Master in High Performance Computing

Cosmic Ray Propagation with High Dimensional Finite Element Method

Supervisor(s):
Luca Heltai,
Alberto Sartori,
Piero Ullio

Candidate:
Jiaxin Wang

5th edition
2018–2019

Abstract

Cosmic Ray Propagation with High Dimensional Finite Element Method

The Galactic synchrotron emission contains abundant physics related to not only the Galactic magnetized interstellar medium but also has prominent effect on understanding the Cosmic microwave background especially the B-mode polarization. To catch up with the growing precision in astrophysical observations, we need to build a consistent numerical framework where simulating the cosmic ray (electron) propagation is a major task. In the master project, we propose to use the finite element method for solving cosmic ray (electron) transport equation within the phase-space of dimension varying from two to six. The numeric package `BIFET` is developed on top of the `deal.II` library with support in the adaptive mesh refinement. We mainly introduce the design and methods in solving advection-diffusion problems and demonstrate its capability and precision with physically simplified tests and examples.

Acknowledgment

My journey through the four years in physics, computing science and Italy has been very pleasant. I am grateful to the PhD program in the astro-particle physics group of SISSA and the high-performance-computing master program conducted jointly by SISSA and ICTP, from which I have gained lots of knowledge and skills. As my PhD supervisors, Prof. Piero Ullio and Prof. Francesca Perrotta have shown me how to do science professionally. I have been inspired by the amazing physical intuition and strict attitude of Prof. Ullio, and started to have a better understanding of the path from an ignorant pupil to an independent scientist.

The PhD project cannot be carried out smoothly without the master program for high-performance-computing, from which I have been trained to be a decent programmer competent for numerical scientific tasks. I have to thank Prof. Luca Heltai who taught me the numerical analysis, the experience of working with him has always been very joyful and inspiring. I cannot imagine how could I survive the intensive training of the master program without the help from Dr. Alberto Sartori, who has always been a great friend and indispensable support for all the students in the master program.

It was a great fortune of knowing Prof. François Boulanger, who introduced and recommended me to the IMAGINE consortium, and from where I met Prof. Tess R. Jaffe, Prof. Torsten A. Enßlin, Prof. Anvar Shukurov, Prof. Marijke Haverkorn and Dr. Theo Steininger. I do not think I could have been even more fortunate than collaborating with Prof. Jaffe and Prof. Enßlin.

I am grateful to all my friends, office mates and colleagues in the theoretical-particle, astro-particle and astrophysical physics groups in SISSA, and from other institutes. Finally I would like to thank my parents and family for their unconditional support and love.

Abbreviations

CR	cosmic ray
CRE	cosmic ray electron/positron
CMB(R)	cosmic microwave background (radiation)
DM	dark matter
FFT	fast Fourier transform
FE	free/thermal electron
FEM	finite element method
GMF	Galactic magnetic field
UHECR	ultra high energetic cosmic ray
ISM	interstellar medium
ISRF	interstellar radiation field
LoS	line of sight
MHD	magneto hydrodynamics
MPI	message passing interface
PDE	partial differential equation
RMHD	radiation magneto hydrodynamics
SNR	supernova remnant

Contents

Abstract iii

Acknowledgment v

Abbreviations vii

Contents ix

Introduction 1

1 Underlying Theory 5
- 1.1 Cosmic Ray Electron Transport . . . 5
- 1.2 Finite Element Method . . . 7
- 1.3 Domain Separation . . . 9

2 BIFET 13
- 2.1 Software Design . . . 13
  - 2.1.1 Domains . . . 15
  - 2.1.2 Sparsity Pattern . . . 16
  - 2.1.3 System Assembling . . . 17
  - 2.1.4 multi-threading support . . . 20
- 2.2 Precision and Performance . . . 21
  - 2.2.1 Performance . . . 21
  - 2.2.2 Precision . . . 23

3 Application Examples 35

Conclusion 39

Bibliography 41

Introduction

In many previous studies about the Galactic synchrotron emission, it was considered convenient to model the cosmic-ray electron [1] (CRE) distribution independently from the structure of the Galactic magnetic field (GMF). This approach is theoretically inconsistent, as we know that the CRE distribution is not physically independent from the magnetic field distribution/configuration. Following the quasi-linear test particle approach [15] in describing the CR propagation through the highly conductive magnetized plasma, the anisotropic spatial and spectral diffusion coefficients are dictated by the specific shape of magnetic turbulence. Although it can be argued that for the independent modelling we focus only on the phenomenological description. With given sufficient astrophysical measurements and appropriate analysis it is always possible to achieve a proper understanding within which the CRE and magnetic field distribution are consistent. Nevertheless, we need to point out that the phenomenological modelling is eventually not helping us in understanding more detailed physical mechanisms. For example, the Galactic synchrotron emission at 10 GHz level mainly [2] results from the CRE with energy roughly around 5 GeV to 30 GeV given the magnetic field strength not stronger than 10 μG and not weaker than 2 μG. Comparing the energy loss time scale to the typical diffusion time scale, one finds that the bulk of the electrons which are contributing to the 10 GHz level synchrotron emissivity in the observer's neighbourhood is mainly from sources (e.g., supernova remnants, pulsars) near the neighbourhood, instead of from faraway sources. Thus it becomes interesting that by analyzing the Galactic synchrotron emission at various frequency and LoS direction we should be able to study the properties of the CR sources, propagation patterns and the magnetic field configurations. This can only be accomplished by including a CRE propagation simulation inside the pipeline of generating synchrotron emission maps, where the synchrotron emissivity we observe is calculated with exactly the same magnetic field as estimating the energy loss of CRE propagation. And so instead of starting with an independent parametric description of CRE phase-space distribution, the con-

[1] By mentioning cosmic-ray electron (CRE) in this dissertation, we actually intend to include both electron and positron, knowing that the energetic positron to electron ratio is less than 0.25 in the Galaxy. The Galactic positrons are largely secondary, which means that they are not directly accelerated/ejected from astrophysical sources but from the interactions between primary CR particles with the ISM. But this "chemical" difference is ignored since we focus more on their kinematics in the magnetized ISM.

[2] According to the Fig. **??**, the CRE emissivity peaks at certain energy scale corresponds to the specific observational frequency and the ambient magnetic field strength. Softer CRE spectral or higher observational frequency pushes the peaking position to higher energy, while on the contrary, stronger magnetic field strength tends to pull the peak to lower energy.

sistent approach requires physical modellings of the interaction between the energetic CRs and the thermal component in the ISM (e.g., ionizing the neutral atoms, scattering with charged particles and low energy photons) and the magnetic turbulence (e.g., being scattered off by magnetic turbulence, amplifying magnetic turbulence through streaming instability).

The necessity of this consistent picture for CR studies was recently pointed out by Blasi et al. [1] and Evoli et al. [4], where they studied the phenomenon of CR being scattered off by magnetic turbulence which in turn is amplified by the CR streaming (in the perturbative regime) before reaching a saturation. The results of their studies pointed out that the interactive picture can explain not only the observed spectral break but also the vertical scale of the CR diffusion region. And such consistent approach is also required in studies which focus on the Galactic ISM, where the CR streaming [11] is important for launching of galactic winds [5, 14], the Parker instability [7] and the multi-phase medium [19]. Since our research scope is narrowed only to the relativistic electron/positron distribution which has dominant contribution to the synchrotron energy loss (since the protons have a synchrotron lifetime of $(m_{\rm p}/m_{\rm e})^4$ times longer than electron and they may lose their energy via other mechanisms without emitting much synchrotron radiation [13]), we do not have to fully resolve the interactive system that is mainly related to CR protons, nor we need to consider the combination of test-particle and test-wave (which describes how the magnetic turbulence responds to the CR flow) approaches. The minimal consistency required for simulating the Galactic synchrotron emission only has to ensure the synchrotron emission we observe is exactly from the synchrotron energy loss during CRE propagation.

To achieve the minimal consistency, we need to numerically solve the CRE transport equation. Numerical packages for simulating CR (not only for electron/positron) propagation have been developed since two decades ago, among which the most popular one is `Galprop` by Strong and Moskalenko [16] where the CR transport equation is solved with the finite-difference method. More recently Evoli et al. [3] released the `DRAGON` package with a similar solver as `Galprop` but support for 3D anisotropic modelling of the CR spatial diffusion. We have also witnessed other numerical attempts with finite-volume method or modified finite-difference method, but unfortunately no open access has been provided to their numerical work [8].

It is well known that for solving a partial differential equation (PDE) or a set of PDEs, there are generally three categories of numerical approaches: finite difference, finite volume and finite element methods. Each method has its particular advantages and disadvantages, while with appropriate numerical techniques they are all suitable for simulating CR propagation with similar precision. Practically, physicists need well developed numerical libraries with minimal programming requirements in modifying the back-end functions for various simulation tasks. Particularly for simulating the CR propagation, we haven't seen any open-source numerical work with the finite-element method in the community mainly because of the complexity in implementing this method from scratch. Besides, there is no package that can provide us a proper discretization beyond 3D with adaptively refined mesh. With such motivations we introduce `BIFET`, the toolkit for solving PDEs in a domain with up-to six dimensions (not including the time coordinate) based on the `deal.II` library (an open source finite element library designed to provide well-documented tools to build finite element codes

for a broad variety of PDEs). This numerical tool can help us efficiently resolve an isotropic phase-space distribution defined within a very generic domain.

Chapter 1

Underlying Theory

1.1 Cosmic Ray Electron Transport

Cosmic rays are referring to the relativistic, generally with energy larger than 1 MeV, charged particles of various species. The Galactic cosmic rays are mainly categorized by a primary component, including proton, electron, helium, carbon, etc., which can be synthesized in the stars, and the secondary [1] component including antiprotons, borons, etc., that are mainly produced during the CR traversing though the Galactic ISM.

It is known that CRs have frequent interactions with the magnetic turbulence, interstellar photons and thermal particles, and because of which the averaged life time for a CR particle staying the the Galaxy is roughly around 10^6 years. The motion of a single charged particle can be well predicted if the ambient magnetic environment is known. Some numerical simulators, e.g., CRPropa [**?**], follow this idea of calculation, which is precise and convenient for studying static magnetic field structure or tracing the properties of the CR motion within a specifically designed field. Another approach (known as the test particle approach) is to treat the CRs as either an uniform or composite fluid, and in turn the motion of each single particle is not traceable anymore. In this way we approximate the collective behavior of CRs by the Fokker-Planck equations, which take ensemble average of linearly perturbed description of the system consists of the electromagnetic field and charged plasma. The quasi-linear approximation was proved to be acceptable even the magnetic perturbation is four times larger than the regular field strength [15], which we consider is enough for studying the Galaxy. In the following we will use the test particle approach and build the numerical solving routine.

The up-to-date understandings towards the observed features of CRs are well reviewed recently by Grenier et al. [6], Strong et al. [17], Tanabashi et al. [18] and the references there in. The general trend of the frontier studies of CRs has been pushed forward to the detailed interaction between CRs and other Galactic components like the ionized gas and magnetic field and thus to the consistent description of the Galactic ecology. The non-linear interactive picture which is meant to be simulated is our final aim, and as the first step we have to focus on building the efficient numerical framework

[1] Note that there exists a different primary and secondary definition which distinct cosmic rays observed above and below the earth's atmosphere.

with its performance well profiled for the near future studies of the minimal consistent scenario discussed above.

In describing the CRE propagation, we commonly start with the phase-space distribution $u_e(\mathbf{x}, \mathbf{q}, t)$ of energetic electrons[2] and approximate their propagation with a single transport equation mainly with physical terms like spatial and spectral diffusion (scattering off magnetic turbulence), advection (streaming with the bulk motion), spectral advection (re-acceleration and energy loss)

$$\begin{aligned} &\partial_t u_e - \nabla_{\mathbf{x}} \cdot (\mathcal{D}_{\mathbf{xx}}(\nabla_{\mathbf{x}} u_e)) - \nabla_{\mathbf{E}} \cdot (\mathcal{D}_{\mathbf{EE}}(\nabla_{\mathbf{E}} u_e)) \\ &+\nabla_{\mathbf{x}} \cdot (\mathbf{V} u_e) + \nabla_{\mathbf{E}} \cdot (\mathbf{b} u_e - \frac{1}{3}(\nabla_{\mathbf{x}} \cdot \mathbf{V}) u_e) = Q \ , \end{aligned} \tag{1.1}$$

where $\mathcal{D}_{\mathbf{xx/EE}}$ represents the spatial/spectral diffusion tensor, $\mathbf{V}$ represents the bulk motion of the CRs, $\mathbf{b}$ indicates continuous energy loss due to several mechanisms like synchrotron emission, inverse-Compton scattering, Coulomb scattering and ionizing ISM, thermal bremsstrahlung. The right-hand-side Q terms stands for astrophysical sources of energetic electrons/positrons.

In the energy loss, here we specify the mechanisms for electrons/positrons, which are slightly different from protons. The inverse Compton scattering describes how energetic electrons/positrons heat ISM photons and kick them to higher frequencies, where the ISM photon field is also known as the interstellar radiation fields (ISRFs, with "fields" for specifying the different components) which consists of various components like CMB photons, star light (covering ultra-violet and optical-inferred bands) and dust emission (mainly covering the inferred bands). Note that the ISRFs are not known purely from observations, but through modelling the radiative transfer [12] of emission and absorption processes in the ISM and tuned to match certain observables, where the dust density and temperature distribution is modelled. Although the Galactic dust emission is not studied in our current work, the future consistent analysis with polarized synchrotron and dust emission should be aware that the dust distribution is not independent from CRE.

In the simplest case we consider electrons and positrons as a single fluid, by doing so we ignore the secondary production of positrons like the decay of protons and heavier nuclei. A better treatment should involve at least protons/positrons and consequently the interaction between CRs and magnetic turbulence. The spatial and spectral diffusion coefficients $\mathcal{D}_{\mathbf{xx}}$ and $\mathcal{D}_{\mathbf{EE}}$ are often defined phenomenologically because of their complexity. The basic features of $\mathcal{D}_{\mathbf{xx/EE}}$ includes that they depend on the regular magnetic field orientation and turbulent amplitude and shape (according to the quasi-linear theory of CR transport). In the quasi-linear theory the diffusion coefficients can be analytically derived as the Fokker-Planck coefficients by solving the radiation-magneto-hydrodynamic (RMHD) system [15]. However the reality is more complicated, with theoretical and recent numerical studies [5, 9] the CR streaming velocity is not always confined to the Alfvén speed, but the decoupling of CRs to the cold ISM where the magnetic turbulence is damped can be modelled by increasing the spatial diffusion

[2] According to recent local measurements up to a few years ago and the standard energy loss of secondary positrons [**?**], the positron excess problem can very likely due to the primary component from nearby pulsars, and so in the following we treat positrons as primaries.

rate along the regular magnetic field orientation.

Although the (already simplified) CRE transport equation sounds complicated from the physical side of view, it can be understood conceptually no more than a non-linear advection-diffusion problem (it is still very non-trivial in practically solving such problem from the numerical side of view).

1.2 Finite Element Method

Here we intend to give a simple description of some important concepts in numerical analysis and especially which are involved in solving the CR transport equation with the finite element method. The basic concept is that the finite element method describes a continuous problem in its weak formulation (applying the Galerkin methods) and approximate solution in a finite functional space. For example consider a linear mapping $A : V \to V$ in a Hilbert space V, a problem is defined as $Au = f$ where u is the solution. Instead of solving $Au = f$ directly, the weak formulation seeks the solution with a test function $v \in V$ and convert the problem into

$$\langle Au, v\rangle = \langle f, v\rangle \ , \tag{1.2}$$

where $\langle\cdot,\cdot\rangle$ represents a bi-linear form (which in the applications here it indicates a domain integral). Then with a set of basis functions $\{\phi_i\} \subset V$ we try to describe $u = \sum_i \mathcal{U}_i\phi_i$ and consequently the weak formulation reads

$$\sum_j \mathcal{U}_j\langle A\phi_j, \phi_i\rangle = \langle f, \phi_i\rangle \ , \tag{1.3}$$

and the solution finding eventually becomes solving the linear algebra that represents the weak form above. Note that the above formulas are defined in the continuous domain. While for the discrete domain where the functional base is described with quadrature points, we use notation u_h for representing the discrete solution.

By the decomposition in the finite functional space in the discrete domain, the solution precision is largely determined by how we choose the functional basis and quadrature points, which are in principle independent from the finite element method itself. For example we can take the Gaussian quadrature which means with arbitrary n points $\{x_i\}$ and weights $\{w_i\}$ in one-dimensional domain $[a, b]$, an integral of function $g(x)$ can be approximated as

$$\int_a^b g(x)dx = \sum_i \epsilon_i g(x_i) \ , \tag{1.4}$$

$$\epsilon_i = \int_a^b \prod_{j\neq i} \frac{x - x_j}{x_i - x_j} dx \ , \tag{1.5}$$

which has a degree of precision at most $2n-1$. By applying the quadrature rule to the

weak formulation we can further write Eq. 1.3 as

$$\sum_j \mathcal{U}_j \sum_k \epsilon_k^2 A(x_k)\phi_j(x_k)\phi_i(x_k) = \sum_k \epsilon_k f(x_k)\phi_i(x_k) \ , \tag{1.6}$$

where the continuous integrals have been approximated by discrete summation, and consequently the solution u is approximated by its discrete counterpart $u_h(x_k) = \sum_i \mathcal{U}\phi_i(x_k)$.

The left hand side integral in Eq. 1.6 is not trivial as it appears. Here we illustrate a more realistic derivation with a one-dimensional diffusion problem, which reads

$$-\partial_x(\alpha\partial_x u) = f(x) \ . \tag{1.7}$$

For its weak formulation we define the functional base $\{\phi_i\}$ and the discrete solution u_h follows from:

$$-\sum_j \mathcal{U}_j \langle \phi_i, \partial_x(\alpha\partial_x\phi_j)\rangle_\Omega = \langle f, \phi_i\rangle_\Omega \ , \tag{1.8}$$

where the problem is defined within the domain Ω with boundary surface $\partial\Omega$. Integrating the left-hand-side by part, we arrive at

$$\sum_j \mathcal{U}_j \left[\langle \partial_x\phi_i, \alpha\partial_x\phi_j\rangle_\Omega - (\alpha\phi_i, \partial_x\phi_j \cdot \hat{n})_{\partial\Omega}\right] = \langle f, \phi_i\rangle_\Omega \ , \tag{1.9}$$

with $\hat{n}$ represents the direction of the boundary surface. In practice the partial derivation of base functions $\partial_x\phi_i$ are pre-defined as the functional base itself. It is also apparent that Eq. 1.9 is in principle a set of linear equations

$$\sum_j \mathcal{M}_{i,j}\mathcal{U}_j = \mathcal{R}_i \ , \tag{1.10}$$

where $\mathcal{M}$ is known as the left-hand-side system matrix, while $\mathcal{R}$ is the right-hand-side system vector. The boundary conditions we have not included in defining the strong formulation usually applies to the boundary integral presented above, where a strong boundary condition requires specific shape of ϕ_i or $\partial_x\phi_i$ at the boundary surface, whereas a nature boundary condition can simplify the integral with vanishing terms. Take the diffusion problem above for example, a strong boundary condition can be $u(x) = g$ for $x \in \partial\Omega$ and consequently the surface integral in Eq. 1.9 should be moved to the right-hand-side by replacing ϕ_i with g. While with a weak boundary condition we can ask $\partial_x u(x) = 0$ for $x \in \partial\Omega$, in which case the surface integral vanishes since $\partial_x\phi_j = 0$ and note that this requirement will not show up explicitly in solving the linear equations.

For some particular problems, e.g., the advection problem (or hyperbolic partial differential equation), we need extra caution with the discretization scheme. In practice for advection problems we use the upwind discontinuous Galerkin method, where the discontinuous means the functional basis is defined independently for each triangulated cell and so the solution u_h do not have to be continuous at the internal boundaries

between two neighbouring cells. Assuming a simple one-dimensional advection problem

$$\partial_x(\beta u) = f(x) \ , \tag{1.11}$$

and the plain weak formulation in functional base ϕ_m reads

$$-(\phi_m, u_h\beta\cdot\hat{n})_{\partial\Omega} + \sum_i \langle u_h, \beta\cdot\partial_x\phi_m\rangle_{T_i} = \sum_i \langle\phi_m, f\rangle_{T_i} \ , \tag{1.12}$$

where $\partial\Omega$ represents the external boundary surface, $T_i \in \mathbb{T}$ represents the volume for each triangulation cell i, the notation $(\cdot,\cdot)$ indicates surface integral while $\langle\cdot,\cdot\rangle$ for volume integral. Then on top which we apply the upwind scheme, which introduces extra internal surface integrals $+\sum_j (u_h^-, \beta\cdot[\phi_m\hat{n}])_{F_j}$ on the left-hand-side, where $F_j \in \mathbb{F}$ represents the internal surface j. $[\phi_m\hat{n}]$ is defined by $[\phi_m\hat{n}] = \phi_m^+\hat{n}^+ + \phi_m^-\hat{n}^-$ where the notation $+$ indicates the quantity in the upwind cell while $-$ for the downwind cell. Note that in the discrete Galerkin method, the functional basis is defined independently for each cell.

1.3 Domain Separation

The `deal.II` library provides triangulation methods for a domain with number of dimensions no higher than three, which is a common setting of a finite element method library, and so for problems defined within higher dimensions (e.g., a CR propagation problem with three spatial dimension and one spectral dimension) we cannot build the mathematical framework directly with its original library functions. To overcome this, we separate the full domain into a spatial sub-domain (denoting the spatial space $\mathbf{x}$) and a spectral sub-domain (denoting the energy/momentum space $\mathbf{q}$). By default the spatial and spectral sub-domains are constructed as hyper-rectangles. The notation $\mathbb{R}^{a+b}$ is defined for distinguishing different dimension settings, where "a" represents the number of dimensions in the spatial sub-domain while "b" represents that in the spectral sub-domain. For example $\mathbb{R}^{1+1}$ setting is built by $\{x_1, q_1\}$, while $\mathbb{R}^{2+1}$ setting is built by $\{x_1, x_2, q_1\}$.

Without any loss of generality, we assume an unspecified time-dependent problem in the form of

$$\partial_t u + \hat{\mathcal{O}}u = f, \tag{1.13}$$

in an arbitrary R^{a+b} dimension setting. The discretization in time can be approached by a sequence of time steps with solutions $u^n(\mathbf{x},\mathbf{q})$ marked by time step index n, i.e., the finite difference approach for the time discretization. In the Rothe's scheme we can rephrase the time-dependent problem as

$$\frac{u^n - u^{n-1}}{t_n - t_{n-1}} + ((1-\theta)\hat{\mathcal{O}}_{n-1}u^{n-1} + \theta\hat{\mathcal{O}}_n u^n) = (1-\theta)f_{n-1} + \theta f_n \ , \tag{1.14}$$

where θ varies within $[0,1]$. $\theta = 1$ and 0 represents implicit and explicit Euler method respectively, while $\theta = 0.5$ is the alleged Crank-Nicolson method. For a time-independent problem, a steady state solution can be found technically by a single

solving step within the θ-scheme we described above.

Intuitively, we express solution u^n inside the cross product Φ of two functional spaces as

$$\begin{aligned} u^n(\mathbf{x},\mathbf{q}) &= \sum_i \sum_\alpha \mathcal{U}^{\alpha i} v_i(\mathbf{x}) w_\alpha(\mathbf{q}) \\ &= \sum_{\alpha,i} \mathcal{U}^{\alpha i} \phi_{\alpha i}(\mathbf{x},\mathbf{q}) \ , \end{aligned} \tag{1.15}$$

where the base function spaces are mathematically defined by

$$\mathcal{V} := span\{v_i \in H^1(\mathbb{R}^a)\} \ , \tag{1.16}$$

$$\mathcal{W} := span\{w_\alpha \in H^1(\mathbb{R}^b)\} \ , \tag{1.17}$$

$$\Phi := span\{\phi_{\alpha i} \in \mathcal{V} \otimes \mathcal{W}\} \ . \tag{1.18}$$

Discretizing a PDE problem over quadrature points yields the weak formulation, where generally we can represent the left-hand-side operator $\hat{\mathcal{O}}$ by a sparse matrix $\mathcal{M}_{\mathbf{xq}}$. Whereas the right-hand-side terms can be assembled into a matrix representative $\mathcal{R}_{\mathbf{xq}}$, and in this way the generic weak formulation has the form

$$\mathcal{M}_{\mathbf{xq}} \cdot vec(\mathcal{U}) = vec(\mathcal{R}_{\mathbf{xq}}) \ . \tag{1.19}$$

The reason for vectorizing (with *vec* denoting the matrix vectorization operation) the solution matrix $\mathcal{U}$ and the right-hand-side matrix $\mathcal{R}$ can be understood via a simplified example. Suppose upon the solution representative $\mathcal{U}$ we apply two independent operations $\hat{\mathcal{O}}_{\mathbf{x}}$ and $\hat{\mathcal{O}}_{\mathbf{q}}$ which live separately in two sub-domains (to be specific, $\hat{\mathcal{O}}_{\mathbf{x}} \equiv \hat{\mathcal{O}}_{\mathbf{x}}(\mathbf{x})$ and $\hat{\mathcal{O}}_{\mathbf{q}} \equiv \hat{\mathcal{O}}_{\mathbf{q}}(\mathbf{q})$). It is thus straight forward to assemble matrix representatives $\mathcal{M}_{\mathbf{x}}$ and $\mathcal{M}_{\mathbf{q}}$ respectively, namely the mapping from the strong formulation to the weak formulation, which reads

$$\hat{\mathcal{O}}_{\mathbf{x}} \hat{\mathcal{O}}_{\mathbf{q}} u \to \mathcal{M}_{\mathbf{x}} \mathcal{U} \mathcal{M}_{\mathbf{q}}^T \ , \tag{1.20}$$

where $(\cdot)^T$ stands for matrix transpose. By default we associate the row indices to quadrature points in the spatial sub-domain. It is obvious at this point that solving a Sylvester-like equation requires a vectorization and consequently the final left-hand-side matrix reads $\mathcal{M}_{\mathbf{xq}} = \mathcal{M}_{\mathbf{q}} \otimes \mathcal{M}_{\mathbf{x}}$. And through this vectorization, we could also assemble $\mathcal{M}_{\mathbf{xq}}$ for even the most generic $\hat{\mathcal{O}}(\mathbf{x},\mathbf{q})$ with a quadrature-point-wise Kronecker product (represented by the symbol $\otimes$).

For physicists who are not very familiar with the finite element method (FEM), we feel obliged to illustrate explicitly the methodology behind assembling the $\mathcal{M}_{\mathbf{xq}}$ (the very same idea goes to assembling the $\mathcal{R}_{\mathbf{xq}}$). A typical example can be a pure spatial diffusion problem, where a strong formulation of the diffusion term (on the left-hand-side of a PDE) can be

$$-\nabla_{\mathbf{x}} \cdot (\mathcal{D}_{\mathbf{xx}} \nabla_{\mathbf{x}} u(\mathbf{x},\mathbf{q})) \ , \tag{1.21}$$

where $\mathcal{D}_{\mathbf{xx}} \equiv \mathcal{D}_{\mathbf{xx}}(\mathbf{x},\mathbf{q})$ represents the spatial diffusion tensor. The standard approach

is to perform a integral (over the phase-space domain) on both hand sides of the strong formulation of the problem with appropriate base functions $\{\phi_{\alpha i} = w_\alpha v_i\}$, which reads

$$-\int_{\Omega_{\mathbf{xq}}} \phi_{\alpha i} \nabla_{\mathbf{x}} \cdot (\mathcal{D}_{\mathbf{xx}} \nabla_{\mathbf{x}} u) \ . \tag{1.22}$$

The continuous Galerkin method, taken as a convenient example for discretizing a pure diffusion problem, instructs $u(\mathbf{x}, \mathbf{q}) = \sum_{\beta,j} \mathcal{U}^{\beta j} \phi_{\beta j}$, and through a integration by part we can express the above term as

$$\sum_{\beta,j} \mathcal{U}^{\beta j} \int_{\Omega} (\nabla_{\mathbf{x}} \phi_{\alpha i}) \mathcal{D}_{\mathbf{xx}} \nabla_{\mathbf{x}} \phi_{\beta j} \ , \tag{1.23}$$

where $\Omega \equiv \Omega_x \otimes \Omega_q$ represents the volume integral in two sub-domains, with the integrand explicitly reads

$$(\nabla_{\mathbf{x}} v_j)^T \cdot (\mathcal{D}(\mathbf{x}, p) \nabla_{\mathbf{x}} v_i) \cdot (w_\alpha w_\beta) \ . \tag{1.24}$$

Note that `deal.II` can handle the discrete integral with continuous or discontinuous base functions in a cell-by-cell manner (based on continuous or discontinuous Galerkin method), so that a common CR propagation problem with diffusion and advection terms can be properly defined. Logically in `BIFET` what we do is to first iterate over active cell-pairs living in the two sub-domains, and then iterates through quadrature points are conducted where the accumulations of $\mathcal{M}_{\mathbf{xq}}$ and $\mathcal{R}_{\mathbf{xq}}$ are done as discrete integrals. Notice that a integral over two sub-domains is required, so we end up with four levels of nested iterations. Although the strong formula was defined in one-dimensional domain or sub-domains, the algorithms are dimension free.

Chapter 2

BIFET

It is known that physical processes and phenomena are conventionally described in the phase-space domain built by time, space and momentum. Depending on the level of detail we focus on, the dimension in which a physical problem lives can be reduced either by integrating over less important coordinates or by assuming certain symmetries. For numerical simulations of cosmic ray (CR) propagation (here we treat CRs as continuous fluids), it is always better to pursue high-dimensional descriptions if not limited by computational methods or resources. Previously without a convenient high-dimensional partial-differential-equation (PDE) solver, we are usually limited to an isotropic CR distribution in the momentum sub-domain and either spherical or cylindrical symmetry in the spatial sub-domain. This has become less favoured as the observation precision has been improved dramatically, and thus simplified modellings are not sufficient for the frontier studies any more.

To cope with the growing requirements in precision and resolution of CR propagation simulation, it is inevitable to consider using mathematically certified libraries to help physicists build numerical simulators properly and efficiently and so to free them from the swamp of mathematics and programming. Here we propose `BIFET`, the bi-domain finite element toolkit, which is a `deal.II` based package that provides convenient functions for solving high-dimensional[1] PDEs. Driven by such motivations, `BIFET` is designed to decompose high-dimensional problems into two sub-domains, e.g., expressing a phase-space distribution with spatial and momentum coordinates separately. The triangulation[2] in each sub-domain can thus be carried out independently, and as well for other mathematical quantities like the finite-element and sparsity pattern. The back-end methods introduced here for assembling high-dimension linear algebra from two sub-domains root deeply in the `deal.II` library.

2.1 Software Design

As mentioned earlier that the main feature we implement in `BIFET` is assembling the linear algebra structure with triangulation performed in two domains independently. In

[1] By high-dimension we mean dimension higher than three.

[2] In geometry, a triangulation is a subdivision of a planar object into triangles, and by extension the subdivision of a higher-dimension geometric object into simplices.

the following we present the technical details related to building the numerical system for solving a high-dimensional PDE. An illustrative `BIFET` workflow chart is presented by Fig. 2.1, where the whole routine mainly consists of two processes, one is shown on the left side of the workflow corresponds to initializing/refining and storing the linear algebra system of the PDEs, while the right side of the workflow displays the operations related to solving the PDE system and interacting with the conditions.

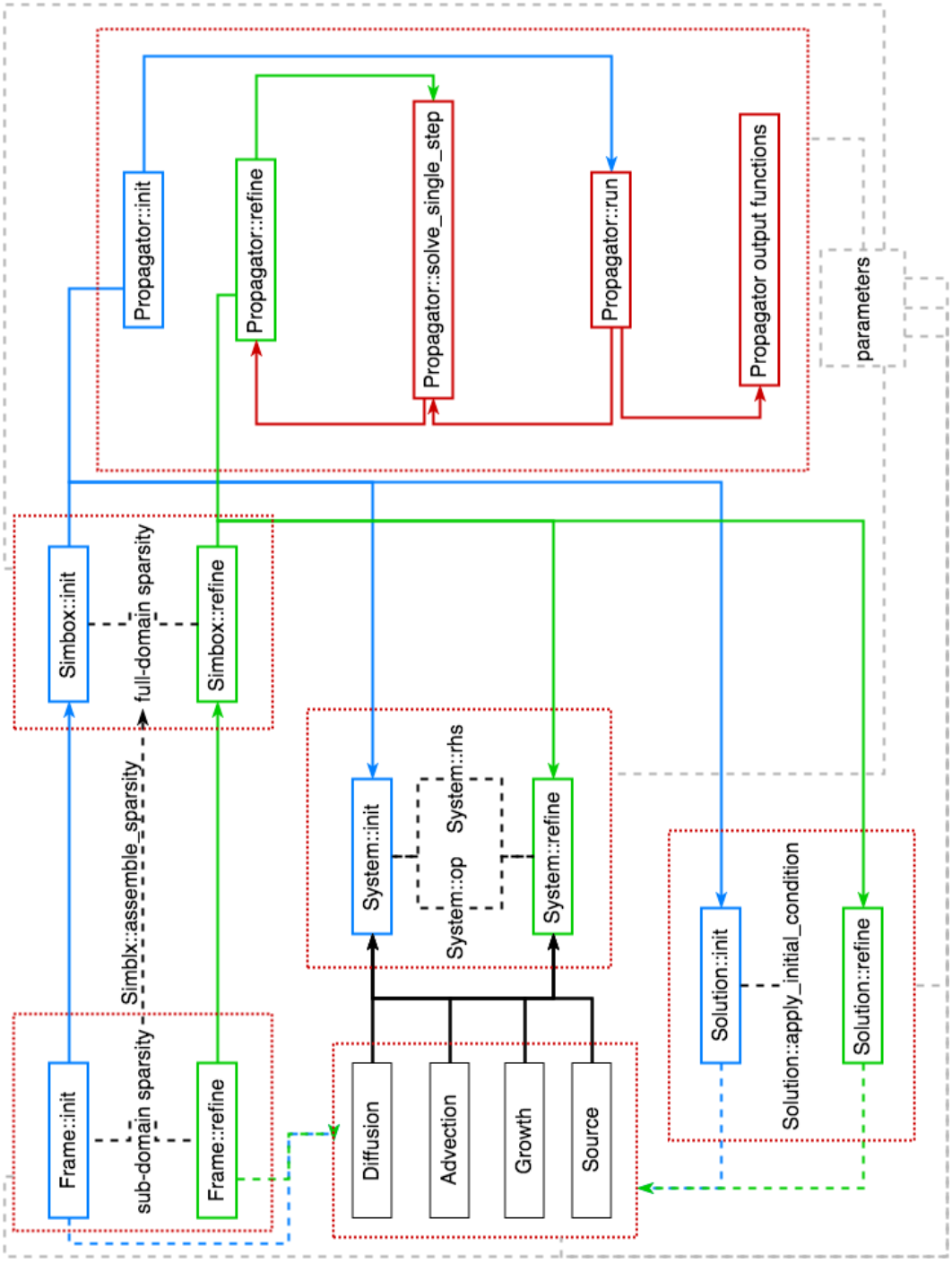

Figure 2.1: `BIFET` workflow.

2.1.1 Domains

While physically we distinguish between spatial and spectral domains, which are not different from the numerical point of view. Technically we distinguish domain discretization with continuous Galerkin method from that with the discontinuous Galerkin. In the *Frame* class we define the basic quantities for describing a domain, which include mesh/grid geometry and size, finite element degrees of freedom, dynamic sparsity pattern, strong boundary condition and hanging node constraints. The process of initializing a single domain setting starts with the given mesh/grid shape, size and discretization, from where the triangulation can be done automatically with built-in method of `deal.II` library. After which, with given finite element method, we can estimate the degrees of freedom and dynamic sparsity pattern according to the initial discretization, where the degrees of freedom represents how many independent unknown variables in the final solution, while the sparsity pattern describes the basic shape of the linear system left-hand-side matrix. Hanging node constraints are important only for continuous Galerkin method, where the solution is required to be continuous at the boundary of two neighbouring cells. Note that these constraints should not be used for the discontinuous Galerkin method. Here we present the implementation of the initializing process in *Frame* class.

```
template <int dim> void Frame<dim>::init() {
  // triangulate simulation box
  dealii::GridGenerator::subdivided_hyper_rectangle(
                                *(this->triangulation),
                                this->block_nums, this->pivot_min,
                                this->pivot_max, true);
  // if min_refine_lv is 0, no refinement operation will be taken
  this->triangulation->refine_global(this->min_refine_lv);
  // enumerate dof
  this->dof_handler->distribute_dofs(*(this->fe));
  // apply dof to constraints
  this->constraints->clear();
  dealii::DoFTools::make_hanging_node_constraints(
                                *(this->dof_handler),
                                *(this->constraints));
  // apply strong boundary
  this->bfmap_init();
  dealii::VectorTools::interpolate_boundary_values(
                                *(this->dof_handler),
                                *(this->bfmap),
                                *(this->constraints));
  this->constraints->close();
  // initialize dynamic sparsity
  this->dsp->reinit(this->dof_handler->n_dofs(),
                                this->dof_handler->n_dofs());
  dealii::DoFTools::make_sparsity_pattern(
```

```
                                    *(this->dof_handler),
                                    *(this->dsp),
                                    *(this->constraints),
                                    /*keep_constrained_dofs=*/false);
  this->sparsity->copy_from(*(this->dsp));
}
```

2.1.2 Sparsity Pattern

The sparsity pattern (as introduced above) for a single sub-domain, i.e., $\mathcal{S}_{\mathbf{x}}$ for the spatial domain and $\mathcal{S}_{\mathbf{q}}$ for the spectral domain, is built during initializing the *Frame* instance. For the system left-hand-side matrix which absorbs the system matrices from two sub-domains, the corresponding sparsity pattern is calculated as

$$\mathcal{S}_{\mathbf{xq}} = \mathcal{S}_{\mathbf{q}} \otimes \mathcal{S}_{\mathbf{x}} \ , \tag{2.1}$$

which is generic and independent of the specific expression of the system matrix itself. In the following we present the implementation of the Kronecker product described in Eq. 2.1. This function is defined in the *Simbox* class along with functions for refining sub-domains.

```
template <int spa_dim, int spe_dim>
void Simbox<spa_dim, spe_dim>::Kronecker_product() {
  // reallocate result DSP
  this->dsp->reinit(
      this->spectral_frame->dsp->n_rows() *
       ↪ this->spatial_frame->dsp->n_rows(),
      this->spectral_frame->dsp->n_cols() *
       ↪ this->spatial_frame->dsp->n_cols());
  // loop through non-zero entries in left DSP
  auto it_left = this->spectral_frame->dsp->begin();
  auto end_left = this->spectral_frame->dsp->end();
  for (; it_left != end_left; ++it_left) {
    auto alpha = it_left->row();
    auto beta = it_left->column();
    // loop through non-zero entries in right DSP
    auto it_right = this->spatial_frame->dsp->begin();
    auto end_right = this->spatial_frame->dsp->end();
    for (; it_right != end_right; ++it_right) {
      // get global indeces
      auto I = alpha * this->spatial_frame->dsp->n_rows() +
       ↪ it_right->row();
      auto J = beta * this->spatial_frame->dsp->n_cols() +
       ↪ it_right->column();
      this->dsp->add(I, J);
```

```
    }
  }
}
```

2.1.3 System Assembling

The sparsity pattern for the full domain is useful in assembling and storing the system left-hand-side matrix. The basic idea of assembling a system matrix is similar to the standard way defined in `deal.II` library, where local matrices are assembled in a cell-wise manner and then distributed into the global matrix. Since we are independently handling two sub-domains, the iteration at cell level is nested, which means a local matrix is not associated to a single cell but to a couple of cells from two sub-domains. The assembling method of local matrices in each domain is still valid, while distributing local matrices to the global matrix requires the same method in `deal.II` and the Kronecker product which merge the global sub-domain system matrices into the global full domain matrix. Note that the Kronecker product in merging two global matrices is not relevant to whether the left-hand-side operators can be decomposed into two sub-domains, since during the cell-wise assembling of the local matrices we naturally use the specific expression (with nested iterations of quadrature points in both sub-domains) of the left-hand-side operators. The snippet below presents the system initialization function for a pure spatial diffusion problem, where the diffusion tensor is defined within *System* class (which applies to the definition of advection vector and source distribution).

```
template <int spa_dim, int spe_dim>
void System_tmp<spa_dim, spe_dim>::Operator::init(
                                  System<spa_dim, spe_dim> *system,
                                  const Simbox<spa_dim, spe_dim> *simbox,
                                  const double &step_time) {
  // step 1, preparation
  // instantiate quadrature rules in two sub-domains
  auto spatial_quadrature_formula =
  std::make_unique<dealii::QGauss<spa_dim>>(
                                  simbox->spatial_frame->fe->degree + 1);
  auto spectral_quadrature_formula =
  std::make_unique<dealii::QGauss<spe_dim>>(
                                 simbox->spectral_frame->fe->degree + 1);
  // prepare finite element base function values in spatial domain
  auto spatial_fev = std::make_unique<dealii::FEValues<spa_dim>>(
                                  *(simbox->spatial_frame->fe),
                                  *spatial_quadrature_formula,
                                  dealii::update_gradients |
                                  dealii::update_quadrature_points |
                                  dealii::update_JxW_values);
  // prepare finite element base function values in spatial domain
```

```
22    auto spectral_fev = std::make_unique<dealii::FEValues<spe_dim>>(
23                                  *(simbox->spectral_frame->fe),
24                                  *spectral_quadrature_formula,
25                                  dealii::update_values |
26                                  dealii::update_quadrature_points |
27                                  dealii::update_JxW_values);
28    // degrees of freedom per cell (DPC) in two sub-domains
29    const unsigned int spatial_dpc = spatial_fev->dofs_per_cell;
30    const unsigned int spectral_dpc = spectral_fev->dofs_per_cell;
31    // number of quadrature points per cell in two sub-domains
32    const unsigned int spatial_q_points =
33    spatial_quadrature_formula->size();
34    const unsigned int spectral_q_points =
35    spectral_quadrature_formula->size();
36    // local to global matrix indices translater
37    auto spatial_l2g =
38    std::make_unique<std::vector<dealii::types::global_dof_index>>(
39                                  spatial_dpc);
40    auto spectral_l2g =
41    std::make_unique<std::vector<dealii::types::global_dof_index>>(
42                                  spectral_dpc);
43    // temporary local (per-cell) matrix caches
44    auto cell_Mx =
45    std::make_unique<dealii::FullMatrix<double>>(spatial_dpc,
46                                                 spatial_dpc);
47    auto cell_Mq =
48    std::make_unique<dealii::FullMatrix<double>>(spectral_dpc,
49                                                 spectral_dpc);
50    // system matrix allocation
51    system->Mxq->reinit(*(simbox->sparsity));
52
53    // step 2, fill system matrix
54    // apply integral with base functions over sub-domains
55    // iterate over sub-domain cells (spatial domain)
56  #ifdef _OPENMP
57    system->omp_cell_distribute(simbox);
58    for (auto spatial_cell = system->it_start;
59         spatial_cell != system->it_end;
60         ++spatial_cell)
61  #else
62      for (const auto& spatial_cell :
63      simbox->spatial_frame->dof_handler->active_cell_iterators())
64  #endif
65      {
66        // initialize finite element values at given cell
67        spatial_fev->reinit(spatial_cell);
```

```
68        // translate local indices to global indices
69        spatial_cell->get_dof_indices(*spatial_l2g);
70        // iterate over sub-domain cells (spectral domain)
71        for (const auto& spectral_cell :
72        simbox->spectral_frame->dof_handler->active_cell_iterators())
73        {
74          spectral_fev->reinit(spectral_cell);
75          // translate local indices to global indices
76          spectral_cell->get_dof_indices(*spectral_l2g);
77          // apply quadrature rule in spectral domain
78          for (unsigned int spectral_qid = 0;
79               spectral_qid < spectral_q_points;
80               ++spectral_qid) {
81            // spectral domain local full matrix
82            for (dealii::types::global_dof_index alpha = 0;
83                 alpha < spectral_dpc;
84                 ++alpha) {
85              for (dealii::types::global_dof_index beta = 0;
86                   beta < spectral_dpc;
87                   ++beta) {
88                cell_Mq->set(alpha, beta,
89                             spectral_fev->shape_value(alpha,
90                                                       spectral_qid) *
91                             spectral_fev->shape_value(beta,
92                                                       spectral_qid) *
93                             spectral_fev->JxW(spectral_qid));
94              } // beta
95            }   // alpha
96            // (clean cache)
97            system->Mq->reinit(*(simbox->spectral_frame->sparsity));
98            // (push local full matrix to global sparse matrix cache)
99            simbox->spectral_frame->constraints
100                                  ->distribute_local_to_global(
101                                          *cell_Mq,
102                                          *spectral_l2g,
103                                          *(system->Mq));
104           // apply quadrature rule in spatial domain
105           for (unsigned int spatial_qid = 0;
106                spatial_qid < spatial_q_points;
107                ++spatial_qid) {
108             // get spatial diffusion tensor at given quadrature point
109             const dealii::Tensor<2, spa_dim, double> coefficient{
110               system->diffusion->Dxx(
111                         spatial_fev->quadrature_point(spatial_qid),
112                         spectral_fev->quadrature_point(spectral_qid),
113                         step_time)};
```

```
114             // spatial domain local full matrix
115             for (dealii::types::global_dof_index i = 0;
116                  i < spatial_dpc;
117                  ++i) {
118               for (dealii::types::global_dof_index j = 0;
119                    j < spatial_dpc;
120                    ++j) {
121                 cell_Mx->set(i, j,
122                     dealii::scalar_product(
123                     spatial_fev->shape_grad(i, spatial_qid),
124                     coefficient *
125                     spatial_fev->shape_grad(j, spatial_qid)) *
126                     spatial_fev->JxW(spatial_qid));
127               } // j
128             }   // i
129             // (clean cache)
130             system->Mx->reinit(*(simbox->spatial_frame->sparsity));
131             // (push local full matrix to global sparse matrix cache)
132             simbox->spatial_frame->constraints
133                                  ->distribute_local_to_global(
134                                         *cell_Mx,
135                                         *spatial_l2g,
136                                         *(system->Mx));
137             // accumulate to global matrix cache
138             system->Operator_Kronecker_accumulate(simbox);
139           } // spatial quadrature point
140         }   // spectral quadrature point
141       }     // spectral cell
142     }       // spatial cell
143 }
```

2.1.4 multi-threading support

In the first version of BIFET we apply multi-threading parallelism mainly to the system assembling process, which has already been illustrated by the snippet above. We will see later in the profiling that by allocate the cell iterations into multiply thread is efficient until the bottleneck from memory accessing is reached. This bottleneck is purely due to the fact that we have to allocate and compute all the non-zero elements of system matrix. To over come which, it is essential in the future to implement MPI support with a matrix-free scheme in system matrix calculation, where the system matrix do not have to be pre-calculated and in turn reduces greatly the computing memory consumption and makes the process easy to be paralleled.

2.2 Precision and Performance

2.2.1 Performance

The common routines associated to building and solving PDE system in `BIFET` are data-intensive. The largest memory consumption comes from assembling PDE operator matrices. Each operator matrix size is defined together by the domain resolution (namely, the number of cells), the base function polynomial order (which determines the degrees of freedom per cell) and the problem dimension. The main idea for computational parallelism is to distribute the workload related to accessing these matrices since the operator matrices always stay in the RAM (random-access memory). At the lowest optimizing level we apply a multi-threading with `OpenMP`[3], which is easy to be implemented and nested inside other packages, i.e., the `IMAGINE` pipeline with multi-node parallelism.

A standard simulation routine of `BIFET` is mainly built by iterations with three major processes: the system initialization, system solver and (non-)adaptive refinement. Fig. 2.2 illustrates the CPU time consumption for handling simple time-independent diffusion and advection problems with $\mathbb{R}^{1+1}$ dimension setting by serial routines in `BIFET`. The CPU time cost of system initialization and refinement are roughly proportional to the square of degrees of freedom, but actually faster thanks to the sparsity in the system matrix. This is expected since the system matrices have their sizes proportional to the square of the total degrees of freedom. For the diffusion problem, we use an iterative solver so that the scaling index is close to 2.0. While for the advection problem, a direct solver is adopted and so the solving time scales almost linearly with respect to the system total degrees of freedom. We also observe that the system initialization and adaptive refinement are computationally at least one magnitude more expensive than the solver, for a problem more complicated than the pure diffusion or advection the difference is larger.

problem\process	initialization	refinement	solver
diffusion	1.59	1.67	1.88
advection	1.38	1.46	1.26

Table 2.1: The scaling index of CPU time consumption as a power-law function of the degrees of freedom in the discretized problems by FEM displayed in Fig. 2.2.

According to the serial profiling, initialization and adaptive refinement processes are the major optimization targets. With further profiling which is not presented here, we find the most time consuming part in both initialization and refinement processes is the assembling system matrix $\mathcal{M}_{\mathbf{xq}}$ with sufficiently high degrees of freedom. By using `OpenMP`, it is possible to fork the *System* objects among the available CPU working threads, where each thread assembles a certain fraction of $\mathcal{M}_{\mathbf{xq}}$ and $\mathcal{R}_{\mathbf{xq}}$. In addition to distributing *System* access among the threads, the refinement process defined within the *Solution* class is optimized by following the very same idea. By increasing the number of threads, the non-optimized and memory-access-related operations gradually

[3] https://www.openmp.org/

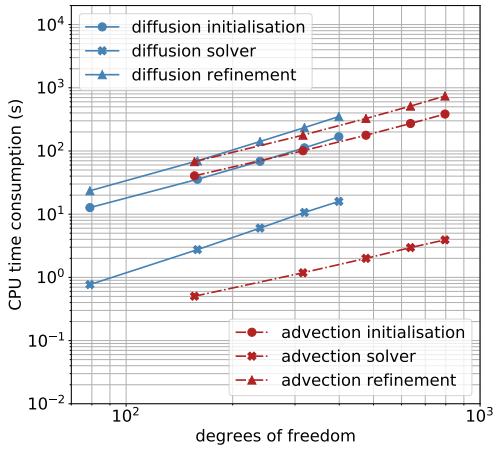

Figure 2.2: CPU time consumption of typical standard `BIFET`routines in serial mode. Iterative solver in diffusion problem results in quadratic scaling, while direct solver in advection problem gives linear scaling. The scaling indices are presented in Tab. 2.1

dominate over the paralleled part in the CPU time consumption when the workload for a problem (e.g., calculating the diffusion or advection coefficient at each supporting point) is not heavy enough, in which cases the strong scaling speedups hit the rooftops as illustrated in Fig. 2.3.

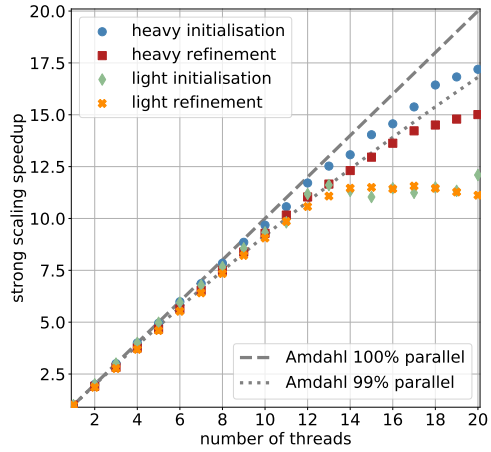

Figure 2.3: Strong scaling speedups of initialization and refinement processes in computationally light and heavy problems. As a benchmark we present Amdahl's law of fully paralleled or 99% paralleled. Rooftops occur in computationally light cases before exploiting the available threads.

2.2.2 Precision

The capability and precision of `BIFET` pipelines in confronting common physical scenarios in CR propagation are illustrated by a series of integrated tests in the following. With an analytically solvable problem, we can compare the numerical solution u_h to the corresponding analytic one u by estimating the L^2 errors at any given testing position $(\mathbf{x}, \mathbf{q})$ in the simulation domain

$$\epsilon_{\mathrm{L}^2,\mathbf{q}} = \sqrt{\int_{\Omega_\mathbf{x}} \left[u(\mathbf{q}) - u_h(\mathbf{q})\right]^2} , \tag{2.2}$$

$$\epsilon_{\mathrm{L}^2,\mathbf{x}} = \sqrt{\int_{\Omega_\mathbf{q}} \left[u(\mathbf{x}) - u_h(\mathbf{x})\right]^2} , \tag{2.3}$$

where for simplicity with built-in library functions provided by `deal.II`, error estimations are calculated in a single sub-domain, e.g., $\epsilon_{L^2,\mathbf{q}}$ is defined as the spatial sub-domain error by interpolating the solution u_h at the given spectral position $\mathbf{q}$.

A pure diffusion or mathematically speaking a parabolic problem, is the simplest testing case we can start with. We prepare a typical strong formulation for the diffusion problem as

$$\partial_t u(\mathbf{x}, \mathbf{q}, t) - \nabla_\mathbf{x} \cdot (\mathcal{D}_\mathbf{xx} \nabla_\mathbf{x} u(\mathbf{x}, \mathbf{q}, t)) = f(\mathbf{x}, \mathbf{q}, t) , \tag{2.4}$$

$$u(\mathbf{x}, \mathbf{q}, t) = 0 , \quad \mathbf{x} \in \partial\Omega_x , \tag{2.5}$$

where a homogeneous strong condition is defined on all boundaries. For the testing purpose, a simple steady-state solution $u(\mathbf{x}, \mathbf{q})$ which satisfies the Dirichlet boundary condition can be pre-defined as

$$u(\mathbf{x}, \mathbf{q}) = S(z)S(x)S(y) , \tag{2.6}$$

where for abbreviation $C(i)$ represents $\cos(\frac{(i-i_{min})\pi}{L_i})$ and $S(i)$ for $\sin(\frac{(i-i_{min})\pi}{L_i})$ with $L_i = i_{max} - i_{min}$ defined as the simulation box length in the spatial coordinate $i \in \{x, y, z\}$. Inspired by the testing cases designed by Kissmann [8], we set a similar anisotropic spatial diffusion tensor

$$\mathcal{D}_\mathbf{xx} = \begin{pmatrix} \alpha z^2 & 0 & 0 \\ 0 & \beta x^2 & \beta xy \\ 0 & \beta xy & \beta y^2 \end{pmatrix} , \tag{2.7}$$

where we set $\alpha \neq \beta$ for anisotropy. The weak formulation of this problem has been presented as an example of domain separation earlier. In the $\mathbb{R}^{1+m}$ setting, the right-hand-side source term which can provide uniquely the pre-defined solution reads

$$f(z) = \frac{\pi^2 \alpha z^2}{L_z^2} S(z) - \frac{2\pi\alpha z}{L_z} C(z) , \tag{2.8}$$

while in the $\mathbb{R}^{2+m}$ dimension setting, its expression should be

$$\begin{aligned} f(z,x) &= f(z)S(x) \\ &\quad + \frac{\pi^2\beta x^2}{L_x^2}S(z)S(x) - \frac{2\pi\beta x}{L_x}S(z)C(x) \ , \end{aligned} \tag{2.9}$$

and finally in the $\mathbb{R}^{3+m}$ setting, the source term is

$$\begin{aligned} f(z,x,y) &= f(z,x)S(y) \\ &\quad - \frac{2\pi^2\beta xy}{L_xL_y}S(z)C(x)C(y) \\ &\quad - \frac{\pi\beta x}{L_x}S(z)C(x)S(y) - \frac{\pi\beta y}{L_y}S(z)S(x)C(y) \\ &\quad + \frac{\pi^2\beta y^2}{L_y^2}S(z)S(x)S(y) - \frac{2\pi\beta y}{L_y}S(z)S(x)C(y) \ . \end{aligned} \tag{2.10}$$

A direct and efficient approach to this problem is to use a time-independent solver with continuous Galerkin method in `BIFET`. Fig. 2.4 displays the spatial sub-domain L^2 errors estimated with different dimension and refinement settings, where in practice the total volume of the spatial sub-domain is fixed by setting $L_x = L_y = L_z = L$. For a numerical solution u_h found with (dis)continuous Galerkin base functions up to polynomial order p, the corresponding L^2 errors should follow h^{p+1} scaling where h represents the homogeneous numerical cell length in each spatial direction. This means at each global refinement level, the total number of elemental cells is L/h in each spatial direction. On the other hand, solutions found with adaptive refinement scheme do not respect the h^{p+1} scaling law since the elemental cells are refined inhomogeneously. Nevertheless, we still managed to find a roughly linear (but slightly steeper) scaling of L^2 errors in the adaptively refined cases with respect to the minimal (but not all) cell length h.

For testing the time-dependent solving routines, we intend to recover the steady-state solutions by a time-dependent solver with the Crank-Nicolson method. The left panel in Fig. 2.5 illustrates the evolving property of the time-dependent solver with fixed time-step difference d while increasing the total evolving step T/d until the minimal error found by the time-dependent solver is reached asymptotically. Note that the minimal evolving steps required for reaching the steady-state solution depends on the specific dimension and resolution settings of a problem. The convergence property of the time-dependent solver is presented by the right panel, where the total evolving time T is fixed. With different spatial resolution defined by L/h, we marked the saturation point beyond which further time discretization becomes redundant.

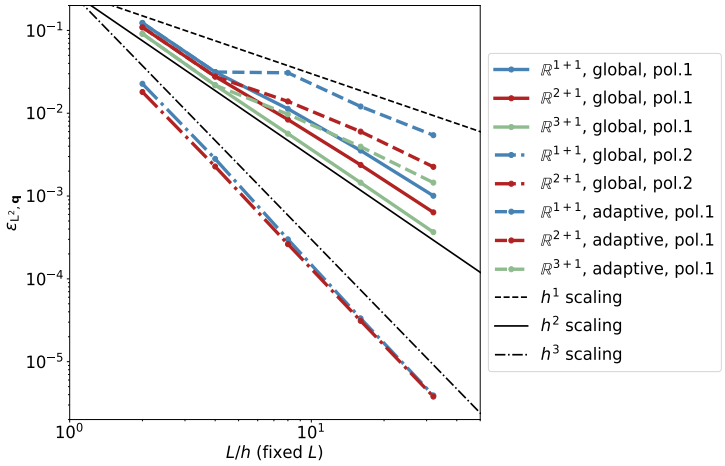

Figure 2.4: Spatial sub-domain L^2 errors measured in solving the spatial diffusion problem with a time-independent solver. "adaptive" indicates the adaptive refinement scheme while "global" indicates the homogeneous global refinement scheme. "pol.ξ" indicates up to the ξ-th order of polynomials are adopted as finite element base functions. Errors estimated with adaptive refinement (adaptive refinement ratio is set as 50%) are plotted according to the same refinement level compared to the globally refined counterparts.

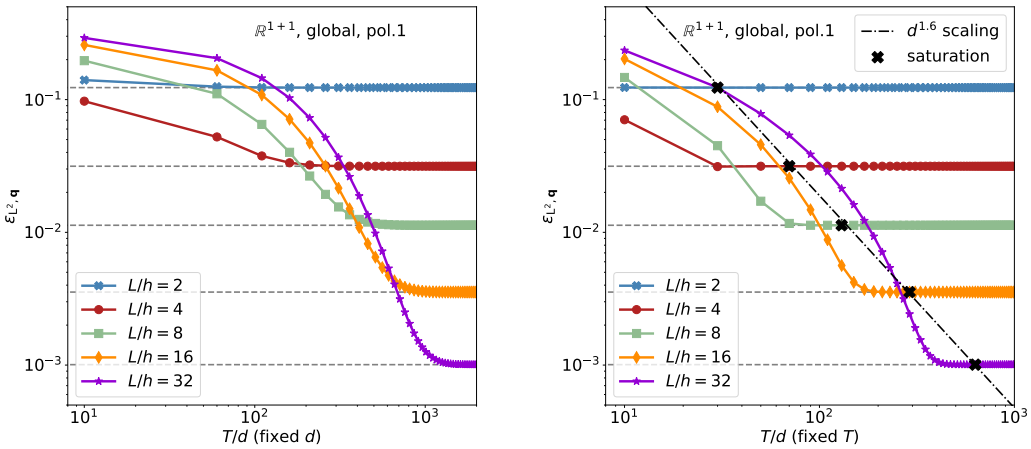

Figure 2.5: Spatial sub-domain L^2 errors measured in solving the spatial diffusion problem with a time-dependent solver. The problem is defined in $\mathbb{R}^{1+1}$ with homogeneously refined mesh and base functions at polynomial order 1. The minimum L^2 errors corresponds to the steady-state solutions are displayed in dashed lines. The saturation positions in the right panel are chosen at where the relative difference between time-dependent and time-independent solutions is below 10^{-6}.

The continuous energy loss (spectral advection) is a typical and important scenario in CR propagation where we can experiment the discontinuous Galerkin method in the spectral sub-domain while keeping the spatial sub-domain safely discretized by the continuous Galerkin method if no spatial advection phenomena shows up. The strong formulation of a simple spectral advection problem is defined as

$$\partial_t u(\mathbf{x},\mathbf{q},t) + \nabla_{\mathbf{E}} \cdot (\mathcal{A}_{\mathbf{qq}} u(\mathbf{x},\mathbf{q},t)) = f(\mathbf{x},\mathbf{q},t) \ , \tag{2.11}$$

$$u(\mathbf{x},\mathbf{q},t) = 0 \ , \quad \mathbf{q} \in \partial\Omega^{+}_{\mathbf{q}} \ , \tag{2.12}$$

where $\partial_{E_i} = \exp(-q_i)\partial_{q_i}$ since the spectral sub-domain can be built in logarithmic scale. Similar to the previous diffusion problem, the spectral sub-domain coordinates are represented by $\{q_x, q_y, q_z\}$. An anisotropic spectral advection vector $\mathcal{A}$ is assumed to be

$$\mathcal{A}_{\mathbf{qq}} = \begin{pmatrix} \eta_z \exp\left(n_z(q_z - q_{z,min})\right) \\ \eta_x \exp\left(n_x(q_x - q_{x,min})\right) \\ \eta_y \exp\left(n_y(q_y - q_{y,min})\right) \end{pmatrix} \ . \tag{2.13}$$

In the $\mathbb{R}^{m+1}$ setting, with a simple right-hand-side source term $f(q_z) = \exp\left(s_z(q_z - q_{z,min})\right)$, the analytic solution which satisfies the homogeneous strong boundary condition reads

$$\begin{aligned} u(z) = & \frac{\exp\left((n_z - s_z)q_{z,min}\right)}{(1+s_z)\eta_z} \exp\left(q_z(1+s_z-n_z)\right) \\ & - \frac{\exp\left((n_z+1)q_{z,min} + (1+s_z)L_{q_z}\right)}{(1+s_z)\eta_z} \exp\left(-n_z q_z\right) \ . \end{aligned} \tag{2.14}$$

For the testing purpose we require $u(\mathbf{x},\mathbf{q},t) = u(q_z)u(q_x)u(q_y)$, then in analogy to the $\mathbb{R}^{m+1}$ case the source term for the $\mathbb{R}^{m+3}$ setting reads

$$\begin{aligned} f(q_z,q_x,q_y) = & \ f(q_z)u(q_x)u(q_y) \\ & + u(q_z)f(q_x)u(q_y) + u(q_z)u(q_x)f(q_y) \ . \end{aligned} \tag{2.15}$$

Note that shifting from the energy coordinate $\mathbf{E}$ derivation to its corresponding logarithmic coordinate $\mathbf{q} = \log(\mathbf{E})$ derivation introduces a diagonal tensor

$$\mathcal{T}_{\mathbf{q}} = \begin{pmatrix} \exp(-q_z) & 0 & 0 \\ 0 & \exp(-q_x) & 0 \\ 0 & 0 & \exp(-q_y) \end{pmatrix} \ , \tag{2.16}$$

which consequently brings itself and $\nabla_{\mathbf{q}}\mathcal{T}_{\mathbf{q}}$ into the weak formulation. Before applying the upwind method and boundary condition, the weak formulation for the advection

term reads

$$\begin{aligned}\nabla_{\mathbf{E}}\cdot(\mathcal{A}_{\mathbf{qq}}u) \;\rightarrow\; & \sum_k\sum_{\beta,j}\mathcal{U}_k^{\beta j}\int_{\Omega_{\mathbf{x}}}\int_{\Omega_{\mathbf{q}}^k}\phi_{\alpha i}^k\mathcal{T}_{\mathbf{q}}\nabla_{\mathbf{q}}\cdot(\mathcal{A}_{\mathbf{qq}}\phi_{\beta j}^k)\\ = & \sum_k\sum_{\beta,j}\mathcal{U}_k^{\beta j}\Bigg[\int_{\Omega_{\mathbf{x}}}\oint_{\partial\Omega_{\mathbf{q}}^k}\phi_{\alpha i}^k\mathcal{T}_{\mathbf{q}}\mathcal{A}_{\mathbf{qq}}\hat{n}_{\mathbf{q}}^k\phi_{\beta j}^k\\ & -\int_{\Omega_{\mathbf{x}}}\int_{\Omega_{\mathbf{q}}^k}(\nabla_{\mathbf{q}}\phi_{\alpha i}^k)\mathcal{T}_{\mathbf{q}}\mathcal{A}_{\mathbf{qq}}\phi_{\beta j}^k\\ & -\int_{\Omega_{\mathbf{x}}}\int_{\Omega_{\mathbf{q}}^k}\phi_{\alpha i}^k(\nabla_{\mathbf{q}}\mathcal{T}_{\mathbf{q}})\mathcal{A}_{\mathbf{qq}}\phi_{\beta j}^k\Bigg]\,,\end{aligned}\tag{2.17}$$

where base functions are independently defined in each spectral cell $\Omega_{\mathbf{q}}^k$. With the upwind method applied in order to ease the oscillation in the solution of an advection problem, each spectral internal surface integral reads

$$\sum_{\beta,j}\int_{\Omega_{\mathbf{x}}}\oint_{\partial\Omega_{\mathbf{q}}^k}\phi_{\beta j}^-\mathcal{T}_{\mathbf{q}}\mathcal{A}_{\mathbf{qq}}(\phi_{\alpha i}^+\hat{n}_{\mathbf{q}}^+ + \phi_{\alpha i}^-\hat{n}_{\mathbf{q}}^-)\,,\tag{2.18}$$

with the wind direction (pointing from downwind cell marked by $-$ to upwind cell marked by $+$) defined by $\mathcal{T}_{\mathbf{q}}\mathcal{A}_{\mathbf{qq}}$.

Spectral L^2 error scaling properties of time-independent solver with various dimension and refinement settings are illustrated by Fig. 2.6. The performance of applying a time-dependent solver to the same problem is illustrated in Fig. 2.7.

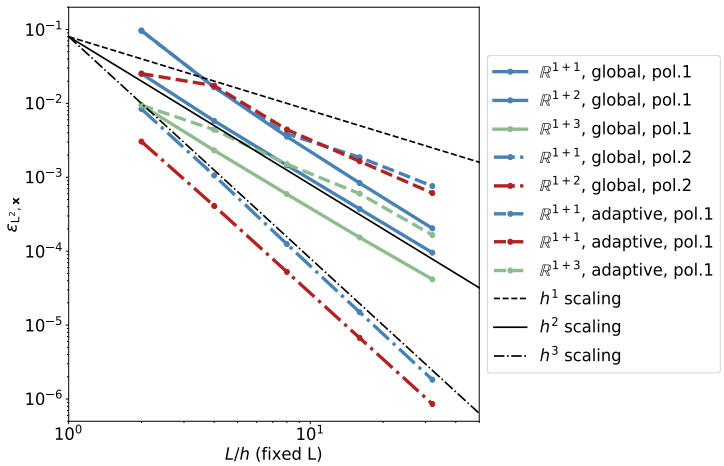

Figure 2.6: Spectral sub-domain L^2 errors measured in solving the spectral advection problem. "adaptive" indicates adaptive refinement scheme while "global" indicates global refinement scheme. "pol.ξ" indicates up to ξ-th order of polynomials are adopted as finite element base functions. Solutions found with adaptive refinement (adaptive refinement ratio is set as 50%) are plotted according to refinement level compared to globally refined counterparts.

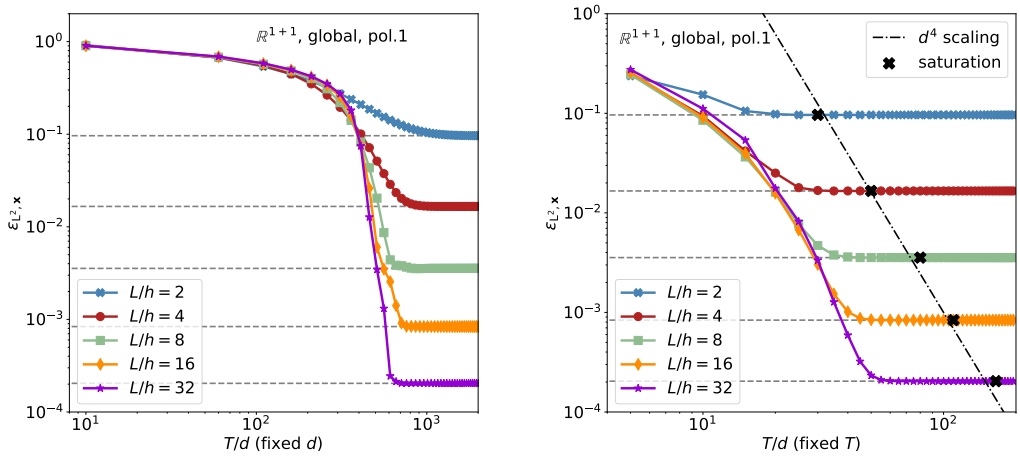

Figure 2.7: Spectral sub-domain L^2 errors measured in solving the spectral advection problem with a time-dependent solver. The problem is defined in $\mathbb{R}^{1+1}$ with homogeneously refined mesh and base functions at polynomial order 1. The minimum L^2 errors corresponds to the steady-state solutions are displayed in dashed lines. The saturation positions in the right panel are chosen at where the relative difference between time-dependent and time-independent solutions is below 10^{-6}.

In the above two testing cases, we have seen the diffusion and advection problems separately. A more realistic problem usually involves both diffusion and advection which need to be solved simultaneously either defined in the same sub-domain or in two sub-domains separately. Here we set up an advection-diffusion problem (with diffusion and advection in the same sub-domain) and approach unconventionally with the continuous Galerkin method as in the pure diffusion problem case and then observe the performance. Despite the fact that discontinuous Galerkin is the standard method for solving an advection-diffusion problem, a continuous Galerkin method however is computationally lighter and easier to be implemented and also interesting to be tested as an alternative approach. The strong formulation of a simple advection-diffusion problem is defined as

$$\partial_t u(\mathbf{x},\mathbf{q},t) + \nabla_{\mathbf{x}} \cdot (\mathcal{A}_{\mathbf{xx}} u(\mathbf{x},\mathbf{q},t)) - \nabla_{\mathbf{x}} \cdot (\mathcal{D}_{\mathbf{xx}} \nabla_{\mathbf{x}} u(\mathbf{x},\mathbf{q},t)) = f(\mathbf{x},\mathbf{q},t) \ , \tag{2.19}$$

$$u(\mathbf{x},\mathbf{q},t) = 0 \ , \quad \mathbf{x} \in \partial\Omega^{+}_{\mathbf{x}} \ , \tag{2.20}$$

$$\hat{\mathbf{n}}_{\mathbf{x}} \cdot (\mathcal{A}_{\mathbf{xx}} - \mathcal{D}_{\mathbf{xx}} \nabla_{\mathbf{x}}) u(\mathbf{x},\mathbf{q},t) = 0 \ , \quad \mathbf{x} \in \partial\Omega^{-}_{\mathbf{x}} \ , \tag{2.21}$$

where $\partial\Omega^{+}$ and $\partial\Omega^{-}$ represent the upper and lower surface bounds respectively. We do not intend to complicate advection or diffusion tensors, and so they are set with constant values

$$\mathcal{A}_{\mathbf{xx}} = \begin{pmatrix} \eta_z \\ \eta_x \\ \eta_y \end{pmatrix} \ , \quad \mathcal{D}_{\mathbf{xx}} = \begin{pmatrix} \alpha & 0 & 0 \\ 0 & \beta & 0 \\ 0 & 0 & \beta \end{pmatrix} \ . \tag{2.22}$$

In the $\mathbb{R}^{1+m}$ setting, with simple time-independent source $f(\mathbf{x},\mathbf{q},t) = \exp(z_{min} - z)$ an analytic solution that satisfies our boundary conditions can be derived as

$$u(z) = (\frac{\exp(z_{min}-z)}{(\alpha+\eta_z)} - \frac{1}{\eta_z}) \exp\left(-\frac{\eta_z(z-z_{max})}{\alpha}\right) - \frac{\exp(z_{min}-z)}{(\alpha+\eta_z)} + \frac{1}{\eta_z} \ . \tag{2.23}$$

In the $\mathbb{R}^{2+m}$ and $\mathbb{R}^{3+m}$ settings we require $u(\mathbf{x},\mathbf{q}) = u(z)u(x)u(y)$ so that the corresponding source functions reads

$$f(z,x) = f(z)u(x) + u(z)f(x) \ , \tag{2.24}$$

$$f(z,x,y) = f(z,x)u(y) + u(z)u(x)f(y) \ . \tag{2.25}$$

The left-hand-side of the time-independent weak formulation reads

$$\sum_{\beta,j} \mathcal{U}^{\beta j} \int_{\Omega} (\mathcal{D}_{\mathbf{xx}} \nabla_{\mathbf{x}} \phi_{\beta j} - \mathcal{A}_{\mathbf{xx}} \phi_{\beta j}) \cdot \nabla_{\mathbf{x}} \phi_{\alpha i} \ , \tag{2.26}$$

in which the surface integral terms vanish due to the boundary conditions.

Fig. 2.8 illustrates the precision of the time-independent solver in two different cases. In the first case we set diffusion coefficient as the same magnitude as the advection coefficient, while in the second case the diffusion term is significantly weaker than the advection. It is known that continuous Galerkin method is not appropriate for solving a pure advection problem, and so by mixing a diffusion term into the advection problem

to suppress artificial oscillation in the solution the continuous Galerkin may become feasible. We should expect that smaller diffusive partition in the advection-diffusion problem requires higher mesh refinement to reach the ideal error scaling law. This is observed in the upper panel of Fig. 2.8 where the ideal error scaling is only achieved with highly refined grid, while in the lower panel of Fig. 2.8 the ideal error scaling is well followed sicne the diffusion term is significant enough at the given mesh resolution. The performance of applying a time-dependent solver to the advection-diffusion problem is illustrated in Fig. 2.9.

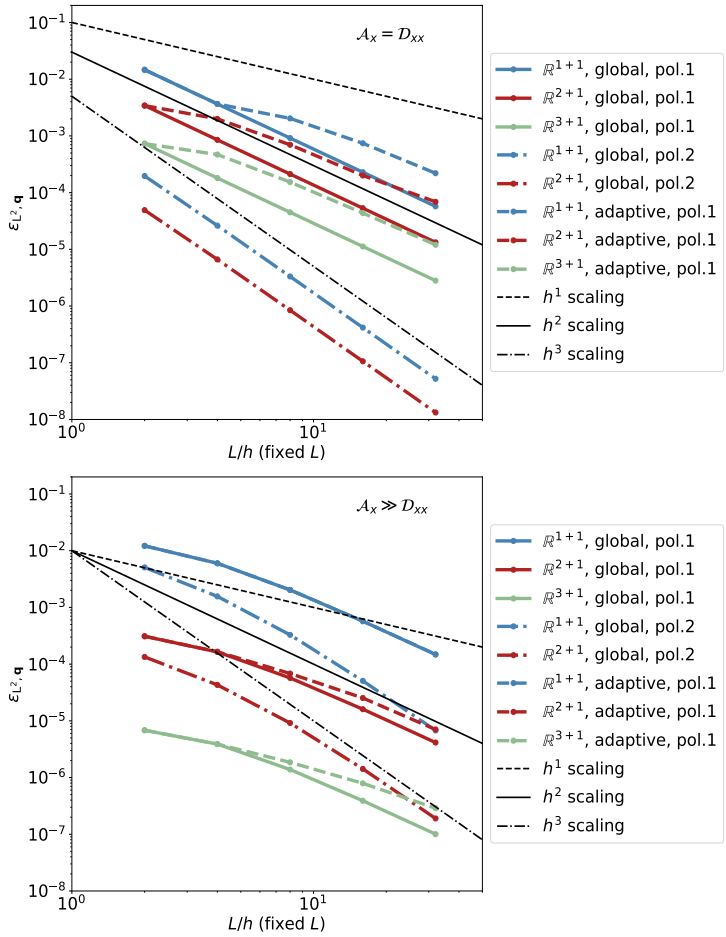

Figure 2.8: Spatial sub-domain L^2 errors measured in solving the advection-diffusion problem. "adaptive" indicates adaptive refinement scheme while "global" indicates global refinement scheme. "pol.ξ" indicates up to ξ-th order of polynomials are adopted as finite element base functions. Solutions found with adaptive refinement (adaptive refinement ratio is set as 50%) are plotted according to refinement level compared to globally refined counterparts.

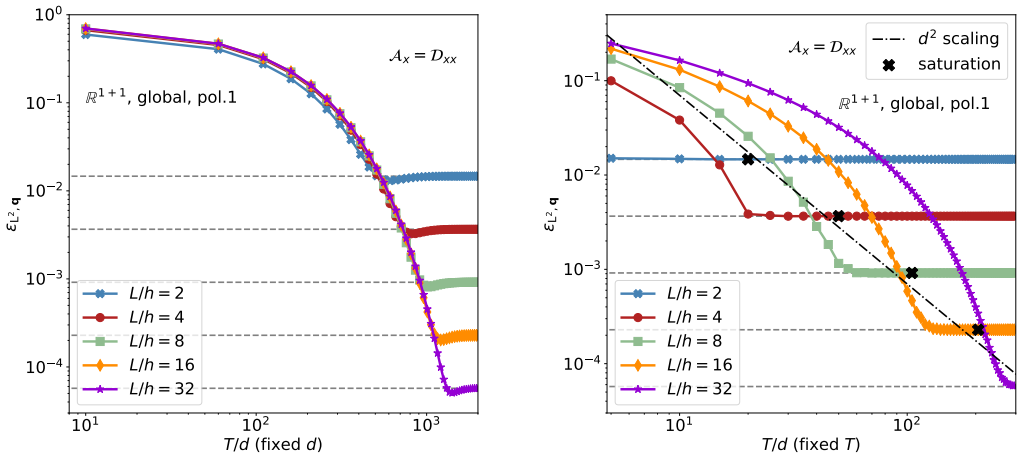

Figure 2.9: Spectral sub-domain L^2 errors measured in solving the advection-diffusion problem with a time-dependent solver. The problem is defined in $\mathbb{R}^{1+1}$ with homogeneously refined mesh and base functions at polynomial order 1. The minimum L^2 errors corresponds to the steady-state solutions are displayed in dashed lines. The saturation positions in the right panel are chosen at where the relative difference between time-dependent and time-independent solutions is below 10^{-6}.

The above tests all focus on problems non-trivially defined in a single sub-domain, from which we have collected some practical experience for more realistic problems which span across the whole domain[4]. Here we define a simple problem with constant and isotropic spatial diffusion and spectral advection as

$$\partial_t u - \nabla_{\mathbf{x}} \cdot (\mathcal{D}_{\mathbf{xx}} \nabla_{\mathbf{x}} u) + \nabla_{\mathbf{E}} \cdot (\mathcal{A}_{\mathbf{qq}} u) = f \ , \tag{2.27}$$

$$u(\mathbf{x}, \mathbf{q}, t) = 0 \ , \quad \mathbf{q} \in \partial\Omega_{\mathbf{q}}^{+} \ , \tag{2.28}$$

$$u(\mathbf{x}, \mathbf{q}, t) = 0 \ , \quad \mathbf{x} \in \partial\Omega_{\mathbf{x}} \ , \tag{2.29}$$

$$\mathcal{A}_{\mathbf{qq}} = \begin{pmatrix} \eta \\ \eta \\ \eta \end{pmatrix} \ , \quad \mathcal{D}_{\mathbf{xx}} = \begin{pmatrix} \alpha & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & \alpha \end{pmatrix} \ . \tag{2.30}$$

Since the operators (diffusion and advection) are independent, we are able to formulate the solution as $u(\mathbf{x}, \mathbf{q}, t) = u(\mathbf{x}, t)u(\mathbf{q}, t)$ and consequently the right-hand-side source as $f(\mathbf{x}, \mathbf{q}, t) = f_x(\mathbf{x}, t)u_q(\mathbf{q}, t) + f_q(\mathbf{q}, t)u_x(\mathbf{x}, t)$. By learning from the simple forms of solutions in previous tests we fill the system with

$$u_x(\xi) = \sin\left(\frac{(\xi - \xi_{min})\pi}{L_\xi}\right) \ , \tag{2.31}$$

$$u_x(\mathbf{x}) = \prod_{\xi} u(\xi) \ , \tag{2.32}$$

$$u_q(q_\xi) = \frac{\exp(q_{\xi,\min})}{\eta(1+s)}\left[\exp\left((1+s)(q_\xi - q_{\xi,\min})\right) - \exp\left((1+s)L_{q_\xi}\right)\right] \ , \tag{2.33}$$

$$u_q(\mathbf{q}) = \prod_{q_\xi} u(q_\xi) \ . \tag{2.34}$$

The weak formulation consists of the spatial component from the weak formulation of the spatial diffusion problem and the spectral component from the weak formulation of the spectral advection problem, and so the discontinuous Galerkin method is used only in the spectral domain where the advection is defined. Before applying the upwind method, the time-independent left-hand-side is represented by

$$\sum_k \sum_{\beta,j} \mathcal{U}_k^{\beta j} \int_{\Omega_{\mathbf{x}}} (\nabla_{\mathbf{x}} v_i \mathcal{D}_{\mathbf{xx}} \nabla_{\mathbf{x}} v_j)$$
$$\times \left[\oint_{\partial\Omega_{\mathbf{q}}^k} w_\alpha^k \mathcal{T}_{\mathbf{q}} \mathcal{A}_{\mathbf{qq}} \hat{n}_{\mathbf{q}}^k w_\beta^k - \int_{\Omega_{\mathbf{q}}^k} (\nabla_{\mathbf{q}} w_\alpha^k) \mathcal{T}_{\mathbf{q}} \mathcal{A}_{\mathbf{qq}} w_\beta^k - \int_{\Omega_{\mathbf{q}}^k} w_\alpha^k (\nabla_{\mathbf{q}} \mathcal{T}_{\mathbf{q}}) \mathcal{A}_{\mathbf{qq}} w_\beta^k \right] . \tag{2.35}$$

Fig. 2.10 displays the measured spatial and spectral L^2 errors with respect to the simulation mesh resolution. By applying a time-dependent solver, the asymptotic error convergence with fixed time-difference and fixed total evolving time are displayed respectively in Fig. 2.11 (for spatial L^2 errors) and Fig. 2.12 (for spectral L^2 errors).

[4] We emphasize that all problems are defined on the full domain, but when no operation is defined in a sub-domain the corresponding weak formulation is usually a trivial mass matrix.

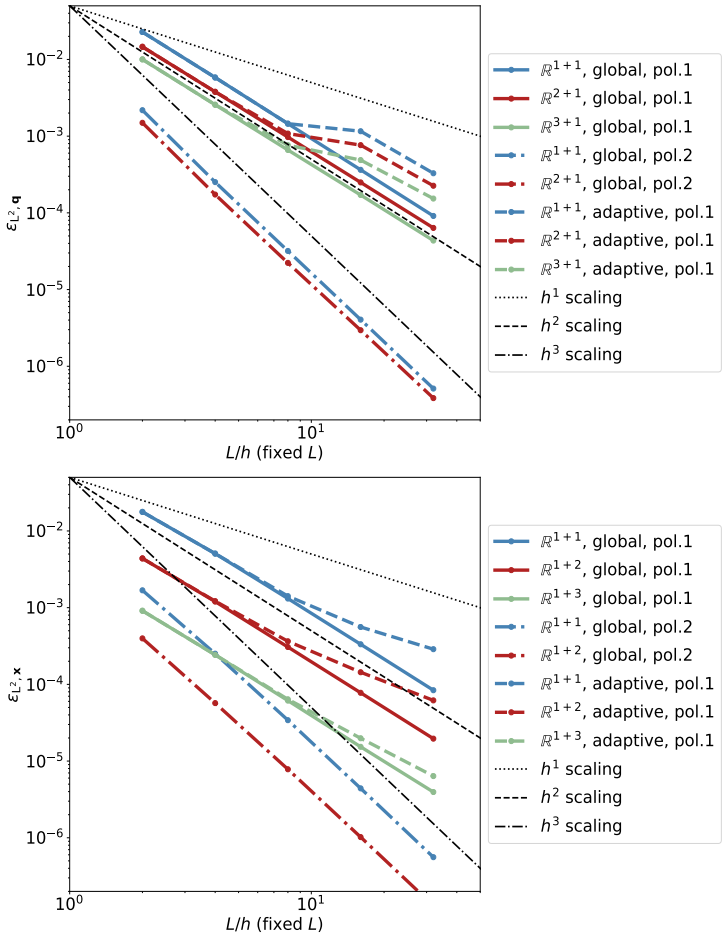

Figure 2.10: Spatial and spectral domain L^2 errors measured in the spatial diffusion with spectral advection problem. "adaptive" indicates adaptive refinement scheme while "global" indicates global refinement scheme. "pol.x" means up to x-th order of polynomials are adopted as finite element base functions. L/h means the number of cells along each spatial dimension. Solutions found with adaptive refinement are placed according to refinement level in comparison with globally refined counterparts. In this illustration we set adaptive refinement ratio as 50%.

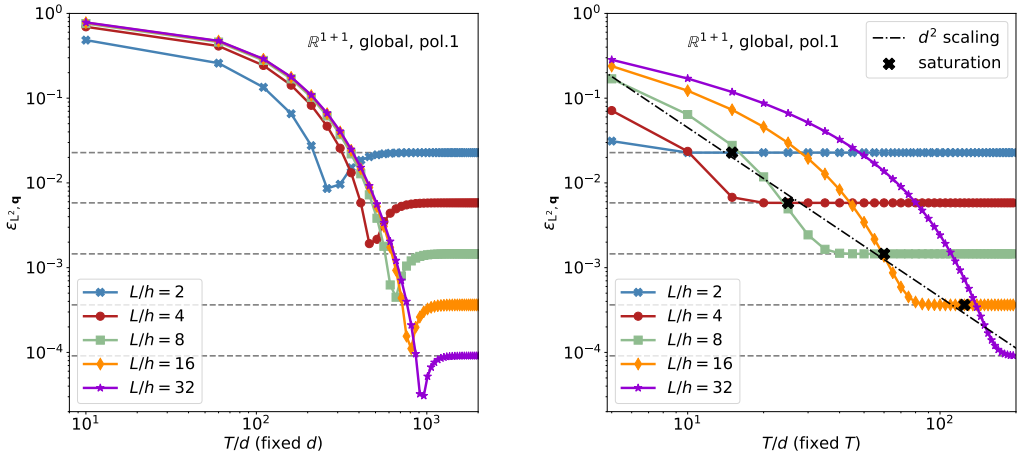

Figure 2.11: Spatial L^2 errors measured in the spatial-diffusion with spectral-advection problem with a time-dependent solver. The testing spatial diffusion problem is defined in $\mathbb{R}^{1+1}$ with homogeneously refined mesh and finite element base functions at polynomial order 1. The minimum L^2 errors corresponds to the steady-state solutions are displayed in dashed lines. The saturation positions in the right panel are chosen at where the relative difference between time-dependent and time-independent solutions is below 10^{-6}.

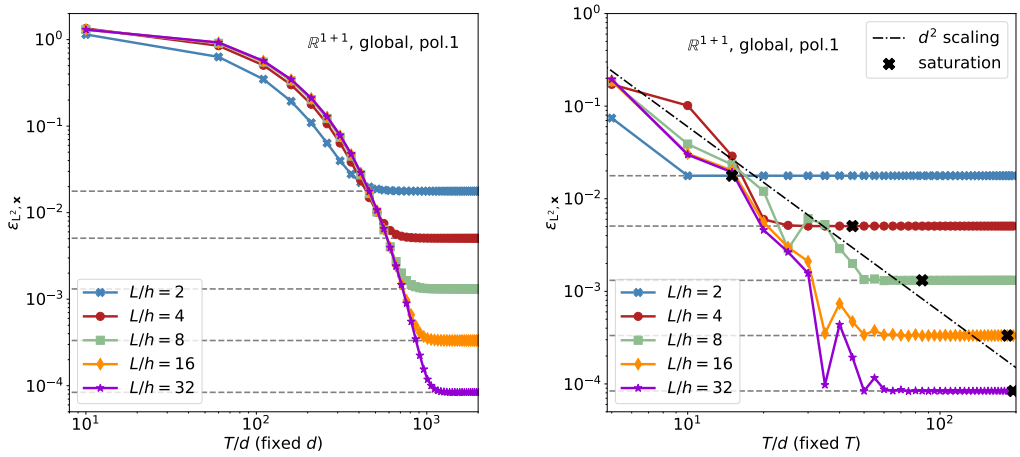

Figure 2.12: Spectral L^2 errors measured in the spatial-diffusion with spectral-advection problem with a time-dependent solver. The testing spatial diffusion problem is defined in $\mathbb{R}^{1+1}$ with homogeneously refined mesh and finite element base functions at polynomial order 1. The minimum L^2 errors corresponds to the steady-state solutions are displayed in dashed lines. The saturation positions in the right panel are chosen at where the relative difference between time-dependent and time-independent solutions is below 10^{-6}.

Chapter 3

Application Examples

Convinced by integrated tests of various typical problems, we move on to illustrate the capacity of `BIFET` in realistic simulations. The examples are designed as one of the commonly adopted simulation settings in previous studies carried out with other simulators like `Galprop` [16] and `DRAGON` [3] where GMF is pre-defined and fixed. We consider a CRE propagation problem with time-independent spatial diffusion plus spectral advection in the $\mathbb{R}^{1+1}$ dimension setting. Homogeneously distributed Galactic magnetic field is assumed without requiring CR feedback, which means no CR streaming instability in the magnetic turbulence. In the $\mathbb{R}^{3+3}$ dimension setting, the simplified CRE propagation is defined as

$$\partial_t \tilde{N} - \nabla_{\mathbf{x}} \cdot (\mathcal{D} \nabla_{\mathbf{x}} \tilde{N}) + \nabla_{\mathbf{E}} \cdot (\mathbf{b} \tilde{N}) = Q \ , \tag{3.1}$$

$$\tilde{N}(\mathbf{x}, \mathbf{q}, t) = 0 \ , \quad \mathbf{q} \in \partial\Omega^{+}_{\mathbf{q}} \ , \tag{3.2}$$

$$\tilde{N}(\mathbf{x}, \mathbf{q}, t) = 0 \ , \quad \mathbf{x} \in \partial\Omega_{\mathbf{x}} \ , \tag{3.3}$$

where $\tilde{N}(E, r)$ represents spherical symmetric CRE differential density. This toy modelling of CRE propagation can be applied to either point source modelled as some exponential profile in a homogeneous diffusive background [**?**], or extended sources in galaxy clusters [**?**] and dwarf galaxies [**?**]. Reducing to the $\mathbb{R}^{1+1}$ dimension with spherical symmetries, the time-independent propagation equation $\partial_t \tilde{N} = 0$ is reformulated as

$$-\frac{1}{r^2}\partial_r(r^2 \mathcal{D} \partial_r \tilde{N}) + \frac{1}{E^2}\partial_E(E^2 \mathbf{b} \tilde{N}) = Q \ , \tag{3.4}$$

$$\partial_r \tilde{N}(r = 0) = 0 \ , \tag{3.5}$$

$$\tilde{N}(r = r_{max}) = 0 \ , \tag{3.6}$$

$$\tilde{N}(E = E_{max}) = 0 \ . \tag{3.7}$$

We are interested in CREs reside within the energy range $E \in [10^{-2}, 10^{3}]$ GeV, where the dominant continuous energy loss mechanisms are Coulomb interactions (neglecting the degree of ionization), non-thermal bremsstrahlung (in strong-shielding limit), inverse Compton scattering and synchrotron emission, which can be approximated mono-chromatically (which means the energy loss rate is approximated as a

function of CR energy alone) as

$$
\begin{aligned}
-\mathbf{b}(E) &= \mathbf{b}_{\rm ic} + \mathbf{b}_{\rm sync} + \mathbf{b}_{\rm coul} + \mathbf{b}_{\rm brem} \ , && (3.8)\\
\mathbf{b}_{\rm sync} &= \frac{c\sigma_T}{4\pi}(B_0\gamma)^2 \\
&\simeq 4.96\times10^{-7}\gamma^2 \ {\rm GeV/Gyr} \ , && (3.9)\\
\mathbf{b}_{\rm ic} &= \frac{4}{3}c\sigma_T\omega\gamma^2 \\
&\simeq 2.08\times10^{-7}\gamma^2 \ {\rm GeV/Gyr} \ , && (3.10)\\
\mathbf{b}_{\rm coul} &= 2.7c\sigma_T n_H m_e c^2(6.85+\ln\gamma) \\
&\simeq 0.96\ln\gamma + 6.58 \ {\rm GeV/Gyr} \ , && (3.11)\\
\mathbf{b}_{\rm brem} &= \frac{175.5\alpha c\sigma_T}{8\pi} n_H m_e c^2\gamma \\
&\simeq 0.02\gamma \ {\rm GeV/Gyr} \ , && (3.12)
\end{aligned}
$$

where c is the light speed, σ_T is the Thomson cross-section, α is the fine structure constant. We assume a typical averaged magnetic field strength $B_0 = 4.0\ \mu$G, averaged hydrogen density $n_H = 1.14\ {\rm cm}^{-3}$, and constant background photon field energy density $w = 0.25\ {\rm eV/cm}^3$. Fig. 3.1 presents the CRE energy loss rates as functions of its total energy. Although the energy loss modelling is not very realistic, it catches the basic feature of the dominating mechanisms at different electron energy range. In addition, the toy modelling of an isotropic spatial diffusion coefficient [3] can be defined as

$$
\begin{aligned}
\mathcal{D}(E) &= D_0(\frac{E/{\rm GeV}}{B_0/\mu{\rm G}})^{1/3} \\
&\simeq 3.15\times10^{-2}\gamma^{1/3} \ {\rm kpc^2/Gyr} \ , && (3.13)
\end{aligned}
$$

where $D_0 = 1.0\times10^2\ {\rm kpc^2/Gyr}$. In analogy to the phenomenon where CREs are produced by the supernova explosion, we could roughly describe the source term Q as

$$
\begin{aligned}
Q(E,r) &= Q_0 g_{\rm snr}(E/{\rm GeV})^{-\kappa} \\
&\simeq 1.99\times10^6\exp(-r/h)\gamma^{-2.2}{\rm cm^{-3}GeV^{-3}Gyr^{-1}} \ , && (3.14)
\end{aligned}
$$

with $Q_0 = 1.0\ {\rm cm^{-3}GeV^{-3}Gyr^{-1}}$, $h = 0.5$ kpc, $\kappa = 2.2$, which are chosen for illustrative purpose. Alternatively, we can replace supernova-remnant-driven profile $g_{\rm snr}$ by a WIMP-annihilation-driven profile

$$
g_{\rm dm}^2 = 2.56\frac{h^6}{(h+r)^2(h^2+r^2)^2} \ , \qquad (3.15)
$$

which is known the Burkert profile [2] for dark matter distribution in dwarf galaxies, where the square comes from how we estimate the annihilation cross-section and the constant 2.56 is set in this example for normalizing the total source density with respect to $g_{\rm snr}$.

Differs from the testing case for spatial diffusion with spectral advection, here we have additional geometric tensors $\mathcal{T}_r = r^{-2}$ and $\mathcal{T}_q = E^{-3}$. The raw (before applying the

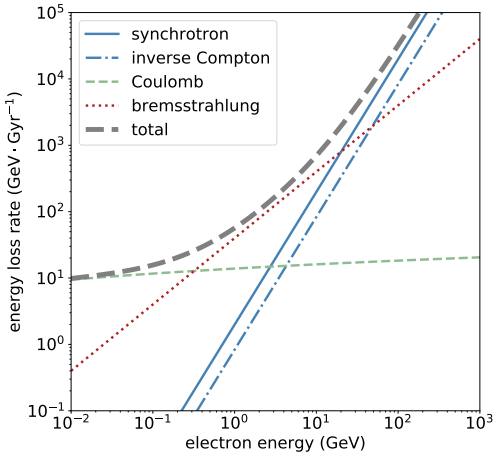

Figure 3.1: CRE continuous energy loss rate in various mechanisms defined in the toy modelling. At low energy scale Coulomb interaction loss (dashed green curve) dominates until around 0.1 GeV level, from where Bremsstrahlung loss (dotted red curve) takes over. When CRE energy goes higher than 10 GeV magnitude, synchrotron loss (solid blue curve) and inverse Compton loss (dash-dot blue curve) become dominant mechanisms.

upwind method) weak formulation of the time-independent left-hand-side consequently has more terms in the spatial domain, which reads

$$\begin{aligned}
&\sum_k \sum_{\beta,j} \mathcal{U}_k^{\beta j} \int_{\Omega_r} (v_i v_j) \Bigg[\oint_{\partial\Omega_q^k} w_\alpha (\mathcal{T}_q \tilde{\mathbf{b}}) \cdot \hat{n}_q w_\beta \\
&- \int_{\Omega_q^k} (\nabla_q w_\alpha^k) \cdot (\mathcal{T}_q \tilde{\mathbf{b}}) w_\beta^k - \int_{\Omega_q^k} w_\alpha^k (\nabla_q \mathcal{T}_q) \cdot \tilde{\mathbf{b}} w_\beta^k \Bigg] \\
&+ \int_{\Omega_q^k} (w_\alpha^k w_\beta^k) \Bigg[\int_{\Omega_r} (\mathcal{T}_r \nabla_r v_i + v_i \nabla_r \mathcal{T}_r) \cdot (\tilde{\mathcal{D}} \nabla_r v_j) \Bigg] ,
\end{aligned} \tag{3.16}$$

where the effective advection coefficient $\tilde{\mathbf{b}} = E^2 \mathbf{b}$, and the effective diffusion coefficient $\tilde{\mathcal{D}} = r^2 \mathcal{D}$.

Fig.3.2 presents the spectral and spatial behaviour of the steady state solutions. The energy spectrum exhibits the expected steepening below 1 GeV due to the transition from diffusion to advection domination and 10 GeV due to the transition of dominant continuous energy loss mechanism illustrated by Fig.3.1. Since the source term Q faces spatial suppression, CRE spectral steepening occurs around lower energy scale and becomes more smooth at higher radii. Meanwhile, the radial flattening in the dark-matter (DM) induced CRE spectral distribution follows the fact that DM induced modelling provides more CREs at high radii than supernova-remnant (SNR) induced modelling.

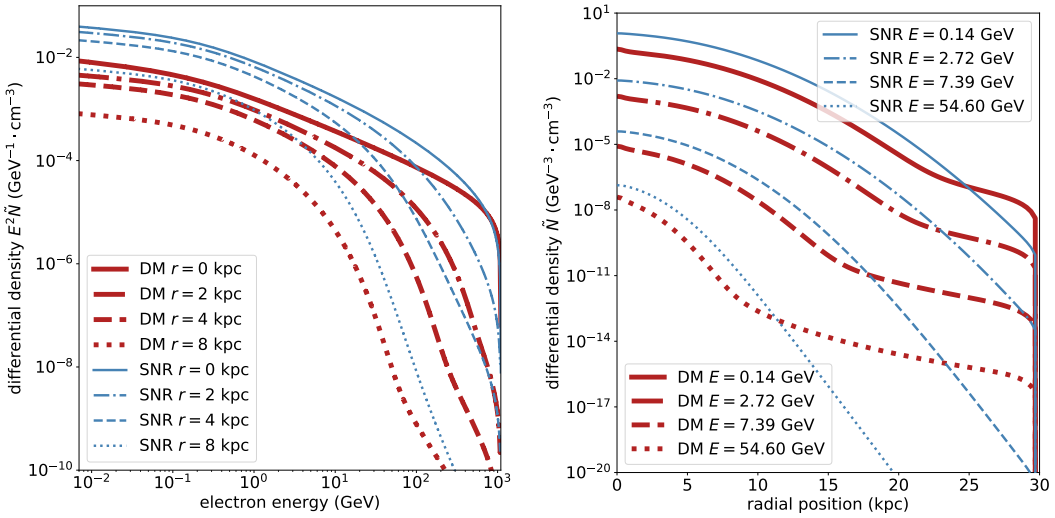

Figure 3.2: Spectral (left) and spatial (right) distribution of CRE differential density $E^2\tilde{N}$ at different radial and energy positions. Thick (red) curves represent results from CRE source distribution in analogy to DM annihilation while thin (blue) curves are from source distribution in analogy to supernova remnants.

CONCLUSION

As demonstrated above, we have successfully built up the framework for handling the high-dimensional PDE system. The multi-threading speedup and precision in solving simple advection-diffusion problems has been examined. We emphasize that this toolkit itself is not fully incomplete from a technical point of view, where we need further MPI parallelism and matrix free method in assembling the system matrix representatives. Towards its application in realistic and complicated CRE propagation, CR-GMF co-evolution and even the RMHD system, we need to implement more auxiliary back-end functions, especially a hyper-propagator class that consists of several single PDE objects. Technically in terms of the solving scheme, we can try to implement the goal-oriented adaptive refinement method [10], and besides, the non-linear PDE system needs extra caution. In the end we should connect the `BIFET` toolkit into either the `hammurabi X` package or directly into the `IMAGINE` engine in order to realize our conceptual picture of consistent simulation and analysis of the Galactic synchrotron emission.

Bibliography

[1] P. Blasi, E. Amato, and P. D. Serpico. Spectral breaks as a signature of cosmic ray induced turbulence in the Galaxy. *Physical Review Letters*, 109(6):061101, jul 2012. ISSN 0031-9007. doi: 10.1103/PhysRevLett.109.061101.

[2] A. Burkert. The Structure of Dark Matter Halos in Dwarf Galaxies. *Symposium - International Astronomical Union*, 171(1):175–178, jul 1996. ISSN 0074-1809. doi: 10.1017/s0074180900232324.

[3] C. Evoli, D. Gaggero, A. Vittino, G. D. Bernardo, M. D. Mauro, A. Ligorini, P. Ullio, and D. Grasso. Cosmic-ray propagation with DRAGON2: I. numerical solver and astrophysical ingredients. *Journal of Cosmology and Astroparticle Physics*, 2017(02):015–015, feb 2017. ISSN 1475-7516. doi: 10.1088/1475-7516/2017/02/015.

[4] C. Evoli, P. Blasi, G. Morlino, and R. Aloisio. Origin of the Cosmic Ray Galactic Halo Driven by Advected Turbulence and Self-Generated Waves. *Physical Review Letters*, 121(2):021102, jul 2018. ISSN 0031-9007. doi: 10.1103/PhysRevLett.121.021102.

[5] R. Farber, M. Ruszkowski, H. Y. K. Yang, and E. G. Zweibel. Impact of Cosmic Ray Transport on Galactic Winds. *The Astrophysical Journal*, 856(2):112, jul 2017. ISSN 1538-4357. doi: 10.3847/1538-4357/aab26d.

[6] I. A. Grenier, J. H. Black, and A. W. Strong. The Nine Lives of Cosmic Rays in Galaxies. *Annual Review of Astronomy and Astrophysics*, 53(1):199–246, aug 2015. ISSN 0066-4146. doi: 10.1146/annurev-astro-082214-122457.

[7] E. Heintz and E. G. Zweibel. The Parker Instability with Cosmic Ray Streaming. *The Astrophysical Journal*, 860(2):97, mar 2018. ISSN 1538-4357. doi: 10.3847/1538-4357/aac208.

[8] R. Kissmann. PICARD: A novel code for the Galactic Cosmic Ray propagation problem. *Astroparticle Physics*, 55(9):37–50, mar 2014. ISSN 09276505. doi: 10.1016/j.astropartphys.2014.02.002.

[9] R. M. Kulsrud and C. J. Cesarsky. The Effectiveness of Instabilities for the Confinement of High Energy Cosmic Rays in the Galactic Disk. *Astrophysical Letters*, 8:189, Mar. 1971.

[10] J. Oden and S. Prudhomme. Goal-oriented error estimation and adaptivity for the finite element method. *Computers & Mathematics with Applications*, 41(5-6): 735–756, mar 2001. ISSN 08981221. doi: 10.1016/S0898-1221(00)00317-5.

[11] C. Pfrommer, R. Pakmor, K. Schaal, C. M. Simpson, and V. Springel. Simulating cosmic ray physics on a moving mesh. *Monthly Notices of the Royal Astronomical Society*, 465(4):4500–4529, mar 2017. ISSN 13652966. doi: 10.1093/mnras/stw2941.

[12] C. C. Popescu, R. Yang, R. J. Tuffs, G. Natale, M. Rushton, and F. Aharonian. A radiation transfer model for the Milky Way: I. Radiation fields and application to high-energy astrophysics. *Monthly Notices of the Royal Astronomical Society*, 470(3):2539–2558, sep 2017. ISSN 0035-8711. doi: 10.1093/mnras/stx1282.

[13] M. J. Rees. Proton Synchrotron Emission from Compact Radio Sources. *Astrophysical Letters*, 2:1, 1968.

[14] M. Ruszkowski, H. Y. K. Yang, and E. Zweibel. Global simulations of galactic winds including cosmic ray streaming. *The Astrophysical Journal*, 834(2):208, feb 2016. ISSN 1538-4357. doi: 10.3847/1538-4357/834/2/208.

[15] R. Schlickeiser. *Cosmic ray astrophysics*. Springer, 2002. ISBN 3540664653.

[16] A. W. Strong and I. V. Moskalenko. Propagation of Cosmic-Ray Nucleons in the Galaxy. *The Astrophysical Journal*, 509(1):212–228, dec 1998. ISSN 0004-637X. doi: 10.1086/306470.

[17] A. W. Strong, I. V. Moskalenko, and V. S. Ptuskin. Cosmic-ray propagation and interactions in the Galaxy. *Annual Review of Nuclear and Particle Science*, 57(1): 285–327, jan 2007. ISSN 0163-8998. doi: 10.1146/annurev.nucl.57.090506.123011.

[18] M. Tanabashi, K. Hagiwara, K. Hikasa, K. Nakamura, Y. Sumino, F. Takahashi, J. Tanaka, K. Agashe, G. Aielli, C. Amsler, M. Antonelli, D. M. Asner, H. Baer, S. Banerjee, R. M. Barnett, T. Basaglia, C. W. Bauer, J. J. Beatty, V. I. Belousov, J. Beringer, S. Bethke, A. Bettini, H. Bichsel, O. Biebel, K. M. Black, E. Blucher, O. Buchmuller, V. Burkert, M. A. Bychkov, R. N. Cahn, M. Carena, A. Ceccucci, A. Cerri, D. Chakraborty, M.-C. Chen, R. S. Chivukula, G. Cowan, O. Dahl, G. D'Ambrosio, T. Damour, D. de Florian, A. de Gouvêa, T. DeGrand, P. de Jong, G. Dissertori, B. A. Dobrescu, M. D'Onofrio, M. Doser, M. Drees, H. K. Dreiner, D. A. Dwyer, P. Eerola, S. Eidelman, J. Ellis, J. Erler, V. V. Ezhela, W. Fetscher, B. D. Fields, R. Firestone, B. Foster, A. Freitas, H. Gallagher, L. Garren, H.-J. Gerber, G. Gerbier, T. Gershon, Y. Gershtein, T. Gherghetta, A. A. Godizov, M. Goodman, C. Grab, A. V. Gritsan, C. Grojean, D. E. Groom, M. Grünewald, A. Gurtu, T. Gutsche, H. E. Haber, C. Hanhart, S. Hashimoto, Y. Hayato, K. G. Hayes, A. Hebecker, S. Heinemeyer, B. Heltsley, J. J. Hernández-Rey, J. Hisano, A. Höcker, J. Holder, A. Holtkamp, T. Hyodo, K. D. Irwin, K. F. Johnson, M. Kado, M. Karliner, U. F. Katz, S. R. Klein, E. Klempt, R. V. Kowalewski, F. Krauss, M. Kreps, B. Krusche, Y. V. Kuyanov, Y. Kwon,

O. Lahav, J. Laiho, J. Lesgourgues, A. Liddle, Z. Ligeti, C.-J. Lin, C. Lippmann, T. M. Liss, L. Littenberg, K. S. Lugovsky, S. B. Lugovsky, A. Lusiani, Y. Makida, F. Maltoni, T. Mannel, A. V. Manohar, W. J. Marciano, A. D. Martin, A. Masoni, J. Matthews, U.-G. Meißner, D. Milstead, R. E. Mitchell, K. Mönig, P. Molaro, F. Moortgat, M. Moskovic, H. Murayama, M. Narain, P. Nason, S. Navas, M. Neubert, P. Nevski, Y. Nir, K. A. Olive, S. Pagan Griso, J. Parsons, C. Patrignani, J. A. Peacock, M. Pennington, S. T. Petcov, V. A. Petrov, E. Pianori, A. Piepke, A. Pomarol, A. Quadt, J. Rademacker, G. Raffelt, B. N. Ratcliff, P. Richardson, A. Ringwald, S. Roesler, S. Rolli, A. Romaniouk, L. J. Rosenberg, J. L. Rosner, G. Rybka, R. A. Ryutin, C. T. Sachrajda, Y. Sakai, G. P. Salam, S. Sarkar, F. Sauli, O. Schneider, K. Scholberg, A. J. Schwartz, D. Scott, V. Sharma, S. R. Sharpe, T. Shutt, M. Silari, T. Sjöstrand, P. Skands, T. Skwarnicki, J. G. Smith, G. F. Smoot, S. Spanier, H. Spieler, C. Spiering, A. Stahl, S. L. Stone, T. Sumiyoshi, M. J. Syphers, K. Terashi, J. Terning, U. Thoma, R. S. Thorne, L. Tiator, M. Titov, N. P. Tkachenko, N. A. Törnqvist, D. R. Tovey, G. Valencia, R. Van de Water, N. Varelas, G. Venanzoni, L. Verde, M. G. Vincter, P. Vogel, A. Vogt, S. P. Wakely, W. Walkowiak, C. W. Walter, D. Wands, D. R. Ward, M. O. Wascko, G. Weiglein, D. H. Weinberg, E. J. Weinberg, M. White, L. R. Wiencke, S. Willocq, C. G. Wohl, J. Womersley, C. L. Woody, R. L. Workman, W.-M. Yao, G. P. Zeller, O. V. Zenin, R.-Y. Zhu, S.-L. Zhu, F. Zimmermann, P. A. Zyla, J. Anderson, L. Fuller, V. S. Lugovsky, and P. Schaffner. Review of Particle Physics. *Physical Review D*, 98(3):030001, aug 2018. ISSN 2470-0010. doi: 10.1103/PhysRevD.98.030001.

[19] J. Wiener, S. P. Oh, and E. G. Zweibel. Interaction of Cosmic Rays with Cold Clouds in Galactic Halos. *Monthly Notices of the Royal Astronomical Society*, 467 (1):stx109, oct 2016. ISSN 0035-8711. doi: 10.1093/mnras/stx109.