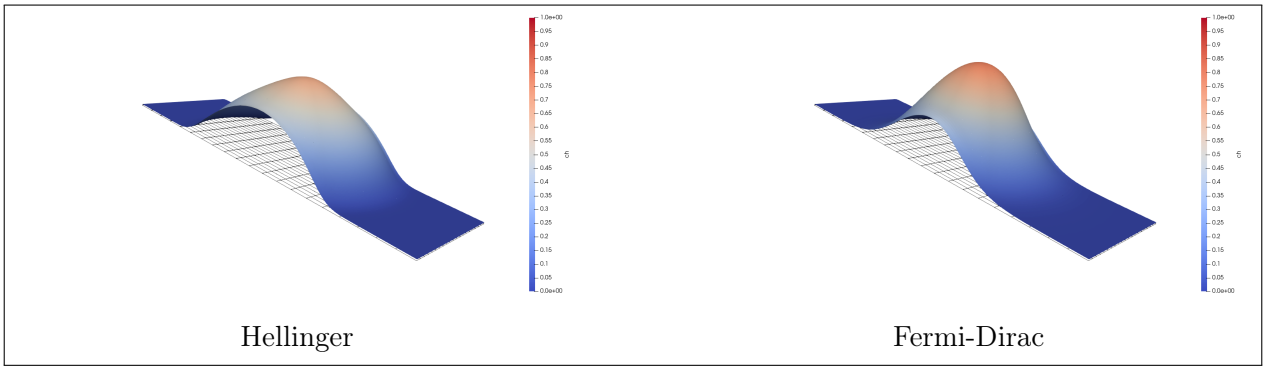


Figure 9: Plots of **Hellinger** and **Fermi-Dirac** calculated solution with Gaussian bell initial datum, Test Case 3

4.4.1 k-FD testing

The issues faced with **Hellinger's** transformation led us to test a modified version of **Fermi-Dirac**'s transformation, the k-**Fermi-Dirac** (k-**FD** for short) as follows:

$$u(w) = \frac{k^w}{1+k^w}, \quad k > 1,$$

The idea is to test whether the slope of the transformation function at 0 affects the scheme. To this purpose we tested for various values of $k = e^{1/2}, e, e^2, e^3, e^4$ on progressively finer meshes. Note that the e-**FD** transformation is just the standard **Fermi-Dirac**. Another important remark is that when calculating all the *entropy quantities* (*entropy function*, *first derivative of entropy function*...), these are the same with respect to the **Fermi-Dirac**, except for a multiplicative constant. This means that all *entropy quantities* have essentially the same shape as the original transformation. To test its performance, we selected Test Case 2 with the data reported in Table 4

Parameter	Name	Value
Ω	Domain	$(0,3)\times(0,1)$
$\mathbf{D}$	Diffusion tensor	$10^{-3}\mathbb{I}_2$
α	Reaction coefficient	$\frac{6}{25 d_{\text{ext}}}\nu^2 = 1$
v	Wave speed	6.45×10^{-2}
`tol`	Newton stopping criterion	10^{-10}
N_{el}	Number of elements	$50, 100, 200, 400, 800, 1600$
ℓ	Polynomial degree of approximation	3
c_{ex}	Exact solution	$\frac{1}{4}\left(1+\tanh\left(8-\sqrt{\frac{\alpha}{24\, d_{\text{ext}}}}\,\xi\right)\right)^2$
τ	Time step	10^{-1}
T	Final time	1
ν	BDF order	5

Table 4: Setup of Test Case 2, k-FD

In Figure 10b, we are plotting the E_c error fo the evaluated solution, while in Figure 10a, we are plotting the shape of the entropy transformation function, $u(w)$, for different values of k. We chose to make a comparison only for the higher BDF order, since it is the only case in which the error does not converge to a plateau. We can appreciate in 10b that for increasing values of k, our estimate improves. We see clearly that, with respect to the standard case ($k = e$): if $k < e$, the estimate is better for coarser meshes, but for finer meshes it becomes worse. If $k > e$, The estimate is worse for coarser meshes, but it becomes better for finer meshes. For high values of k (i.e., e^3, e^4), the method becomes unstable with a coarse mesh and diverges. These results lead to two main observations: the choice of an optimal k is not fundamental to improving the performance of the code, since for all $k > e$ results proved to be similar. Also, the performance of a transformation depends both on the steepness of the entropy transformation at $w = 0$ and its asymptotic properties. By comparing the values of **Hellinger**'s entropy function second derivative with respect to the other transformation choices, we observe that as $c \to 0$, the function goes to infinity more rapidly. Of course the same happens as $c \to 1$, but since $c = 1$ is a stable equilibrium of the stationary equation (1a), it does not significantly affect the scheme.

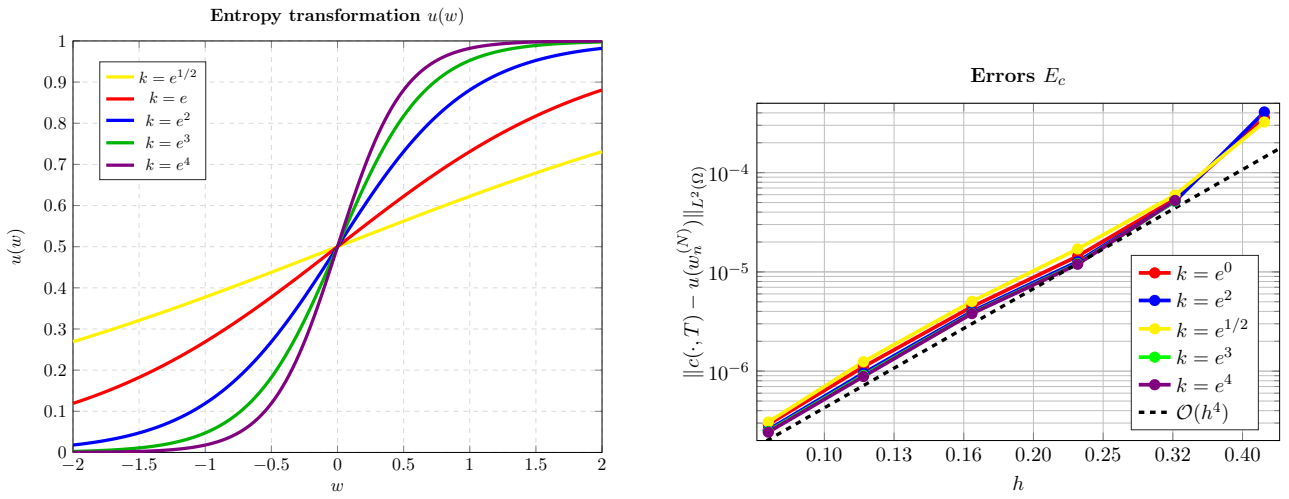
5. Performance of nonlinear iterative solvers

Another issue of the numerical method is the choice of the iterative method to solve Equation 12. In particular, the choice of Newton's method comes with a downside: the method is very sensitive to the choice of the initial condition. This section, aims to find a more robust iterative method.

5.1. Absract framework

Let us consider the same setting of [8] for a nonlinear system of equations. Let the nonlinear problem in a real Hilbert space X with inner product $(\cdot,\cdot)_X$ and induced norm $\|\cdot\|_X$:

$$u \in X: \quad F(u) = 0 \quad \text{in } X', \tag{16}$$



(a) Plot of entropy transformation

(b) E_c errors for different k

Figure 10: k-FD test results

where $F : X \to X'$ is a nonlinear operator and X' denotes the dual space of X. In weak form:

$$u \in X : \quad \langle F(u), v \rangle = 0 \quad \forall v \in X, \tag{17}$$

with $\langle \cdot, \cdot \rangle$ signifying the duality pairing in $X' \times X$. The approach is based on the use of preconditioning operators. For given $v \in X$, we introduce a linear and invertible preconditioning operator:

$$A(v) : X \to X', \tag{18}$$

which allows us to transform the problem into $A(u)^{-1}F(u) = 0$. This gives rise to a fixed-point iteration:

$$u^{n+1} = u^n - A(u^n)^{-1}F(u^n), \quad n \geq 0, \tag{19}$$

This general scheme recovers several well-known iterative methods. In this case, we considered the following methods:

- **Richardson Iterations**
$$w^{n+1} = w^n - \delta F(w^n), \quad \delta \in \mathbb{R}^+.$$
- **"Warm-Start" Newton**
$$\begin{aligned} w^{n+1} &= w^n - \delta F(w^n), && n < k \\ w^{n+1} &= w^n - J(F(w^n))^{-1}F(w^n). && n \geq k \end{aligned}$$
- **Alternating Newton** Given $k \in \mathbb{N}$
$$\begin{aligned} w^{n+1} &= w^n - \delta F(w^n), && n = mk, m \in \mathbb{N} \\ w^{n+1} &= w^n - J(F(w^n))^{-1}F(w^n). && n \text{ otherwise} \end{aligned}$$

where $J(F(u^n))$ denotes the Jacobian of F evaluated at u^n. Recalling that the standard **Newton**'s method has this form: *Given the n-th iteration w^n, the next iteration is given by:*

$$w^{n+1} = w^n - J(F(w^n))^{-1}F(w^n).$$

This method has a quadratic convergence rate, meaning that every iteration is such that:

$$\|e^n\| = \|w_{ex} - w^{n+1}\| \leq C\|w_{ex} - w^n\|^2,$$

while the **Richardson** method has a linear rate of convergence. Consequently, all of these methods are going to be **slower** than Newton in any case.

5.1.1 Richardson iteration

The first issue with Richardson Iterations is the estimate of the optimal parameter δ. In [8] it is proven that in order to obtain convergence we need

$$\delta \in \left(0, \frac{2\nu_0}{L_F^2}\right).$$

The parameters ν_0 and L_F, described in [8], depend on $F(u)$. The optimal parameter is given by:

$$\delta = \frac{2}{\lambda_{max} + \lambda_{min}} \approx \frac{2}{\lambda_{max}},$$

(see [15]), where λ_{min} and λ_{max} are the minimum and maximum eigenvalues of $J(F(u^n))$, respectively. This last approximation is admissible in the case of ill-conditioned matrices. To estimate the optimal value of λ, we decided to use a **Power Method** [14]. The algorithm reads as reported in Algorithm 1

Algorithm 1 Power Method

1: Initialise random vector v
2: Normalise v
3: $n_{it} = 0$
4: **for** $n_{it} < n_{max}$ or $J(w^n)v < tol$ **do**
5: Evaluate $F(w^n)$
6: Evaluate $F(w^n + \varepsilon v)$
7: Approximate $J(w^n)v \approx \frac{|F(w^n+\varepsilon v) - F(w^n)|)}{\varepsilon}$
8: Approximate $\lambda_{max} \approx v' \cdot J(w^n)v$
9: Evaluate $v_{new} = \frac{J(w^n)v}{\|v\|}$ for the next iteration
10: $n_{it}++$
11: **end for**

The method obviously comes with an a priori computational cost, especially for high degrees of freedom, but it typically gives a very good estimate in a few iterations. To save some calculations, in most cases the optimal δ is calculated only once at the beginning of the code.

5.1.2 Warm-Start Newton

We will denote by n_{FP} the number of **Richardson** iterations and by n_{newt} the **Newton** iterations. Also, for this algorithm, it will be useful to define the following:

Definition 2. *The relative improvement between two iterations is defined as*

$$\rho_{n+1} = \frac{|F(w^{n+1})|}{|F(w^n)|}.$$

Naturally, a good algorithm is such that at all iterations $\rho \ll 1$.

Algorithm 2 Warm-Start

Estimate δ by Power Method
Initialise w_0
$n_{it} = 0$
stopFP = false
while $n_{it} < n_{max}$ **do**
  if $n_{it} < n_{FP}$ **then**
    $w^{n+1} = w^n - \delta F(w^n)$
    if $\rho_{n+1} > 0.9$ **then**
      stopFP = true
      if $\rho_{n+1} > 1$ **then**
        $w^{n+1} = w^n$
      end if
    end if
  else if $n_{newt} > n_{it} \geq n_{FP}$ **or** stopFP == true **then**
    $w_*^{n+1} = w^n - \delta F(w^n)$
    $w_N^{n+1} = w^n - J(F(w^n))^{-1} F(w^n)$
    $w^{n+1} = \alpha w_*^{n+1} + (1-\alpha) w_N^{n+1}$
  else
    $w^{n+1} = w^n - J(F(w^n))^{-1} F(w^n)$
  end if
  $n_{it} + +$
end while

The method reported by Algorithm 2 has two mandatory phases and one optional intermediate phase. In the **first phase**, it performs a few **Richardson** iterations before switching. Typically, the number of iterations in this phase is fixed, but if the relative improvement goes over a certain threshold (typically 0.9), it means that it is performing poorly, and it skips to the next phase. In most of the tests, however, this feature will not be used. On the other hand, if $\rho_n > 1$, the method discards the last iteration and skips to the next phase. In the **second phase** (which is optional), it mixes the **Richardson** and **Newton** iterations to allow for a smooth transition. Of course, computing both updates is very costly, so in a more practical scenario

$$w_*^{n+1} = w^n,$$

given that $\delta F(w^n) \approx 0$ after only a few iterations. This makes the second phase equivalent to a **Relaxed Newton** iteration. The **third phase** is just **Newton**'s method. The addition of a "relaxed phase" turns out to be necessary, since typically **Newton's method** can overshoot the estimate of the next iteration as shown in Table 5.

Residual	**Newton**	**Warm-Start**	**Warm-Start (Relaxed)**
Iteration Richardson 1	/	$4.476 \cdot 10^{-2}$	$4.476 \cdot 10^{-2}$
Iteration Richardson 2	/	$4.386 \cdot 10^{-2}$	$4.386 \cdot 10^{-2}$
Iteration Richardson 3	/	$4.347 \cdot 10^{-2}$	$4.347 \cdot 10^{-2}$
Iteration Newton 1	$1.328 \cdot 10^{-1}$	$1.328 \cdot 10^{-1}$	$6.958 \cdot 10^{-2}$

Table 5: Residual of the first four iterations of **Warm-Start** and **Newton**'s method in Test Case 1

We can see that, without relaxation, the first iteration of **Newton**'s method is identical to the **Warm-Start** one, meaning that **Newton**'s method can't tell the difference if two initial guesses are too close and ignores the previous iterations. Moreover, the estimate gets notably worse. The second phase is added to guarantee a smooth transition between the methods.

5.1.3 Alternating Newton

As suggested by the name, the method alternates k iterations of the **Richardson** method with one iteration of **Newton**'s method. The algorithm is reported in Algorithm 3

Algorithm 3 Alternating Newton

Estimate δ by Power Method
Initialise w_0
$n_{it} = 0$
while $n_{it} < n_{max}$ **do**
  if $n_{it} + 1 \neq 0 \mod (k+1)$ **then**
    $w^{n+1} = w^n - \delta F(w^n)$
  else
    $w^{n+1} = w^n - J(F(w^n))^{-1} F(w^n)$
  end if
  $n_{it} + +$
end while

5.2. Numerical results

The test cases are analogous to the ones performed in Section 4. For all the following tests, I will not be reporting the numerical error of the schemes. This is because whenever two schemes with the same tolerance converged, they converged to the same solution, since they were solving the same problem. Hence, the "goodness" of the scheme is evaluated only on the computational cost and whether it did or did not reach convergence. **Richardson**'s results won't be featured as it always failed to converge within a reasonable number of iterations. As a convention, in all shown results "—" represents failure to reach convergence.

5.2.1 Test Case 4

The first test is designed to compare the computational costs of the various methods using the parameter configuration detailed in Table 6. To perform this comparison, I measure five performance indicators: the average number of iterations required per time step, the average total elapsed time per time step (expressed in seconds), the average time per individual iteration (in seconds), the average percentage of computational time dedicated to Richardson iterations per time step, and the corresponding percentage for Newton iterations.

Parameter	Name	Value
Ω	Domain	$[0,1] \times [0,1]$
$\mathbf{D}$	Diffusion tensor	$\mathbb{I}_2$
`Ntol`	Newton stopping criterion	10^{-10}
`FPtol`	Fixed point stopping criterion	10^{-10}
`WSit`	Fixed point Warm-Start iterations	3
`Altit`	Alternating fixed point iterations	3
N_{el}	Number of elements	300
ℓ	Polynomial degree of approximation	3
c_{ex}	Exact solution	$\frac{1}{4}\big(\cos(2\pi x)\cos(2\pi y) + 2\big)e^{-t}$
τ	Time step	$0.5, 0.25, 0.125$
T	Final time	2
ν	BDF order	1

Table 6: Setup of Test Case 4

This test is not designed to make the scheme fail, but rather to make a first comparison of the computational cost required by the different methods.
We can readily make a few observations from the results in Tables 7, 8 and 9. First of all, the quantities shown remain consistent regardless of the time step choice. This means that if, when changing the time step, a quantity increases, decreases or stays the same for a method, it does so for every other method as well. The **Warm-Start** method takes approximately the same time as **Newton**'s method for large time steps. The number of iterations increases consistently with the number of fixed point iterations and relaxed **Newton** iterations. The **Alternating** method is **by far** the slowest.

$\Delta t = 0.5$	**Newton**	**Warm-Start**	**Alternating**
# of Iterations	8	12.5	18.25
Elapsed Time (s)	12.64	12.87	30.01
Average Time (s)	1.5781	1.0981	1.2612
Fixed Point (%)	0	12.8	76.2
Newton (%)	100	87.2	23.8

Table 7: Comparison of different iterative method computational cost for Test Case 4, $\Delta t = 0.5$

$\Delta t = 0.25$	**Newton**	**Warm-Start**	**Alternating**
# of Iterations	8.75	13.63	18.88
Elapsed Time (s)	13.73	17.49	32.0446
Average Time (s)	1.572	1.3329	1.2534
Fixed Point (%)	0	12.9	76.3
Newton (%)	100	87.1	23.7

Table 8: Comparison of different iterative method computational cost for Test Case 4, $\Delta t = 0.25$

$\Delta t = 0.125$	**Newton**	**Warm-Start**	**Alternating**
# of Iterations	9.44	14.13	19.5
Elapsed Time (s)	15.12	19.07	33.2987
Average Time (s)	1.602	1.41	1.2823
Fixed Point (%)	0	12.0	76.1
Newton (%)	100	88.0	23.8

Table 9: Comparison of different iterative method computational cost for Test Case 4, $\Delta t = 0.125$

5.2.2 Test Case 5

The next test is meant to test the capacity of the methods with a large time step. In the test, I varied both the time step and the polynomial degree. The setup is described in Table 10. The test was performed for $\ell = 1, ..., 5$ but all tests with $\ell \geq 4$ failed to converge, so they will not be featured in the results. Moreover, testing on orders of **BDF** higher than 2 produced the same results as **BDF2**, so only **BDF1** and **BDF2** will be shown. The results are shown in Tables 11 and 12. The test shows clearly that **Warm-Start** is a more robust alternative to **Newton**, especially for low polynomial degrees and low orders of BDF. Moreover, since it performs a few more iterations with respect to **Newton**, "stabilizing" the initial guess is typically not very expensive. On the other hand, **Alternating** turns out to be slightly more robust than classical **Newton**, but definitely worse than **Warm-Start**. Nonetheless, its cost turns out to be completely unfeasible, especially when compared to the small advantages. The reasons for this are probably to be found in the same phenomenon described in Table 5. Due to its inefficiency, I did not test the **Alternating** method further.

5.2.3 Test Case 6

In this test, I wanted to assess the robustness of the **Warm-Start** method with respect to a bad initial guess. In order to do so, I chose to perturb the initial datum using a parameter ε_0 such that

$$w_{0,\varepsilon} = w_0 + \varepsilon_0 v$$

with v a random vector such that

$$\|w_0\|_{L^2} = \|v\|_{L^2}$$

Parameter	Name	Value
Ω	Domain	$(0,3)\times(0,1)$
$\mathbf{D}$	Diffusion tensor	$10^{-3}\mathbb{I}_2$
`Ntol`	Newton stopping criterion	10^{-10}
`FPtol`	Fixed point stopping criterion	10^{-10}
`WSit`	Fixed point Warm-Start iterations	10
`Altit`	Alternating fixed point iterations	3
N_{el}	Number of elements	200
ℓ	Polynomial degree of approximation	$1,2,3,4,5$
c_{ex}	Exact solution	$\frac{1}{4}\left(1+\tanh\left(8-\sqrt{\frac{\alpha}{24\,d_{\text{ext}}}}\,\xi\right)\right)^2$
τ	Time step	$0.5, 0.25$
T	Final time	2
ν	BDF order	$1,2$

Table 10: Setup of Test Case 5

$\Delta t = 0.25$	$BDF1$			$BDF2$		
ℓ	**Newton**	**WS**	**Alternating**	**Newton**	**WS**	**Alternating**
1	7.2	13.1	36.7	6.5	17.1	39.8
2	—	14.5	36	8.3	19.4	36
3	—	—	—	11.8	23.3	35.6

Table 11: Comparison of different iterative method average number of iterations in Test Case 5, $\Delta t = 0.25$

The setup is described in Table 13.
From the results in Tables 14 and 15, we can conclude that **Warm-Start** is more robust with respect to **Newton** when we have a worse initial guess. First of all, all test were performed both with 3 and 10 initial **Richardson** iterations, yielding the same results each time (except for the number of iterations), so 14 and 15 feature only the tests done with 3 **Richardson** iterations. From the tests that I have run, we can observe that both methods have a radius of convergence for "good" initial guesses. This is not determined uniquely; it depends mostly on the polynomial degree and the number of elements, which implies that it depends on the number of degrees of freedom as well. For small perturbations $(0.05, 0.1)$, the results are very similar between the two schemes. On the other hand, the **Warm-Start** can withstand more perturbed guesses $(0.2, 0.5)$, but only for low degree polynomials.
Both methods turn out to be very sensitive to the choice of a good initial guess, especially for a high number of degrees of freedom. Interestingly, increasing the perturbation does not usually imply an increase in the number of iterations. This may suggest that a few iterations are usually more than enough to move from a bad guess to a correct one. In the case of $N = 200$, the scheme failed for $\ell \geq 4$ even without any perturbation. However, the blow-up did not occur in a few iterations, but after 16 iterations. This shows that **Warm-Start** can fix a bad initial guess, but not errors that occur in the long run.

5.2.4 Test Case 7

In this test, I wanted to perform a comparison of the performance of both methods when faced with a long runtime. The setup is shown in Table 16. The purpose of this test is not to challenge the convergence of the two schemes, but rather to see how they deal with error accumulation.
The results in Table 17 show clearly that for a long time simulation, the **Warm-Start** method gives no advantage whatsoever. This can be explained by the fact that the final result of a **Warm-Start** iteration should be equivalent to the result of **Newton**'s method with a good initial guess, so there is no difference in error accumulation and for a long time the two methods eventually converge to the same result.

5.2.5 Brain Mesh Test

This last test, I wanted to test the capacities of the **Warm-Start** method with respect to a real case scenario. I chose to perform the same test as in [1], with the objective of simulating the spreading of α-synuclein, modeled

$\Delta t = 0.5$	$BDF1$			$BDF2$		
ℓ	**Newton**	**WS**	**Alternating**	**Newton**	**WS**	**Alternating**
1	9.3	12.5	38	8.3	11.8	37.5
2	—	14.5	—	—	—	—
3	—	—	—	—	—	—

Table 12: Comparison of different iterative method average number of iterations in Test Case 5, $\Delta t = 0.5$

Parameter	**Name**	**Value**
Ω	Domain	$(0,3) \times (0,1)$
$\mathbf{D}$	Diffusion tensor	$10^{-3}\mathbb{I}_2$
`Ntol`	Newton stopping criterion	10^{-10}
`FPtol`	Fixed point stopping criterion	10^{-10}
`WSit`	Fixed point Warm-Start iterations	$3, 10$
ε_0	Perturbation	$0.05, 0.1, 0.2, 0.5$
N_{el}	Number of elements	$50, 200$
ℓ	Polynomial degree of approximation	$1, 2, 3, 4, 5$
c_{ex}	Exact solution	$\frac{1}{4}\left(1 + \tanh\left(8 - \sqrt{\frac{\alpha}{24\, d_{\text{ext}}}}\,\xi\right)\right)^2$
τ	Time step	10^{-1}
T	Final time	1
ν	BDF order	1

Table 13: Setup of Test Case 6

$N = 50$	$\varepsilon_0 = 5 \times 10^{-2}$		$\varepsilon_0 = 10^{-1}$		$\varepsilon_0 = 2 \times 10^{-1}$		$\varepsilon_0 = 5 \times 10^{-1}$	
ℓ	**Newton**	**WS**	**Newton**	**WS**	**Newton**	**WS**	**Newton**	**WS**
1	6.3	11.0	6.3	11.1	—	11.2	—	18.6
2	6.4	10.7	6.5	10.7	—	11.0	—	—
3	9.0	17.5	—	—	—	—	—	—
4	10.2	15.0	—	—	—	—	—	—

Table 14: Comparison of Newton and Warm-Start number of iterations on mesh with $N = 50$, for varying initial perturbation amplitude ε_0 for Test Case 6

$N = 200$	$\varepsilon_0 = 5 \times 10^{-2}$		$\varepsilon_0 = 10^{-1}$		$\varepsilon_0 = 2 \times 10^{-1}$		$\varepsilon_0 = 5 \times 10^{-1}$	
ℓ	**Newton**	**WS**	**Newton**	**WS**	**Newton**	**WS**	**Newton**	**WS**
1	6.1	10.1	6.2	10.1	—	10.1	—	10.6
2	9.1	13.4	10.8	13.6	—	15.2	—	—
3	8.9	13.3	—	13.1	—	—	—	—
4	—	—	—	—	—	—	—	—

Table 15: Comparison of Newton and Warm-Start number of iterations on mesh with $N = 200$, for varying initial perturbation amplitude ε_0 for Test Case 6

by the FKPP equation. The brain geometry was segmented starting from a structural Magnetic Resonance Image of a brain from the OASIS-3 database [11]. The final mesh is a polytopal mesh consisting of 534 elements constructed by agglomerating the elements of the initial triangular mesh. The setup is described in Table 18 With both methods, the algorithm didn't succeed to reach convergence for $t = 25$, but it diverged at $t = 3.95$. However, this prompted some important observations about the **Warm-Start** method. First of all, if the number of **Richardson** iterations is kept fixed, the solution diverges much earlier. This is due to the fact that, even though the residual is descending, when it switches with **Newton**'s method, it uses an initial guess with a lower residue, but not necessarily a better one. This kind of behavior didn't show up in previous test, probably due to the fact that they weren't done on complex geometries or with too much high conditioning numbers. Secondly, the blow up occurred, in any case, after some time, and abruptly, meaning that in this case we are

Parameter	Name	Value
Ω	Domain	$(0,3)\times(0,1)$
$\mathbf{D}$	Diffusion tensor	$10^{-3}\mathbb{I}_2$
`Ntol`	Newton stopping criterion	10^{-10}
`FPtol`	Fixed point stopping criterion	10^{-10}
`WSit`	Fixed point Warm-Start iterations	3
N_{el}	Number of elements	$50, 200$
ℓ	Polynomial degree of approximation	$1, 2, 3, 4, 5$
c_{ex}	Exact solution	$\frac{1}{4}\left(1+\tanh\left(8-\sqrt{\frac{\alpha}{24\,d_{\text{ext}}}}\,\xi\right)\right)^2$
τ	Time step	2.5×10^{-2}
T	Final time	10
ν	BDF order	1

Table 16: Setup of Test Case 7

	$N=50$		$N=200$	
ℓ	**Newton**	**WS**	**Newton**	**WS**
1	4.0	8.4	4.0	8.4
2	7.0	11.0	4.2	9.0
3	5.3	12.2	4.8	9.2
4	5.3	10.2	7.9	11.4
5	9.0	12.9	—	—

Table 17: Comparison of Newton and Warm-Start number of iterations on Mesh with $N = 50, 200$ for Test Case 7

Parameter	Name	Value
Ω	Domain	Brain Mesh
$\mathbf{D}_{ext}$	Diffusion tensor diagonal component	$8\cdot10^{-5}$
$\mathbf{D}_{xy}$	Diffusion tensor xy axonal component	$8\cdot10^{-6}$
$\mathbf{D}_{yx}$	Diffusion tensor yx axonal component	0
η_0	Penalty coefficient	2
`Ntol`	Newton stopping criterion	10^{-6}
`FPtol`	Fixed point stopping criterion	10^{-6}
`iWSit`	Fixed Point Warm-Start terations	3
N_{el}	Number of elements	534
ℓ	Polynomial degree of approximation	1
τ	Time step	2.5×10^{-2}
T	Final time	25
ν	BDF order	6

Table 18: Setup of Brain Mesh Test

not dealing with a "bad" initial guess, but rather with a very ill-conditioned problem or probably the code struggles with some numerical artifact that shows up at $t = 3.95$. The poor performance in **"Warm-Start"** is generally associated with a relative improvement that is too close to 1. For this reason, only in this case, the code switches to **Newton**'s method if $\rho_n > 0.9$. We are presented once again with the limitations of the method, namely the fact that it turns out to be practically useful only for problems in which the initial guess does not allow **Newton** to converge.

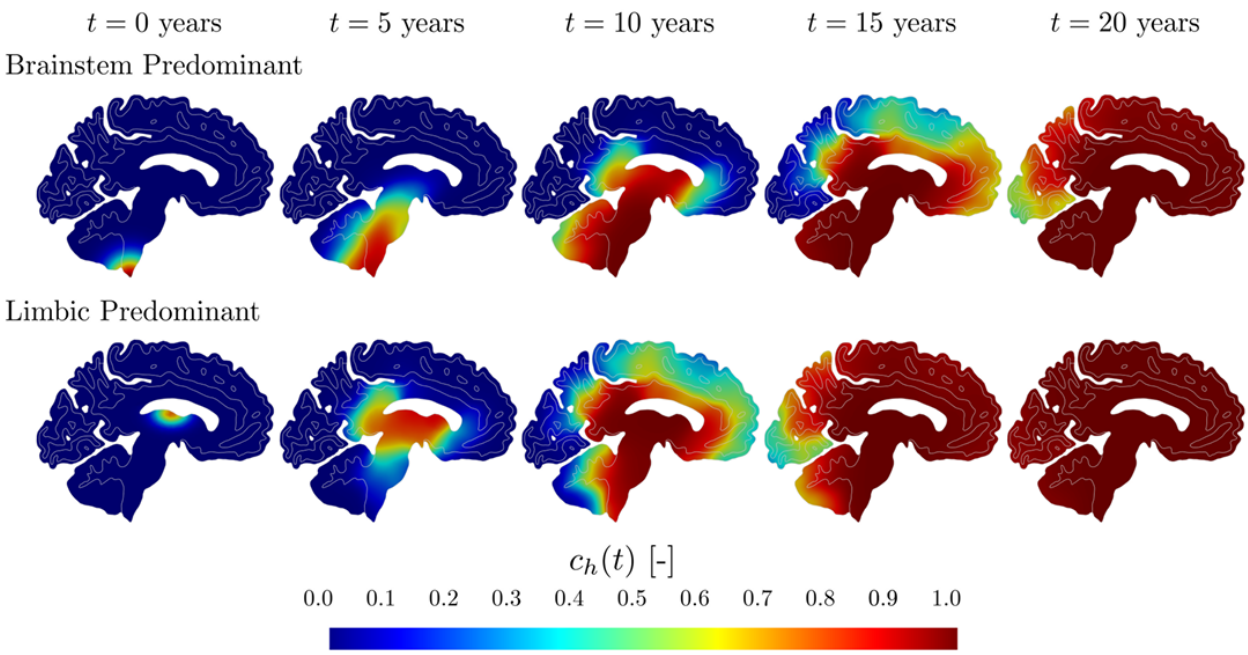


Figure 11: Brain Mesh Test: Patterns of α-synuclein concentration at different stages of the pathology

6. Conclusions

In this thesis, we explored the structure-preserving BDF–LDG for the FKPP equation, with a focus on two aspects that were left open in the original work: the choice of the entropy transformation and the robustness of the iterative solver. Regarding the entropy transformations, our numerical experiments show that the theoretical validity of a transformation is a necessary but not sufficient condition for practical applicability. The Shannon-Boltzmann function proved to be a feasible alternative to the standard Fermi-Dirac transformation, preserving optimal convergence rates, though at the cost of a systematically larger error. This degradation is attributable to the unboundedness of the exponential, which, while not threatening convergence due to the stability of the equilibrium $c = 1$, does introduce a quantitative penalization in accuracy. The Hellinger transformation, despite satisfying all theoretical requirements, failed consistently on traveling wave solutions. The failure mechanism was identified as excessive numerical diffusion near the sharp front, induced by the exploding second derivative of the entropy function as $c \to 0$, which reduces the sensitivity of the scheme in the most critical region of the solution. This result highlights that the asymptotic behavior of the entropy transformation near the equilibria plays a decisive role in practice, a consideration that is not captured by the theoretical assumptions alone. The k-Fermi-Dirac family confirmed this picture: the performance of the scheme depends jointly on the steepness of the transformation at $w = 0$ and on its asymptotic properties, and no single value of k proved uniformly superior across all mesh sizes. Regarding the iterative methods, the Warm-Start approach emerged as the most effective alternative to classical Newton's method. By preceding the Newton iterations with a small number of Richardson steps and incorporating a relaxed transition phase, the method significantly improves robustness in challenging configurations, particularly for large time steps and low BDF orders; at a modest additional computational cost. The relaxation phase proved essential: without it, Newton's method disregards the improvement provided by the Richardson pre-iterations and can produce a worse iterate than the starting point. The Alternating Newton method, by contrast, offered only marginal robustness gains over classical Newton while incurring a substantially higher computational cost, making it impractical for this class of problems. Both Newton and Warm-Start remain sensitive to the polynomial degree, and do not provide a reliable remedy when the solver fails due to error accumulation over long time horizons rather than a poor initial guess. In the long-time simulation test, the Warm-Start provided no advantage over Newton, confirming that its benefit is localized to the initial transient phase of each time step. On realistic geometries, it was proven that the Warm-Start is more robust alternative to Newton's method, but its possibilities are limited. The algorithm is structured in such a way that it is slightly more expensive than classical Newton, which makes it feasible at all times. Overall, the results suggest that the Fermi-Dirac transformation remains the most robust and accurate choice within the framework of [10], and that the Warm-Start method represents a practically valuable improvement to Newton's iteration, although its use does not guarantee better results at all times. Future work could investigate the extension of these methods to three-dimensional brain geometries, the development of adaptive strategies

for the Richardson parameter δ, and a more systematic theoretical analysis of the basin of convergence of the Warm-Start or Newton's method.

References

[1] P.F. Antonietti, M. Corti, S. Gómez, and I. Perugia. A Structure-Preserving LDG Discretization of the Fisher–Kolmogorov Equation for Modeling Neurodegenerative Diseases. *Mathematics and Computers in Simulation*, 241:351–366, 2026.

[2] P.F. Antonietti et al. Numerical Simulation of Protein Diffusion and Aggregation in Neurodegenerative Diseases with the Discontinuous Galerkin Method. 2019.

[3] F. Bonizzoni, M. Braukhoff, A. Jüngel, and I. Perugia. A Structure-Preserving Discontinuous Galerkin Scheme for the Fisher–KPP Equation. *Numerische Mathematik*, 146(1):119–157, 2020.

[4] R.A. Fisher. The Wave of Advance of Advantageous Genes. *Annals of Eugenics*, 7(4):355–369, 1937.

[5] A.D. Grigoriev et al. Boundary Element Method for Diffusion–Reaction Problems in Brain Imaging. 2020.

[6] S. Gómez, A. Jüngel, and I. Perugia. Structure-Preserving Local Discontinuous Galerkin Method for Nonlinear Cross-Diffusion Systems. *arXiv preprint*, 2025.

[7] E. Hairer and C. Lubich. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. Springer, 2006.

[8] W. Heid and T.P. Wihler. Adaptive Iterative Linearisation Galerkin Method for Nonlinear Problems. *Journal of Scientific Computing*, 2019.

[9] M. Jucker and L.C. Walker. Self-Propagation of Pathogenic Protein Aggregates in Neurodegenerative Diseases. *Nature*, 501(7465):45–51, 2013.

[10] A. Jüngel. The Boundedness-by-Entropy Method for Cross-Diffusion Systems. *Nonlinearity*, 28(6):1963–1984, 2015.

[11] P.J. LaMontagne, T.LS. Benzinger, J.C. Morris, S. Keefe, R. Hornbeck, C. Xiong, E. Grant, J. Hassenstab, K. Moulder, A.G. Vlassenko, M.E. Raichle, C. Cruchaga, and D. Marcus. OASIS-3: Longitudinal Neuroimaging, Clinical, and Cognitive Dataset for Normal Aging and Alzheimer Disease. *medRxiv*, 2019.

[12] H.G. Pemberton, L.E. Collij, F. Heeman, A. Bollack, M. Shekari, G. Salvado, I.L. Alves, D.V. Garcia, M. Battle, C. Buckley, A.W. Stephens, S. Bullich, V. Garibotto, F. Barkhof, J. D. Gispert, G. Farrar, and AMYPAD consortium. Quantification of Amyloid PET for Future Clinical Use: A State-of-the-Art Review. *European Journal of Nuclear Medicine and Molecular Imaging*, 49(10):3508–3528, 2022.

[13] S.B. Prusiner. Biology and Genetics of Prions Causing Neurodegeneration. *Annual Review of Genetics*, 47:601–623, 2013.

[14] C. Reiter. Easy Algorithms for Finding Eigenvalues. *Mathematics Magazine*, 63(3):173–178, 1990.

[15] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, 2 edition, 2003.

[16] S. Salsa. *Partial Differential Equations in Action: From Modeling to Theory*. Springer, 3 edition, 2016.

[17] R. Schneider, Y. Xu, and A. Zhou. An Analysis of Discontinuous Galerkin Methods for Elliptic Problems. *Advances in Computational Mathematics*, 25(1):259–286, 2006.

[18] L. Stefanis. α-Synuclein in Parkinson's Disease. *Cold Spring Harbor Perspectives in Medicine*, 2(2):a009399, 2012.

[19] T.B. Thompson et al. Protein Aggregation and Spreading in Neurodegenerative Diseases: Discrete and Continuous Models. *Journal of the Mechanics and Physics of Solids*, 2017.

[20] L.C. Walker and M. Jucker. Neurodegenerative Diseases: Expanding the Prion Concept. *Annual Review of Neuroscience*, 38:87–103, 2015.

[21] J. Weickenmeier, M. Jucker, A. Goriely, and E. Kuhl. A Physics-Based Model Explains the Prion-Like Spreading of Alzheimer's Disease. *Proceedings of the National Academy of Sciences*, 116(11):4671–4676, 2019.

Abstract in lingua italiana

L'equazione di Fisher–Kolmogorov–Petrovsky–Piskunov è un'equazione alle derivate parziali ampiamente utilizzata per modellare fenomeni che coinvolgono meccanismi di reazione–diffusione. Una delle sue possibili applicazioni è la modellazione delle malattie neurodegenerative, quali il morbo di Alzheimer e il morbo di Parkinson, che sono caratterizzate dalla propagazione di proteine mal ripiegate attraverso il tessuto cerebrale.
Questa tesi descrive un framework Local Discontinuous Galerkin a preservazione di struttura per l'approssimazione numerica dell'equazione di Fisher–Kolmogorov–Petrovsky–Piskunov, basato su una riformulazione in variabili entropiche al fine di garantire che la soluzione discreta soddisfi i vincoli appropriati in ogni istante di tempo.
Vengono analizzati due aspetti del framework. In primo luogo, si studia l'influenza della trasformazione entropica sull'accuratezza e sulla robustezza dello schema, considerando come alternative alla trasformazione standard di Fermi–Dirac le funzioni di Shannon–Boltzmann, di Hellinger e la famiglia k-Fermi–Dirac. Gli esperimenti numerici mostrano che l'ammissibilità teorica non è sufficiente a garantire buone prestazioni pratiche. In particolare, la trasformazione di Hellinger fallisce sistematicamente nel caso di soluzioni con fronti ripidi a causa di un'eccessiva diffusione numerica, mentre la trasformazione entropica di Shannon–Boltzmann converge con ordini ottimali ma produce errori più elevati. La famiglia k-Fermi–Dirac evidenzia che le prestazioni dipendono congiuntamente dalla ripidità della trasformazione in prossimità di $w = 0$ e dal suo comportamento asintotico.
In secondo luogo, si affronta la sensibilità del metodo di Newton rispetto alla scelta dell'approssimazione iniziale, proponendo e testando un approccio iterativo di tipo "Warm-start", che precede le iterazioni di Newton con un numero ridotto di passi di Richardson e una fase di transizione rilassata. I test numerici dimostrano che tale metodo migliora significativamente la robustezza in regimi problematici, in particolare per passi temporali grandi e bassi gradi polinomiali, a fronte di un modesto costo computazionale aggiuntivo, mentre non offre vantaggi nelle simulazioni a lungo termine, nelle quali domina l'accumulo degli errori.

Parole chiave: equazione di Fisher–Kolmogorov, metodi structure-preserving, Local Discontinuous Galerkin, trasformazione entropica, metodi iterativi, malattie neurodegenerative.

Acknowledgements

I would like to sincerely thank my supervisor, Paola F. Antonietti, for proposing this project, for giving me the opportunity to work on it in Vienna, for her genuine kindness, and for the trust she placed in my abilities from the very first moment.
I would also like to deeply thank Prof. Perugia for hosting me in Vienna and for making me feel welcome at all times, for her understanding, and for the support she showed throughout the work and after.
Lastly, I would like to profoundly thank Mattia Corti for following my journey, for his invaluable help, and for the support and esteem he showed throughout the entire project.
The brain MRI images were provided by OASIS-3: Longitudinal Multimodal Neuroimaging: Principal Investigators: T. Benzinger, D. Marcus, J. Morris; NIH P30 AG066444, P50 AG00561, P30 NS09857781, P01 AG026276, P01 AG003991, R01 AG043434, UL1 TR000448, R01 EB009352. AV-45 doses were provided by Avid Radiopharmaceuticals, a wholly-owned subsidiary of Eli Lilly.