Simulating 3D Radiation Transport

a modern approach to discretisation and an exploration of probabilistic methods

Frederik De Ceuster

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
of
University College London.

Department of Physics and Astronomy
University College London

11th January 2022

I, Frederik De Ceuster, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Dedicated to everyone who ever asked me a question,

or ever tried to explain or teach me something.

Written for myself,
five years ago.

Abstract

Light, or electromagnetic radiation in general, is a profound and invaluable resource to investigate our physical world. For centuries, it was the only and it still is the main source of information to study the Universe beyond our planet. With high-resolution spectroscopic imaging, we can identify numerous atoms and molecules, and can trace their physical and chemical environments in unprecedented detail. Furthermore, radiation plays an essential role in several physical and chemical processes, ranging from radiative pressure, heating, and cooling, to chemical photo-ionisation and photo-dissociation reactions. As a result, almost all astrophysical simulations require a radiative transfer model. Unfortunately, accurate radiative transfer is very computationally expensive. Therefore, in this thesis, we aim to improve the performance of radiative transfer solvers, with a particular emphasis on line radiative transfer. First, we review the classical work on accelerated lambda iterations and acceleration of convergence, and we propose a simple but effective improvement to the ubiquitously used Ng-acceleration scheme. Next, we present the radiative transfer library, Magritte: a formal solver with a ray-tracer that can handle structured and unstructured meshes as well as smoothed-particle data. To mitigate the computational cost, it is optimised to efficiently utilise multi-node and multi-core parallelism as well as GPU offloading. Furthermore, we demonstrate a heuristic algorithm that can reduce typical input models for radiative transfer by an order of magnitude, without significant loss of accuracy. This strongly suggests the existence of more efficient representations for radiative transfer models. To investigate this, we present a probabilistic numerical method for radiative transfer that naturally allows for uncertainty quantification, providing us with a mathematical framework to study the trade-off between computational speed and accuracy. Although we cannot yet construct optimal representations for radiative transfer problems, we point out several ways in which this method can lead to more rigorous optimisation.

Impact statement

The main goal of this thesis is to optimise the trade-off between computational speed and accuracy in radiative transfer computations. Radiation transport plays an undeniably crucial role throughout astrophysics, but also has many applications outside astronomy, ranging from medical imaging to nuclear engineering. Therefore, the impact of the results presented in this thesis can reach far beyond the astrophysical context in which they were developed. In particular, we identify the following four main contributions of this work:

- An improved version of Ng-acceleration of convergence for iterative processes, which is ubiquitously used in radiative transfer applications (see Section 2.3.3). The proposed improvements are not restricted to radiative transfer applications and thus can be used to accelerate convergence in any type of iterative process.

- The Magritte open-source software library for radiative transfer modelling and synthetic observations (see Chapter 3). This library can leverage both multi-node and multi-core parallelism as well as GPU offloading, and is among other state-of-the-art solvers currently being used in astrophysical research (see Chapter 4). Moreover, due to its modular design, if not as a whole, the individual modules of Magritte, such as e.g. the ray-tracer, can still readily be used in many other applications dealing with unstructured data.

- The ParAcAbs open-source C++ headers for parallelisation and acceleration abstractions (see Chapter 3). This interface with corresponding data structures allows one to implement loop parallelisation and GPU offloading in a portable way, and can readily be used in any C++ program.

- A heuristic algorithm that allows to reduce the size of a typical radiative transfer input model by about an order of magnitude without significant loss of accuracy on the output, thus reducing the computational cost of radiative transfer simulations by more than an order of magnitude (see Chapter 5).

Apart from these four main contributions, this thesis, moreover, has much potential to impact future research, especially in the use of probabilistic numerical methods for radiative transfer (see Chapter 6). We demonstrated the potential of these methods for optimisation and uncertainty quantification, and outlined several directions for future research.

Acknowledgements

Many of the early breakthroughs in astronomy arose from the insight that humans have no special place or role in our Universe. Although this might be true for our physical Universe, it certainly is not for my personal Universe, which can only be what it is, because many people are who they are.

First and foremost, I would like to express my sincere gratitude to my four supervisors: Revd. Dr. Jeremy Yates, Prof. Dr. Leen Decin, Prof. Dr. Peter Boyle, and Dr. James Hetherington, for their persistent faith in me, for the freedom to explore my own research paths, and for their encouraging guidance along the way. I would also like to thank my examiners: Dr. Malcolm Gray and Prof. Dr. Jason McEwen, for their effort in reviewing my thesis and for the kind and interesting discussions during my viva.

Furthermore, I would like to thank all the past and present members of team L.E.E.N. and the Institute of Astronomy at KU Leuven. I feel extremely privileged to have had such a warm and welcoming work environment, and am honoured to call many of you my friends. Also a sincere thank you to Dr. Clare Jenner for the delightful collaboration at DiRAC, all the help and advice, and the continued encouragement for my research. Many thanks also go to James Legg and Marcus Kiel for the insightful discussions and ploughing through some books with me.

I would also like to thank Intel Corporation for their interest in my research, and for providing the necessary funding. At this point, I would like to emphasise my gratitude towards Prof. Dr. Leen Decin and Revd. Dr. Jeremy Yates, especially for their initial faith in me, when providing me with this opportunity, and also thank Prof. Dr. Jean-Pierre Locquet for his faith and understanding during our brief collaboration that preceded this project. It was only along the way that I realised that my seemingly curved career path is actually a geodesic, and hence optimal in many aspects, which only further demonstrates your massive impact on my Universe.

Fortunately, there is also more to life than science. So, finally, I would like to thank Eva, who not only manages to tolerate my quirky being, but celebrates it by joining me in discovering life beyond science.

Contents

List of Figures

List of Tables

Chapter 1

Introductory material

What I cannot create, I do not understand.

– Richard P. Feynman

1.1 Prologue

Light, or electromagnetic radiation in general, is a profound and invaluable carrier of information that allows humans, animals, and machines alike to perceive their physical environments. From the mundane assessment of the milk fraction in your tea, based on its colour, to the extraordinary endeavour of determining the chemical composition of (astrophysical) objects, based on their atomic and molecular spectra, it is light, through its well-understood interactions with its medium, that allows us to meticulously study and perhaps even understand most of the world around us. Moreover, especially in astrophysics and cosmology, electromagnetic radiation is often the only source of information available. With the noteworthy exceptions of cosmic rays [1], neutrinos [2], gravitational waves [3], and space missions [4], our understanding of the Universe is in large part solely based on observations of light. Given the ever more observations made with ever improving telescopes, one thus might hope to be assured that one day we might be able to understand the inner workings of the Universe and our place within it.

Unfortunately, as often, *in between dreams and deeds, laws stand in the way, and practical difficulties*[1]. The *laws* that govern electromagnetism are currently not the main problem anymore. Although they eluded scientists for centuries, the

[1]After *Het Huwelijk* (1910) by the Belgian writer and poet Willem Elsschot (1882 - 1960).

unification of electricity and magnetism by Maxwell [5], followed by the quantum mechanical description by Planck [6–8], Einstein [9], and contemporaries, finally culminated in what is now the most accurate and precisely tested scientific theory ever: quantum electrodynamics [10–12]. What remains, however, are the *practical difficulties*. After all, what does it mean to *know the physics*? A glimpse, for instance, at the intricate structures that emerge from the simple *laws* in John Conway's Game of Life[2] [13] already demonstrates that even perfect knowledge of the underlying rules is not enough to understand or even imagine all possible consequences of a system. It is, however, enough to *simulate* the system. Hence, despite the more philosophical questions about the intrinsic value of our understanding of physics, we can at least aim for the ability to simulate or re-create, thus satisfying Feynman's necessary condition for understanding, and also, when done efficiently, even aim for the ability to anticipate behaviour and make predictions. This (perhaps weakened) form of understanding through simulation certainly holds powerful promises. However, simulations also entail new practical difficulties which, without proper care, can turn into fundamental difficulties, for instance regarding the obtainable accuracy with practically feasible computation time and resources.

Today in astrophysics and cosmology, but especially in the study of stellar and planetary atmospheres, we find ourselves fairly confident in our basic understanding of most of the microscopic physics and chemistry. Nevertheless, we are facing great challenges with *practical difficulties* when it comes to identifying the dominant mechanisms to explain (or postdict), let alone predict, observations. A key difficulty is that our observations often show the combined result of various different processes resulting in a convoluted view that is difficult to disentangle and interpret. Moreover, the situation is further complicated since radiation also plays an active role in the evolution of the medium, as it can provide pressure, an efficient heating or cooling mechanism, and affect chemical reactions. This intricate interplay between electromagnetic radiation and its medium, and the ways in which the underlying physical and chemical processes are revealed in observations are studied under

[2]Try it yourself, for instance, at `playgameoflife.com`.

the denominator: *radiative transfer*, a field initiated by giants like Schuster [14], Schwarzschild [15], Eddington [16], and Jeans [17], that evolved from being a highly mathematical endeavour (see, e.g. the work by Chandrasekhar [18]) to being a pillar of modern computational physics (see e.g. [19]).

In this thesis, we aim to address some of the practical difficulties encountered in radiative transfer by leveraging the opportunities provided by modern computing.

1.2 The radiating Universe

Electromagnetic radiation plays a crucial role in our Universe. According to general relativity and the Big Bang hypothesis, electromagnetic radiation dominated[3] the evolution of our Universe in the first tens of thousands of years after the Big Bang and the elusive inflationary phase. Then, while electrons and protons (re)combined about 300 000 years later, radiation decoupled from matter, out of equilibrium, and became more interesting as the Universe became transparent, since light could now carry information because its properties correlated with the particular path it travelled through the incipient inhomogeneous medium. This first light can still be observed as the cosmic (now microwave) background radiation. In the billions of years of cosmic evolution that followed, electromagnetic radiation continued its large scale impact as the light of the first stellar and quasi-stellar objects re-ionised the highly abundant neutral hydrogen, rendering the Universe opaque again in certain frequency ranges. Also on smaller scales, radiation remained to have a distinct impact where it can exert a radiative pressure or act as a very efficient heating or cooling mechanism, for instance, in the stellar winds that arise from evolved stars (see e.g. [21], the references therein, and also Chapter 4). Furthermore, various photo-dissociation and photo-ionisation reactions can have a significant impact on chemistry (see e.g. [22–25]). Electromagnetic radiation thus actively shaped the evolution of the Universe and its contents, until at last (at least) also humans discovered its use and started to describe and decipher the light to study their place as observers.

[3]Electromagnetic radiation dominated in terms of energy density in approximately equal shares with neutrinos, and potentially with other relativistic particles (see e.g. [20]).

1.3 Radiative transfer theory

Light, or electromagnetic radiation in general, can be described in many different yet compatible ways, ranging from waves to particles, or unified as quantized excitations of the electromagnetic field. In the theory of radiation transport, colloquially referred to as *radiative transfer*, electromagnetic radiation is described on an intermediate level as a directed stream of energy. The radiation field is characterised by the *specific monochromatic intensity*, $I_\nu(\boldsymbol{x},\hat{\boldsymbol{n}})$, i.e. the energy transported in a certain direction in a certain frequency bin. This is a function of frequency[4],ν, position, $\boldsymbol{x}$, and direction, $\hat{\boldsymbol{n}}$, as characterised by a unit vector.

1.3.1 The radiative transfer equation

Every interaction between the radiation field and the medium can be characterised by a corresponding change in the specific monochromatic intensity, as described by the *radiative transfer equation* in the observer frame [18, 26],

$$\begin{aligned}\hat{\boldsymbol{n}}\cdot\nabla I_\nu(\boldsymbol{x},\hat{\boldsymbol{n}}) \;&=\; \eta_\nu(\boldsymbol{x}) \;-\; \chi_\nu(\boldsymbol{x})\, I_\nu(\boldsymbol{x},\hat{\boldsymbol{n}}) \\ &\quad+ \oint \mathrm{d}\Omega' \int_0^\infty \mathrm{d}\nu'\; \Phi_{\nu\nu'}(\boldsymbol{x},\hat{\boldsymbol{n}},\hat{\boldsymbol{n}}')\, I_{\nu'}(\boldsymbol{x},\hat{\boldsymbol{n}}').\end{aligned} \tag{1.1}$$

This equation relates the change in specific monochromatic intensity, $I_\nu(\boldsymbol{x},\hat{\boldsymbol{n}})$, along a ray in direction, $\hat{\boldsymbol{n}}$, and indicated by a directional derivative, $\hat{\boldsymbol{n}}\cdot\nabla$, to the local emissivity, $\eta_\nu(\boldsymbol{x})$, and opacity, $\chi_\nu(\boldsymbol{x})$, of the medium. The emissivity quantifies what is gained in intensity, whereas the opacity quantifies what is lost as a fraction of the intensity. Scattering introduces an extra contribution, characterised by the redistribution function, $\Phi_{\nu\nu'}(\boldsymbol{x},\hat{\boldsymbol{n}},\hat{\boldsymbol{n}}')$, which gives the probability for radiation of frequency, ν', incoming along direction, $\hat{\boldsymbol{n}}'$, with a corresponding infinitesimal solid angle, $\mathrm{d}\Omega'$, to be scattered in direction, $\hat{\boldsymbol{n}}$, and shifted to frequency, ν.

The heuristic interpretation of the transfer equation is simple enough, as it sums the different contributions to the change in intensity. Nevertheless, it can also be derived more formally from conservation of energy, or as a special case of the

[4]For historical reasons the dependence on frequency, ν, is indicated with a subscript. Throughout this thesis, we define $f_\nu \equiv f(\nu)$ and $f_{\nu'} \equiv f(\nu')$, for any function, f, but only for those subscripts.

Boltzmann transport equation (see e.g. [27]). In general, one could also introduce a time-dependence in the transfer equation, however, throughout this thesis, we will assume that length scales are small enough and time scales are large enough, such that any time-dependence in the radiation field is negligible. Furthermore, we note that we defined the redistribution function in a slightly different way than in [28], such that it incorporates the scattering opacity.

The transfer equation was first formulated in its heuristic form in 1905 by Schuster [14]. However, due to its link with the transport equation its true origin can be traced further back to 1872, with the work of Boltzmann [29]. The latter also highlights the link between radiative transfer and neutron transport, which is of practical interest, for instance, in nuclear engineering. Although this thesis is mostly concerned with the transport of electromagnetic radiation in astrophysical applications, many of the techniques and results presented here can also be applied to other forms of transport. Our formulation and notation here is probably most akin to that of Cannon [30, 31]. Further background on the history of the theory of radiation transport can be found, for instance, in the excellent review by Shore [32].

1.3.2 Optical depth

The coupling between the radiation field and the medium, as described by the transfer equation (1.1), allows for a new and natural measure of distance in radiative transfer. The *optical depth*, originally introduced by Schwarzschild [15], is defined as,

$$\tau_\nu(\boldsymbol{x}_1,\boldsymbol{x}_2) \ \equiv \ \int_{\boldsymbol{x}_1}^{\boldsymbol{x}_2} \text{d}x' \ \chi_\nu(\boldsymbol{x}'), \quad \text{i.e. such that,} \quad \partial_{\tau_\nu} \ = \ \frac{1}{\chi_\nu(\boldsymbol{x})} \ (\hat{\boldsymbol{n}}\cdot\nabla). \tag{1.2}$$

The integral should be interpreted as an integral of the opacity over the light ray (or null geodesic) connecting $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. The optical depth quantifies the amount of material along the line of sight that impedes the propagation of radiation. If the opacity is strictly positive everywhere, it can be interpreted mathematically as a metric. However, as we will see in the next chapter, this is not always the case. We note that, since the opacity is frequency-dependent, also the optical depth, and thus the relevant measure of distance, depends on the considered frequency range.

Since the optical depth is the natural distance measure in radiative transfer, it can be used to elegantly rewrite the transfer equation (1.1) as,

$$\left(1 + \partial_{\tau_\nu} - \mathcal{F}_\nu\right) I_\nu(\boldsymbol{x},\hat{\boldsymbol{n}}) \ = \ S_\nu(\boldsymbol{x}), \tag{1.3}$$

which has the form of a linear integro-differential operator acting on the radiation field. The integral operator accounting for scattering is defined as,

$$\mathcal{F}_\nu(\boldsymbol{x},\hat{\boldsymbol{n}})[\cdot] \ \equiv \ \frac{1}{\chi_\nu(\boldsymbol{x})} \oint \mathrm{d}\Omega' \int_0^\infty \mathrm{d}\nu' \, \Phi_{\nu\nu'}(\hat{\boldsymbol{n}},\hat{\boldsymbol{n}}')[\cdot], \tag{1.4}$$

and the right hand side of equation (1.3), sourcing the differential equation, making it inhomogeneous, is often referred to as the *source function*, and defined as,

$$S_\nu(\boldsymbol{x}) \ \equiv \ \frac{\eta_\nu(\boldsymbol{x})}{\chi_\nu(\boldsymbol{x})}. \tag{1.5}$$

The formulation in terms of the optical depth is particularly useful in the absence of scattering, as it allows us to write a formal solution to the radiative transfer equation.

1.3.3 Formal solution

Consider a domain enclosed by a boundary. Define the path length, s, along a ray originating on the boundary at s_0, such that for some s_1, we have that $s_0 < s < s_1$ lies in the domain. The radiation field, emissivity, opacity, and optical depth can all be (re)parametrised using the path length along the ray. In the absence of scattering, the formal solution to the transfer equation along a ray originating at s_0 reads,

$$I_\nu(s) \ = \ I_\nu^{\mathrm{b}} \, e^{-\tau_\nu(s_0,s)} \ + \int_{s_0}^{s} \mathrm{d}s' \, \eta_\nu(s') \, e^{-\tau_\nu(s',s)}, \tag{1.6}$$

in which I_ν^{b} is the radiation field at the boundary. The formal solution can be interpreted as the sum (or integral) over all contributions to the radiation field along the line of sight, each attenuated by a factor $e^{-\tau_\nu(s',s)}$, accounting for the optical depth between the source of the contribution at s' and the observer at s. Equation (1.6) is only a formal solution, since we still have to evaluate the integral.

1.3.4 Second-order formulation

There are many different equivalent formulations of the transfer equation (1.1) that describe the radiation field. For instance, one can take moments of the transfer equation by integrating out the directional ($\hat{\boldsymbol{n}}$) dependence. This yields an equation relating the mean intensity and flux, and has a particularly nice physical interpretation because of its similarity to the equations of fluid dynamics (see e.g. [27]).

A similar, yet numerically more interesting formulation, originally proposed by Schuster [14] and generalised by Feautrier [33], can by found by describing the radiation field in terms of another set of variables. Define a mean intensity-like (u) and a flux-like quantity (v) along a ray with direction, $\hat{\boldsymbol{n}}$, as,

$$u_\nu(\boldsymbol{x},\hat{\boldsymbol{n}}) \equiv \frac{1}{2}\Big(I_\nu(\boldsymbol{x},\hat{\boldsymbol{n}}) + I_\nu(\boldsymbol{x},-\hat{\boldsymbol{n}})\Big), \tag{1.7}$$

$$v_\nu(\boldsymbol{x},\hat{\boldsymbol{n}}) \equiv \frac{1}{2}\Big(I_\nu(\boldsymbol{x},\hat{\boldsymbol{n}}) - I_\nu(\boldsymbol{x},-\hat{\boldsymbol{n}})\Big). \tag{1.8}$$

Equivalently, we also define a new operator to represent the scattering contribution,

$$\mathcal{P}^{\pm}_\nu(\boldsymbol{x},\hat{\boldsymbol{n}})[\cdot] \equiv \Big(\mathcal{F}_\nu(\boldsymbol{x},\hat{\boldsymbol{n}}) \pm \mathcal{F}_\nu(\boldsymbol{x},-\hat{\boldsymbol{n}})\Big)[\cdot]. \tag{1.9}$$

From here onward, we drop all ν, $\boldsymbol{x}$, and $\hat{\boldsymbol{n}}$-dependencies for simplicity of notation. We proceed by adding and subtracting the transfer equation (1.1), once for, $\hat{\boldsymbol{n}}$, and once for, $-\hat{\boldsymbol{n}}$. Note that flipping the sign of $\hat{\boldsymbol{n}}$ also flips the sign of the operator ∂_{τ_ν}. This yields a coupled set of linear first-order differential equations in, u, and, v,

$$u + \partial_\tau v - \mathcal{P}^{+}(u+v) = S, \tag{1.10}$$

$$v + \partial_\tau u - \mathcal{P}^{-}(u+v) = 0. \tag{1.11}$$

These equations can neatly be cast into matrix form as,

$$\begin{pmatrix} 1-\mathcal{P}^{+} & \partial_\tau - \mathcal{P}^{+} \\ \partial_\tau - \mathcal{P}^{-} & 1-\mathcal{P}^{-} \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} S \\ 0 \end{pmatrix}. \tag{1.12}$$

In the particular but common case that the scattering redistribution function only depends on the angle between the incoming and outgoing radiation, i.e. $\Phi_{\nu\nu'}(\hat{\boldsymbol{n}},\hat{\boldsymbol{n}}') = \Phi_{\nu\nu'}(\hat{\boldsymbol{n}}\cdot\hat{\boldsymbol{n}}')$, one can show that $\mathcal{P}^{+}v = \mathcal{P}^{-}u = 0$, such that the system simplifies to,

$$\begin{pmatrix} 1-\mathcal{F} & \partial_\tau \\ \partial_\tau & 1-\mathcal{F} \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} S \\ 0 \end{pmatrix}. \tag{1.13}$$

Apart from the obvious coupling between u and v, we should note that these equations also still couple different frequencies, ν and ν', and different directions, $\hat{\boldsymbol{n}}$ and $\hat{\boldsymbol{n}}'$. Therefore, in order to solve this system on a ray-by-ray basis, we need to treat scattering in an iterative way and view it as a contribution to the source function. Defining new scattering source terms, $\Psi^{\pm} \equiv \mathcal{P}^{\pm}[u+v]$, equation (1.12) yields,

$$\begin{pmatrix} 1 & \partial_\tau \\ \partial_\tau & 1 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} S \\ 0 \end{pmatrix} + \begin{pmatrix} \Psi^{+} \\ \Psi^{-} \end{pmatrix}. \tag{1.14}$$

Now the coupling between different rays, i.e. different directions $\hat{\boldsymbol{n}}$, only comes from the second term on the right hand side, $\Psi^{\pm}$. Solving equations (1.10) and (1.11), once for u and once for v, yields a set of linear second-order differential equations, which are the generalised versions of the Schuster-Feautrier equations [14, 33],

$$\begin{pmatrix} 1-\partial_\tau^2 & 0 \\ 0 & 1-\partial_\tau^2 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} 1 & -\partial_\tau \\ -\partial_\tau & 1 \end{pmatrix} \begin{pmatrix} S+\Psi^{+} \\ \Psi^{-} \end{pmatrix}. \tag{1.15}$$

These can be solved for u and v in an iterative way for each ray independently, assuming for $\Psi^{\pm}$ the values obtained in the previous iteration.

In the absence of scattering, i.e. $\Psi^{\pm} = 0$, the generalised Schuster-Feautrier equations for u and v effectively decouple, and reduce to,

$$\left(1-\partial_\tau^2\right) u = S, \tag{1.16}$$

$$\left(1-\partial_\tau^2\right) v = -\partial_\tau S. \tag{1.17}$$

Hence, the radiation field can be computed from (1.16) and deriving v using (1.11), as $v = -\partial_\tau u$. The numerical solution of this system is described in Appendix A.

1.3.5 Lambda iteration

As formulated in (1.1), the radiative transfer equation is an integro-partial differential equation that is in general notoriously difficult to solve. Its solution is furthermore complicated by the fact that in many cases the emissivity, $\eta_\nu(\boldsymbol{x})$, and opacity, $\chi_\nu(\boldsymbol{x})$, depend on the radiation field (see e.g. Chapter 2). This, usually strongly non-linear, coupling means that the transfer equation can only be solved numerically in an iterative way. The resulting iterative process is often referred to as Lambda iteration (or Λ-iteration), after the corresponding Λ-operator, which is generally defined such that it yields the radiation field when acting on the state of the medium,

$$\text{radiation field} \ \equiv \ \Lambda\left[\text{state of the medium}\right], \tag{1.18}$$

while the state of the medium itself depends on the radiation field. The precise definition of the Λ-operator depends on how the state of the medium is described and which form of the radiation field is required. We will give the explicit example for line radiative transfer in Section 2.3.1.

Simple as equation (1.18) may be, it can already illustrate two fronts at which we will battle the radiative transfer problem in this thesis. The radiation field is computed by repeatedly solving (1.18) until convergence, where convergence can be defined, for instance, such that the change in the radiation field between two consecutive solutions must be below a certain threshold. The time it takes to solve the problem is thus proportional to: (I) the time it takes to compute the radiation field given the medium, and (II) the number of iterations it takes for the radiation field to converge. Most of this thesis is concerned with reducing contribution (I), whereas, in Section 2.3, we specifically attempt to improve on (II).

1.3.6 Moving media & the co-moving frame

So far, we only considered the radiative transfer equation in the observer frame (1.1), i.e. when solving the transfer equation, all frequency-dependent quantities have to be considered in the same (observer) frame. In static media, this is trivial, since all observers anywhere in the medium will share the same reference frame. In moving media, however, if two observers are moving with respect to each other, and if they observe light propagating along the direction in which they are moving, they will not agree on its frequency, due to the Doppler shift between their frames.

When solving the radiative transfer equation on a ray-by-ray basis for each point individually (as we will do in Chapter 3), the difference in reference frame can be accounted for by Doppler shifting all frequency-dependent quantities into the frame of the point under consideration. In particular, light being emitted at a frequency, ν, in a point that is moving with a velocity, $\Delta \boldsymbol{v}$, with respect to the observer, looking in direction, $\hat{\boldsymbol{n}}$, will be received at a frequency, ν_0, such that,

$$\frac{\nu}{\nu_0} \ = \ \frac{1 + \Delta \boldsymbol{v} \cdot \hat{\boldsymbol{n}} / c}{\sqrt{1 - \Delta \boldsymbol{v} \cdot \Delta \boldsymbol{v} / c^2}} \ \underset{\Delta v \ll c}{\approx} \ 1 + \frac{\Delta \boldsymbol{v} \cdot \hat{\boldsymbol{n}}}{c}. \tag{1.19}$$

Throughout this thesis, we will assume non-relativistic speeds, $v \ll c$, such that the approximation above is always valid, and can be used to cast all frames into the observer frame.

When solving the radiative transfer equation globally for all points, rays, and frequencies (as we will do in Chapter 6), we cannot shift all frames individually and have to account for the Doppler shifts explicitly in the radiative transfer equation. This can be derived, for instance, from a covariant formulation of the transfer equation (see e.g. [26]) and yields the radiative transfer equation in the co-moving frame. In the non-relativistic limit, accounting only for Doppler shifts and neglecting aberration effects (as we implicitly did before), the co-moving frame version of the transfer equation can be obtained by replacing the directional derivative with,

$$\hat{\boldsymbol{n}} \cdot \nabla \ \rightarrow \ \hat{\boldsymbol{n}} \cdot \nabla \ - \ \big(\hat{\boldsymbol{n}} \cdot \nabla \big(\hat{\boldsymbol{n}} \cdot \boldsymbol{v}(\boldsymbol{x}) / c\big)\big) \, \nu \, \partial_\nu, \tag{1.20}$$

in which, $\boldsymbol{v}(\boldsymbol{x})$, is the velocity field of the medium. The additional term can be interpreted as the change in monochromatic intensity, due to the Doppler shift that is caused by the local gradient in the velocity of the medium.

1.4 Simulating radiation transport

Over the years, several different methods have been devised to solve the radiative transfer problem, from the semi-analytic methods by Chandrasekhar [18] to the computer simulations of today (see e.g. [19]). These simulations can be subdivided into two main categories, based on their solution strategy: Monte Carlo solvers on the one hand, and formal solvers on the other.

1.4.1 Monte Carlo solvers

Monte Carlo solvers mimic the physical photon transport by propagating a large number of virtual photon packets through a model of the medium (see e.g. the review by Noebauer & Sim [34]). The interactions with the medium are determined by a stochastic process sampling from the appropriate probability distributions, while the intensity of the radiation field can be determined by averaging the number of photon packets propagating in a certain direction in a certain frequency bin. Some examples of Monte Carlo solvers that are used in astrophysical applications are: RadMC-3D [35], Skirt [36], CMacIonize [37], and some components of Torus [38]. The main issue with this approach is that the trajectories of these photon packets are randomly determined by the properties of the medium. This implies that they can get trapped in opaque regions, impeding them from contributing much to the overall radiation field. Hence, a large number of packets need to be propagated, which can significantly increase the required computation time. Although many techniques have been devised to avoid the trapping of photon packets (see e.g. [39]), it remains challenging for Monte Carlo solvers to efficiently obtain accurate results, especially at medium to high optical depths [40]. Nevertheless, these solvers remain popular for their ease of implementation and their evocative physical interpretation.

1.4.2 Formal solvers

Formal solvers compute the radiation field by directly solving the radiative transfer equation, given a model of the medium. This can be done in several different ways. A particularly popular way is to use ray-tracing, i.e. by tracing rays through a model of the medium and solving the transfer equation along each of those rays, such as e.g.

in SPHRAY [41], 3D-PDR [42], and LAMPRAY [43]. Alternatively, also finite element methods have been applied to discretise the transfer equation on the computational domain, such as e.g. in [44–46]. Finite element solvers are especially useful when scattering is important, since they can naturally couple the radiation field along different directions, whereas ray-tracing solvers only couple the radiation field along individual rays, and thus require an iterative solution scheme to allow for scattering.

1.4.3 Hybrids

Furthermore, there are also hybrid solvers that combine both Monte Carlo and formal methods, such as RATRAN [47] and its 3D successor LIME [48]. The latter has been widely used to model atomic and molecular lines in 3D models of various astrophysical objects (see e.g. [49–56]). Both RATRAN and LIME will be considered state-of-the-art solvers in terms of their accuracy and will be used to benchmark against in this thesis.

The evolution from (semi-)analytic models to computer simulations introduced a dependency on the available computing resources. As a result, when building a model, considerations about ease of calculation were replaced with considerations about the ease of implementation as a computer program, and the ability to leverage the numerous processing units and accelerators distributed over the various nodes in modern (super)computers. Therefore it is key to view any development in radiative transfer simulation in the context of the landscape of modern computing.

1.5 The landscape of modern computing

While, in the last century, our mathematical and physical understanding of radiation transport problems has grown quite steadily, the power, variety, and availability of the computational tools to solve them has been and still is drastically growing. This rapid growth bears several opportunities but challenges too, also for radiative transfer research. We distinguish here between paradigm shifts in hardware and software.

1.5.1 Hardware

The evolution of modern computer hardware can be characterised by the increasing amount of layers upon layers of *parallelism* and the increasing *heterogeneity* in the available processing units (see e.g. [57]). Parallelism refers to the simultaneous execution of computations to increase the computational throughput, for instance, by dividing a computation over the different compute nodes in a system or the different cores inside a processor. Heterogeneity refers to the use of specialised hardware specifically optimised to perform certain types of computations, such as graphics processing units (GPUs), or field programmable gate arrays (FPGAs).

Up until 2004, the steady increase in processing power was mainly due to the steady increase in the clock speed of processors, commonly referred to as frequency scaling. In each clock cycle a certain number of instructions can be executed, such that an increase in clock speed implies an increase in computing power. The steady increase in clock speed was powered both by Moore's law [58] and Dennard scaling [59]. Moore's law states the observation that the transistor density in dense integrated circuits doubles about every two years[5]. Smaller components result in reduced circuit delays, which in turn allow a higher frequency of operation. Dennard scaling states the observation that smaller components also allow for a decrease in operating voltage, such that the overall power use per surface area remains constant, which is important to guarantee sufficient heat dissipation within the processor.

Nevertheless, in 2004 Intel had to cancel development of two new processors code-named Tejas and Jayhawk due to heat and power consumption problems in their attempt to further increase the clock speed [61]. Consequently, Intel pivoted its focus to the development of dual-core processors, which ushered in the era of multi-core parallel processing. Although frequency scaling had come to an end, single-core performance still improved for certain types of computations, for instance by cleverly interleaving the execution of several instructions (pipelining), or with the use of (ever larger) vector instructions that can act with a single instruction on multiple data items (SIMD). The latter are ideally suited to accelerate the linear algebra required in various multi-media or scientific applications. With the rise of the digital multi-media and gaming industry, vector processors in the form of graphics processing units (GPUs) became mainstream. Moving specialised functionality into

[5]It should be noted that this doubling rate has been revised a few times, from doubling every year in 1965 [58], to doubling every two years after 1980 [60], and about every three years now.

a separate processor while reserving the main processor for more general-purpose computations allows for more aggressive optimisation in the GPU, resulting in more cores with much wider vector lengths.

Although, computer performance steadily keeps increasing through continued innovation, it has become much more difficult to efficiently utilise the different features of a system and reach its peak performance. In the era of frequency scaling, any program would automatically run faster with any increase in clock speed. Now, in order to leverage their power, a program must explicitly be designed to use, for instance, the different cores inside a processor, or to offload computations to a GPU. This highlights the importance of proper software design, and puts a lot of the responsibility to deliver computational performance on the software engineer.

In recent years, several initiatives have emerged that are trying to alleviate this burden and are working towards a more unified programming model, such as for instance, OpenMP[6], Kokkos[7] [62], RAJA[8] [63], and SYCL[9], as implemented e.g. in OneAPI [64]. Currently, there is, unfortunately, not yet much convergence towards a single best programming model or implementation, leaving research software engineers with a plethora of possibilities and trade-offs to consider (see e.g. [65]). As a result, for the time being, portability and in particular performance portability, is not a given, and building computationally intensive software still requires careful design as we will discuss in Section 3.2.

Finally, we note that with the proliferation and increasing bandwidth of the internet, it has become increasingly less important where the computer hardware is hosted, resulting in an increased and much wider availability, and the possibility to concentrate many computing resources into large (academic or commercial) data centres. On the one hand, software engineers could argue that this justifies a specialisation of their work towards a certain class of system, say e.g. clusters of commodity many-core CPUs, since any user will almost always be able to obtain those resources. Furthermore, the availability of newer hardware can decrease the

[6]See also `www.openmp.org`.
[7]See also `github.com/kokkos/kokkos`.
[8]See also `github.com/LLNL/RAJA`.
[9]See also `www.khronos.org/sycl`.

demand for the older commodity hardware, thus further increasing their availability, while newer hardware might be more scarce due to the high demand. Hence, it is not always necessary to follow the latest hardware trends, and targeting a single simple architecture is certainly a valid strategy for applications that already perform well on those resources. On the other hand, however, in order to keep up with the rapidly evolving state-of-the-art in computer hardware, and to be able to achieve peak performance, as required for instance in large-scale astrophysical simulations, it is necessary to evolve the software along with the available hardware.

1.5.1.1 Layers of parallelism & heterogeneity

Below we describe the different levels of parallelism and heterogeneity that can be found in a modern computing cluster and that will be considered later in this thesis.

Message passing on distributed memory The most obvious form of parallelism can be achieved by connecting multiple processors over a network and distribute the computational workload over them. There are several different programming models to manage the work between the different processors, such as e.g. message passing, which is particularly popular in scientific computing. In message passing, as specified e.g. by the message passing interface (MPI) standard [66], the processes communicate with each other by sending messages over the network.

Multi-threading on shared memory Having multiple processing cores allows a machine to process multiple threads of execution (often just referred to as *threads*) in parallel with a single processor, thus increasing its computational throughput. A key aspect of this programming model is that all threads have access to the same, hence shared, memory. Therefore, they can share data and there is no need for explicit communication between the threads. A popular way in scientific computing to implement multi-threading is, for instance, using OpenMP, which provides a set of compiler directives and standardised functions to create threads and share work between them. It should be noted that also the message passing strategy can be applied on shared memory systems. In fact, many MPI implementations provide optimisations to detect and exploit shared memory, avoiding explicit communication.

Vector instructions Single-core performance can also be improved by parallel design, for instance, by using special instructions that can act on multiple[10] data items (SIMD or vector instructions). These are especially useful in scientific computing as they provide intrinsic instructions for vector, matrix or tensor operations. Usually, vectorisation can be left to the compiler, which will attempt to use vector instructions from a certain level of optimisation, or when explicitly specified in the compiler flags. However, one can also explicitly use vector instructions in code by directly calling the intrinsic functions. Most (mathematics) libraries, however, already provide optimised (and vectorised) operations on their objects (see e.g. [67, 68]).

Heterogeneity Many modern computers host, apart from the general-purpose central processing unit (CPU), also other processing units, often referred to as accelerators, for instance, in the form of a graphics processing unit (GPU), tensor processing unit (TPU), or in some specialised cases even field programmable gate arrays (FPGAs). Since these devices have an entirely different architecture, they require additional software layers to provide access to the instruction set and compute elements. A popular example specifically for Nvidia GPUs is CUDA [69].

1.5.2 Software

The evolution of modern software is clearly characterised by a transition towards data-driven or learned algorithms, often broadly labelled as machine learning (ML) or so-called Software 2.0 (see e.g. [70]). Although certainly not every type of computation can be performed (better) in this way, machine learning techniques also increasingly emerge outside the data-intensive and in the compute-intensive industries, for instance, by emulating simulations, or in novel solution methods such as physics-informed neural networks (see also Sections 1.5.2.1 and 1.5.2.2 below). In a sense, this is just the compute-intensive industries following the developments in hardware, since hardware vendors and system designers are increasingly designing and optimising their products specifically targeting machine learning applications. Instead of competing with the hype, they become the hype.

[10]For example, the AVX-512 extensions from the x86 instruction set architecture can act on 512 bits simultaneously, i.e. 16 single-precision or 8 double-precision floating-point numbers.

As examples of machine learning techniques appearing in compute-intensive industries, we briefly discuss emulation and physics-informed neural networks.

1.5.2.1 Emulation instead of simulation

The idea of an emulator is to create a function that (1) can estimate the output of a simulation, and (2) is fast to evaluate. Once the emulator is created, it can then be used to replace the original simulation to obtain results at a lower computational cost. The crux is to find a way to create an emulator that can balance the trade-off between accuracy and evaluation speed. Originally, it was envisioned to do this by simple linear interpolation between a set of known simulation outputs, see e.g. the classic 1989 paper by Sacks et al. [71]. Nowadays, however, impressive results are obtained by interpolating with a neural network, i.e. by training a neural network on a set of input-output pairs generated by the simulation. This has been successfully applied in various compute-intensive applications ranging from chemistry to climate modelling, reporting speed-ups of several orders of magnitude (see e.g. [72–74]).

1.5.2.2 Physics-informed neural networks

Physics-informed neural networks (PINNs) work similarly to emulation, but bypass the need for computationally expensive simulation input-output pairs by directly training the output of the network to satisfy the relevant physical laws [75–77]. For instance, for a radiative transfer simulation with as output the radiation field, the neural network is trained to satisfy the radiative transfer equation in every point, direction, and frequency bin in the model. Practically, this can be enforced by minimising the difference between the left and right hand sides of the transfer equation (see e.g. [78]). This is related to finite element methods in the sense that instead of approximating the desired function with a linear combination of basis functions, it is approximated by a neural network (see also Section 6.5.3).

1.6 Overview of this thesis

This thesis is structured as follows: In the next chapter, Chapter 2, we introduce the theory of atomic and molecular line radiative transfer, and review the classical solution methods. In particular, we introduce the accelerated Lambda iteration scheme by Rybicki & Hummer, and we propose an improved Ng-acceleration of convergence scheme. In Chapter 3, we present MAGRITTE, our newly developed software library for 3D radiative transfer simulation, and review its design and performance. Next, in Chapter 4, we present some applications of MAGRITTE in stellar wind research. In Chapter 5, we study the particular importance of discretisation for the numerical solution of radiative transfer problems, and present a heuristic algorithm to reduce the size of typical input models, thus significantly reducing the computational cost for radiative transfer simulations. In Chapter 6, we introduce a probabilistic view on solving partial differential equations, in particular the radiative transfer equation, and put our heuristic reduction method of the previous chapter on a stronger mathematical footing. Finally, a summary and conclusions of this thesis are formulated in Chapter 7.

The work in this thesis also partially appears in the following three publications:

I F. De Ceuster, J. Yates, P. Boyle, L. Decin, and J. Hetherington. **MAGRITTE: a new multidimensional accelerated general-purpose radiative transfer code.** *Proceedings of the International Astronomical Union*, vol. 14, no. S343, p. 381, 2018. (Reference [79].)

II F. De Ceuster, W. Homan, J. Yates, L. Decin, P. Boyle, and J. Hetherington. **MAGRITTE, a modern software library for 3D radiative transfer – I. Non-LTE atomic and molecular line modelling.** *Monthly Notices of the Royal Astronomical Society*, vol. 492, no. 2, p. 1812, 2019. (Reference [28].)

III F. De Ceuster, J. Bolte, W. Homan, S. Maes, J. Malfait, L. Decin, J. Yates, P. Boyle, and J. Hetherington. **MAGRITTE, a modern software library for 3D radiative transfer – II. Adaptive ray-tracing, mesh construction, and reduction.** *Monthly Notices of the Royal Astronomical Society*, vol. 499, no. 4. p. 5194, 2020. (Reference [80].)

Although there is no strict correspondence between the chapters in this thesis and these papers, roughly, Chapters 2 and 3 are based on Papers I and II, and Chapter 5 is mainly based on Paper III.

Chapter 2

Atomic & molecular line transfer

Lines are very difficult to learn.

– Benedict Cumberbatch

2.1 Introduction

Atomic and molecular lines are the characteristically narrow emission or absorption features that appear in the spectra of atoms and molecules. They are caused by electronic, rotational, or vibrational transitions between the quantized energy levels. The energy difference in these line transitions can be emitted or absorbed as a photon. Since each state, i, has a strictly defined energy, E_i, and since the energy of a photon is related to its frequency, the line transitions have a characteristic frequency, $\nu_{ij} \equiv |E_i - E_j|/h$, in which, $h \approx 6,626 \times 10^{-34}$ J s, is the Planck constant.

Spectral lines played an absolutely crucial role in the development of modern physics and astronomy. Their theoretical interpretation was key in the development of quantum mechanics and the discovery of the atomic structure (see e.g. the work by Bohr [81]). Since the configuration of energy levels is unique for each atom or molecule, they all have a characteristic set of possible transitions with corresponding photon frequencies. This makes them an exquisite diagnostic tool that allows us to identify atoms and molecules for various applications on Earth, but also in particular in space. See, for instance, the heroic PhD thesis by Payne [82] reporting the detection of various elements in stellar atmospheres in 1925, and the first detections of molecules in space by Douglas & Herzberg [83], and Weinreb et al. [84]. In addition to merely identifying atoms and molecules, lines can also trace the physical and chemical conditions of the medium they pass through. Their narrow extent in frequency space makes them particularly sensitive to Doppler shifts, which allows us to extract the velocity structure of the medium along the line of sight. Furthermore, the width of the lines is determined by the thermal and turbulent motions of the medium, and the combined information about the abundances of several species hints at the possible chemistry taking place. All this made spectroscopic analysis one of the most fruitful topics in modern astronomy.

However, since such a wealth of information is encoded in spectral lines, line formation is a very intricate process. It is furthermore complicated by the relatively large speed of light ($c \approx 299\ 792\ 458$ m/s), causing almost instantaneous coupling between the radiative processes over large parts of the medium that, hence, all require a self-consistent solution.

In this chapter, we describe the different contributions to the coupling between radiation field and medium, and the different ways to self-consistently solve it.

2.2 Physical problem

In the following, we outline the mathematical description of line radiative transfer and indicate the various assumptions we make. This is far from a complete description of the subject, but gives the minimal ingredients required to study the critical issues in line radiative transfer that we aim to address in this thesis, and the models that we apply them to. A comprehensive description with derivations from first principles can be found, for instance, in Hubeny & Mihalas [26].

2.2.1 Line emissivity & opacity

The *emissivity* and *opacity* resulting from a line transition between a higher energy level, i, and a lower level, j, are given in terms of the Einstein A_{ij}, B_{ji}, and B_{ij} coefficients, and the populations, $n_i(\boldsymbol{x})$, of the quantized energy levels, such that,

$$\eta_\nu^{ij}(\boldsymbol{x}) = \frac{h\nu}{4\pi}\,\phi_\nu^{ij}(\boldsymbol{x})\,n_i(\boldsymbol{x})\,A_{ij}, \tag{2.1}$$

$$\chi_\nu^{ij}(\boldsymbol{x}) = \frac{h\nu}{4\pi}\,\phi_\nu^{ij}(\boldsymbol{x})\,\big(n_j(\boldsymbol{x})\,B_{ji} - n_i(\boldsymbol{x})\,B_{ij}\big), \tag{2.2}$$

where, A_{ij}, and, B_{ij}, account, respectively, for spontaneous and stimulated emission and, B_{ji}, accounts for absorption. The function, $\phi_{\nu}^{ij}(\boldsymbol{x})$, describes the line profile (see Section 2.2.2). Note that, since the resulting stimulated emission is proportional to the intensity, just like an opacity, it is treated here as negative absorption. Equations (2.1) and (2.2) thus couple the state of the quantized energy levels of the atoms and molecules in the medium to the radiation field, through the emissivity and opacity that appear in the radiative transfer equation (1.1).

2.2.2 Line profile function

Both line emissivity and opacity are proportional to the *line profile function*, $\phi_{\nu}^{ij}(\boldsymbol{x})$, which describes the distribution of the line in frequency space. Note that in equations (2.1) and (2.2), we assumed complete frequency redistribution, such that the emission and absorption line profile are equal. This common assumption is valid as long as the considered densities and temperatures are high enough such that sufficiently frequent collisions can de-correlate emission and absorption events.

There are several physical mechanisms that can cause the line profile to deviate from an idealised delta distribution, $\phi_{\nu}^{ij} = \delta(\nu - \nu_{ij})$. There is natural broadening, caused by the quantum mechanical uncertainty in the energy of the transition, $h\nu_{ij}$, which is related to its lifetime, and results in a Lorentzian profile. In addition, there is pressure broadening, caused by the effects of other particles perturbing the transition process, for instance, by interrupting it by a collision, or by collectively affecting the configuration of energy levels. Depending on the exact mechanism, this can cause a variety of different profile shapes. Furthermore, there is Doppler broadening due to the Doppler shifts caused by the motions in the gas, which results in a Gaussian line profile. In principle, the combined effect of all these broadening mechanisms should be considered, using the convolution of the profile functions. For instance, combining a Lorentzian and a Gaussian profile would result in what is called a Voigt profile. However, for the applications that we consider in this thesis, i.e. the outflows of evolved stars, Doppler broadening will be the dominant effect (see e.g. [85]), and thus we can limit ourselves to Gaussian line profiles[1].

[1]However, all our methods can also readily be applied to all other types of line profile functions.

In particular, throughout this thesis, we assume Gaussian line profile functions,

$$\phi_{\nu}^{ij}(\boldsymbol{x}) \ = \ \frac{1}{\delta\nu_{ij}(\boldsymbol{x})\sqrt{\pi}} \ \exp\left[-\left(\frac{\nu-\nu_{ij}}{\delta\nu_{ij}(\boldsymbol{x})}\right)^{2}\right], \tag{2.3}$$

resulting from the Doppler shifts caused by the thermal and turbulent motions of the atoms and molecules in the medium. Note that we normalised the integral over frequency. The characteristic line width,

$$\delta\nu_{ij}(\boldsymbol{x}) \ \equiv \ \frac{\nu_{ij}}{c} \ \sqrt{v_{\text{therm}}(\boldsymbol{x})^{2} \ + \ v_{\text{turb}}(\boldsymbol{x})^{2}}, \tag{2.4}$$

is determined by the local mean thermal and turbulent velocities of the medium. The thermal component will be determined below, while the turbulent component will be estimated for each model independently, based on the hydrodynamics.

2.2.3 Non local thermodynamic equilibrium

Many early line radiative transfer models assumed the state of the medium and the radiation field to be in *local thermodynamic equilibrium* (LTE), i.e. particle velocities, level populations, and radiation field are all completely determined by one value: the local gas temperature, $T(\boldsymbol{x})$. This greatly simplified calculations, but turned out to be inaccurate, even in some very common situations (see e.g. [86]). The weakened alternative for LTE is to instead only assume *kinetic equilibrium*, i.e. to only assume a Maxwell-Boltzmann distribution for the particle velocities. Models that only make the assumption of kinetic equilibrium are often referred to as non-LTE, to emphasise the contrast with the earlier more common LTE models. As a result, the mean local velocity of the gas particles, that appears in equation (2.4) for instance, can be characterised by,

$$v_{\text{therm}}(\boldsymbol{x}) \ = \ \sqrt{\frac{2k_{\text{B}}T(\boldsymbol{x})}{m_{\text{spec}}}}, \tag{2.5}$$

where, m_{spec}, is the atomic or molecular mass of the gas species under consideration, and $k_{\text{B}} \approx 1.381 \times 10^{-23}$ J/K is the Boltzmann constant. Assuming non-LTE, the level populations are not only determined by the local gas temperature, but instead, their dynamics has to be computed consistently with the radiation field. This dynamics is governed by a system of kinetic rate equations.

2.2.4 Kinetic rate equations

The *kinetic rate equations* describe the evolution of the populations, $n_i(\boldsymbol{x})$, of the quantized energy levels, i. They are given by a master equation, which, in the co-moving frame, can be written as,

$$\frac{\partial n_i(\boldsymbol{x})}{\partial t} = \sum_{j=1}^{N} n_j(\boldsymbol{x}) \, P_{ji}(\boldsymbol{x}) - n_i(\boldsymbol{x}) \sum_{j=1}^{N} P_{ij}(\boldsymbol{x}). \tag{2.6}$$

The components of the matrix, $P_{ij}(\boldsymbol{x})$, denote the transition rates from level, i, to level, j. Hence, for each level, i, we have that, $P_{ii}(\boldsymbol{x}) = 0$. As such, the positive term in equation (2.6) sums all transitions into level, i, while the negative term sums all transitions away from that level. The transition rates are composed of a radiative part, $R_{ij}(\boldsymbol{x})$, and a collisional part, $C_{ij}(\boldsymbol{x})$, such that,

$$P_{ij}(\boldsymbol{x}) = R_{ij}(\boldsymbol{x}) + C_{ij}(\boldsymbol{x}). \tag{2.7}$$

The *radiative part* can be expanded further in terms of the Einstein coefficients and the average radiation intensity in the line,

$$R_{ij}(\boldsymbol{x}) = \begin{cases} A_{ij} + B_{ij} J_{ij}(\boldsymbol{x}), & \text{for } i > j \\ B_{ji} J_{ij}(\boldsymbol{x}), & \text{for } i < j \end{cases} \tag{2.8}$$

where, $J_{ij}(\boldsymbol{x})$, is the local mean intensity in the spectral range of the transition, ij. It is computed by averaging the specific monochromatic intensity, $I_\nu(\boldsymbol{x},\hat{\boldsymbol{n}})$, over all directions, $\hat{\boldsymbol{n}}$, and integrating it over the line profile, $\phi_\nu^{ij}(\boldsymbol{x})$, such that,

$$J_{ij}(\boldsymbol{x}) \equiv \oint \frac{\mathrm{d}\Omega}{4\pi} \int_0^\infty \mathrm{d}\nu \; \phi_\nu^{ij}(\boldsymbol{x}) \; I_\nu(\boldsymbol{x},\hat{\boldsymbol{n}}). \tag{2.9}$$

The *collisional part* of the transition rates is composed of the collisional rates, K_{ij}^{p}, for each collision partner, p, weighted by their respective abundances, $n^{p}(\boldsymbol{x})$,

$$C_{ij}(\boldsymbol{x}) \ = \ \sum_{p \in C} K_{ij}^{p}(\boldsymbol{x}) \ n^{p}(\boldsymbol{x}), \tag{2.10}$$

in which C is the set of collision partners. The position dependence in the collisional rates, $K_{ij}^{p}(\boldsymbol{x})$, stems from their dependence on the local gas temperature of the species under consideration. Their values can be determined experimentally and we will assume them as a given (see also Section 2.2.5).

Finally, we assume that all radiative time scales are much smaller than any other relevant time scale in the system. This is known as the assumption of *statistical equilibrium*. As a result, we can approximate equation (2.6) in the static limit, i.e. assuming for all levels, i, that, $\partial_t n_i(\boldsymbol{x}) = 0$. This renders the system of differential equations (2.6) into a linear system of equations in terms of the level populations.

2.2.5 Atomic & molecular line data

The atomic and molecular line data, such as: energy levels, Einstein coefficients, and collisional rates, can, in theory, all be calculated from first principles. In practice, however, this turns out to be very challenging. Therefore, in many cases, we have to resort to experimental values to complement the theoretical (or "ab initio") data. This data is conveniently stored in online databases, such as the Leiden Atomic and Molecular Database[2] (LAMDA, [87]). However, one crucial remark is that both the theoretical and experimental values have (sometimes very large) uncertainties associated with them. In Chapter 6, we present a way to take these into account, but, for now, we assume them to be exact.

There are several relations between the atomic and molecular line data. Hence, usually only a sufficient subset of the data is given from which the other values can be derived. For instance, the Einstein, A_{ij}, and, B_{ij}, coefficients are related as,

$$A_{ij} \ = \ \frac{2h\nu_{ij}^{3}}{c^{2}} \ B_{ij}, \tag{2.11}$$

[2]The database can be found at `home.strw.leidenuniv.nl/~moldata`.

whereas the Einstein, B_{ji}, and, B_{ij}, can be related as,

$$g_j B_{ji} \;=\; g_i B_{ij}, \tag{2.12}$$

in which, g_i, denotes the statistical weight of level, i. Both relations can be derived from the condition of detailed balance in thermodynamic equilibrium (see e.g. [27]). Similarly, the collisional excitation rates can be related to the de-excitation rates as,

$$K^p_{ji}(\boldsymbol{x}) \;=\; K^p_{ij}(\boldsymbol{x})\,\frac{g_i}{g_j}\,\exp\left(\frac{h\nu_{ij}}{k_{\text{B}}T(\boldsymbol{x})}\right), \tag{2.13}$$

which also can be derived assuming detailed balance in thermodynamic equilibrium.

Now all the necessary physics, data, and corresponding underlying assumptions are in place, we are finally in the position to solve the line radiative transfer problem.

2.3 Solution methods

The goal is to solve the radiative transfer equation (1.1), with the emissivity and opacity given by (2.1) and (2.2). This is complicated because the level populations that appear in the emissivity and opacity have to be determined from the kinetic rate equations (2.6), in which the transition rates (2.8) depend on the radiation field. Although the radiative transfer equation is linear in the radiation field and the kinetic rate equations are linear in the level populations, they are coupled in a non-linear way. The resulting system thus requires an iterative solution method.

2.3.1 Lambda iteration

Dropping the position dependence on all variables, the kinetic rate equations (2.6) for each level, i, assuming statistical equilibrium ($\partial_t n_i(\boldsymbol{x}) = 0$), can be written as,

$$\begin{aligned} &\sum_{j,\,j<i} \left\{ n_i A_{ij} \;-\; \left(n_j B_{ji} - n_i B_{ij}\right) J_{ij} \right\} \\ -&\sum_{j,\,j>i} \left\{ n_j A_{ji} \;-\; \left(n_i B_{ij} - n_j B_{ji}\right) J_{ij} \right\} \\ +&\sum_{j=1}^{N} \left\{ n_i C_{ij} \;-\; n_j C_{ji} \right\} \;=\; 0. \end{aligned} \tag{2.14}$$

It is important to remember that the radiation field, and thus, J_{ij}, depends on the level populations through the line contributions to the emissivity and opacity (see equations 2.1 and 2.2). This dependence can be expressed with a Lambda operator (see also Section 1.3.5). We define this operator such that it yields the mean intensity in the line when acting on the set of all level populations, $\mathbf{N} \equiv \{n_i(\boldsymbol{x}), \text{ for all } \boldsymbol{x} \text{ and } i\}$,

$$J_{ij} \;=\; \Lambda_{ij}\left[\mathbf{N}(J_{ij})\right], \tag{2.15}$$

where we explicitly denoted the dependence of, $\mathbf{N}$, on, J_{ij}. In most practical cases, it is unfeasible to explicitly invert the Lambda operator, i.e. directly solve the radiative transfer equation (1.1) and substitute the result in terms of the level populations into the kinetic rate equations (2.14). Instead, we solve equation (2.15, and thus 2.14) in an iterative way, by evaluating J_{ij} using the values from the previous iteration, i.e.,

$$J_{ij}^{(n+1)} \;=\; \Lambda_{ij}\left[\mathbf{N}\big(J_{ij}^{(n)}\big)\right], \tag{2.16}$$

where, n, indicates the iteration number. Unfortunately, this method, often referred to as *Lambda iteration*, converges notoriously slowly towards the true solution (see e.g. [88]), and additional action is required to make it practically feasible.

2.3.2 Accelerated Lambda iteration

Over the years, various methods have been devised to accelerate the convergence of Lambda iterations (for an overview, see e.g. [26] and the references therein). We use the operator splitting method, originally introduced in the context of radiative transfer by Cannon [89, 90], in a similar way to Rybicki & Hummer [91]. The idea is to split the Lambda operator into an approximated part, Λ^{*}_{ij}, that can easily be evaluated and inverted for the current level populations, and a residual part, $\Lambda_{ij} - \Lambda^{*}_{ij}$, that can easily be evaluated for the populations of the previous iteration. Hence,

$$J_{ij} \;=\; \Lambda^{*}_{ij}[\mathbf{N}] \;+\; \left(\Lambda_{ij} \;-\; \Lambda^{*}_{ij}\right)[\mathbf{N}^{\dagger}], \tag{2.17}$$

where the dagger (†) indicates that the quantity is evaluated using the previous iteration. In this way, the contribution of the level populations of the previous iteration is minimised. The kinetic rate equations (2.14) can thus be rewritten as,

$$\begin{aligned} &\sum_{j,\, j<i} \left\{ n_i A_{ij} - \left(n_j B_{ji} - n_i B_{ij}\right) \left(\Lambda^*_{ij}[\mathbf{N}] + J^{\text{eff}}_{ij}\right) \right\} \\ &- \sum_{j,\, j>i} \left\{ n_j A_{ji} - \left(n_i B_{ij} - n_j B_{ji}\right) \left(\Lambda^*_{ij}[\mathbf{N}] + J^{\text{eff}}_{ij}\right) \right\} \\ &+ \sum_{j=1}^{N} \left\{ n_i C_{ij} - n_j C_{ji} \right\} = 0, \end{aligned} \tag{2.18}$$

where we introduced the effective mean intensity in the line, defined as,

$$J^{\text{eff}}_{ij} \equiv \left(\Lambda_{ij} - \Lambda^*_{ij}\right) [\mathbf{N}^\dagger]. \tag{2.19}$$

Note that the effective mean intensity is now the only quantity in (2.18) that still depends on the level populations of the previous iteration.

Clearly, the choice of approximated Lambda operator (ALO; i.e. Λ^*_{ij}) is essential for the success of this acceleration scheme. In some cases, the diagonal part of the true Lambda operator already suffices [92], however, in 3D models, a non-local ALO is often preferred. We follow the construction proposed by Rybicki & Hummer [91].

From the first equation in (1.15), we can identify the Lambda operator, as defined in (2.15), for the Schuster-Feautrier form of the radiative transfer equation,

$$\Lambda_{ij}[\mathbf{N}] = \mathcal{L}_{ij}\left[S + \Psi^+ - \partial_\tau \Psi^-\right], \tag{2.20}$$

where we used the auxiliary linear operator, $\mathcal{L}_{ij}$, defined as,

$$\mathcal{L}_{ij}[\,\cdot\,] \equiv \frac{1}{2} \oint \frac{\mathrm{d}\Omega}{4\pi} \int_0^\infty \mathrm{d}\nu \; \phi^{ij}_\nu \left(1 - \partial^2_{\tau_\nu}\right)^{-1} [\,\cdot\,]. \tag{2.21}$$

Following [91], we can construct an approximation to the Lambda operator by considering only the diagonal band of a certain width, w, of the matrix representation of the auxiliary operator, $\mathcal{L}_{ij}$. We call this operator, $\mathcal{L}^*_{ij}$. The operator, $\mathcal{L}^*_{ij}$, is easy enough to invert, due to its band diagonal structure. However, using it as the ALO would render (2.18) into a system of non-linear equations for the level populations, which would still be hard to solve. In order to retain linearity in equation (2.18), we instead define our ALO as,

$$\Lambda^*_{ij}[\mathbf{N}] \equiv \frac{n_j^\dagger B_{ji} - n_i^\dagger B_{ij}}{n_j B_{ji} - n_i B_{ij}} \, \mathcal{L}^*_{ij}\left[\frac{\eta^{ij}(\mathbf{N})}{\chi(\mathbf{N}^\dagger)}\right], \tag{2.22}$$

in which we only evaluate the line emissivity in the argument of $\mathcal{L}^*_{ij}$ with the new level populations, and add an extra factor, which goes to unity as the level populations converge. Since the line emissivity is linear in the level populations (see equation 2.1), the statistical equilibrium equation will remain linear in the new level populations, and can be written as,

$$\begin{aligned} &\sum_{j,\, j<i} \left\{ n_i A_{ij} - \tilde{\Lambda}^*_{ij}[n_i] - \left(n_j B_{ji} - n_i B_{ij}\right) J^{\text{eff}}_{ij} \right\} \\ &- \sum_{j,\, j>i} \left\{ n_j A_{ji} - \tilde{\Lambda}^*_{ji}[n_j] - \left(n_i B_{ij} - n_j B_{ji}\right) J^{\text{eff}}_{ij} \right\} \\ &+ \sum_{j=1}^{N} \left\{ n_i C_{ij} - n_j C_{ji} \right\} = 0, \end{aligned} \tag{2.23}$$

in which the effective mean intensity, defined in (2.17), is now explicitly given by,

$$J^{\text{eff}}_{ij} = J_{ij}(\mathbf{N}^\dagger) - \mathcal{L}^*_{ij}\left[\frac{\eta^{ij}(\mathbf{N}^\dagger)}{\chi(\mathbf{N}^\dagger)}\right], \tag{2.24}$$

and we introduced another auxiliary approximated operator,

$$\tilde{\Lambda}^*_{ij}[n_i] \equiv \frac{h}{4\pi}\left(n_j^\dagger B_{ji} - n_i^\dagger B_{ij}\right) \mathcal{L}^*_{ij}\left[\frac{A_{ij}\, \nu\, \phi^{ij}_\nu\, n_i}{\chi(\mathbf{N}^\dagger)}\right]. \tag{2.25}$$

Note that this operator is linear in the new level populations and not symmetric in the level indices i and j. However, since our ALO (2.22) is symmetric in these indices, it can be implemented in the transition matrix. Furthermore, although we started in equation (2.14) with an ($N_{\text{lev}} \times N_{\text{lev}}$)-matrix equation, with N_{lev} the number of energy levels in the species, we now, in case of a non-local ALO in equation (2.23), have

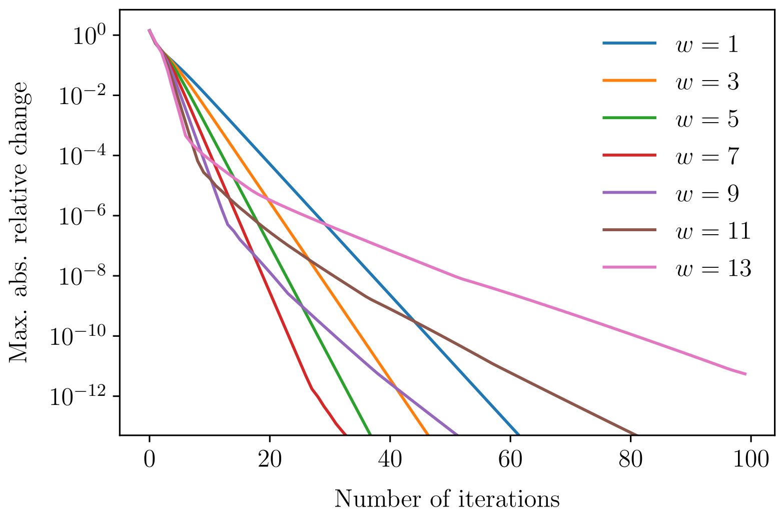

Figure 2.1: Convergence behaviour, represented by the maximum absolute relative change between consecutive iterations, for several band-diagonal ALOs with different widths, w, computed with Magritte for problem 1a of the van Zadelhoff benchmark [93] (see also Section 3.4).

an ($N_\mathrm{p}N_\mathrm{lev} \times N_\mathrm{p}N_\mathrm{lev}$)-matrix equation, with N_p the number of points, since the ALO couples between different spatial points. In case of a local ALO, the resulting system still decouples for each spatial point. Fortunately, since for reasonable bandwidths the ALO is very sparse, also the resulting system of equations (2.23) will be sparse.

In summary, using operator splitting, the convergence of Lambda iterations is improved, since the better the approximation, Λ^{*}_{ij}, for the Lambda operator, Λ_{ij}, the smaller the values for J^{eff}_{ij} will be, and hence, the smaller the dependence in (2.23) on the previous (less accurate) approximation to the level populations, which makes solving (2.23) more like solving the full problem than like solving the iterative one.

Figure 2.1 shows the convergence behaviour over 100 iterations for several band-diagonal approximated Lambda operators with different widths, w, such that, for instance, $w = 1$, implies a diagonal matrix, $w = 3$, is a tridiagonal matrix, and so on. The vertical axis shows the maximum of the absolute change between two consecutive iterations divided by their average. Initially, we see indeed that the wider the band-diagonal of the ALO, the better the convergence. However, for widths, $w \geq 7$, there is an elbow point at which convergence significantly slows down, and it slows down more and earlier in the iteration process the wider the band-diagonal. By investigating the level populations throughout the iteration process, we can identify that for all bandwidths the steepest convergence occurs when for the entire model the iterative solution is either an overestimate or underestimate for the true solution, i.e. when the change in level populations between consecutive iterations has the same sign everywhere. This is what happens in the first few iterations. Once the iterative solution is close enough to the true solution that for some points it is an overestimate and for others an underestimate, i.e. when the change in level populations between consecutive iterations has a different sign for different points, then convergence significantly slows. This is what happens at the elbow. Currently, we do not have a satisfying explanation as to why this happens. We also did not find any reference to this phenomenon in literature. It should be noted, however, that in practice the convergence criterion is often already defined at a maximum absolute relative change of 10^{-4}, and thus outside the regime where this phenomenon occurs, see e.g. [94][3]. Therefore, practically, it is probably only of little significance. Furthermore, since (at least empirically) we can signal the issue by monitoring the sign of the changes between consecutive iterations, it can easily be cured by lowering the bandwidth, w.

2.3.3 Acceleration of convergence

Accelerated Lambda iterations are often combined with *acceleration of convergence* methods to further reduce the required number of iterations. One popular technique is Ng-acceleration [95], named after Kin-Chue Ng, who proposed it in 1974. Later, it was introduced in the context of Lambda iterations by Buchler & Auer [92, 96]. However, in mathematics and computer science literature, the same method goes under many different names, such as e.g. Anderson-acceleration, after [97], or more generally: non-linear acceleration [98]. The main reason for its popularity is its general ease of implementation and the fact that it can be applied to any iterative scheme. We find, however, that in radiative transfer literature [26, 92, 95], this method is often presented in quite a restrictive way. Therefore, we present here the non-linear acceleration scheme, more in line with mathematics literature [98].

[3]More precisely, [94] quote a relative accuracy of 10^{-4} in the mean intensity, which indeed also roughly corresponds to a maximum absolute relative change of 10^{-4} in the level populations.

We start by approximating the $\Lambda_{ij}[\mathbf{N}(\cdot)]$-operator from equation (2.16) with a linear operator, L, such that the resulting (linearised) iteration process is given by,

$$J^{(n+1)} = \mathsf{L}\big[J^{(n)}\big]. \tag{2.26}$$

Finding the limit of this iteration process corresponds to finding the fixed point, J_{FP}, of the L-operator. After $N-1$ iterations, we can attempt to make a fixed-point estimate, J_{FPE}, with a linear combination (or average) of the, M, previous iterates,

$$J_{\text{FPE}} \equiv \sum_{n=1}^{M} c_n \, J^{(N-n)}, \quad \text{with the condition that,} \quad \sum_{n=1}^{M} c_n = 1. \tag{2.27}$$

The necessity for the additional condition on the sum of the coefficients will become clear below. For now, it can be viewed as a requirement to interpret the fixed-point estimate as an average of the previous iterates. The desired coefficients, c_n, that would provide a good estimate for the fixed point, can be obtained, for instance, by minimising the error in the (linearised) fixed-point condition, i.e. $J_{\text{FPE}} = \mathsf{L}\big[J_{\text{FPE}}\big]$, with respect to some norm. In that case, $\boldsymbol{c} \equiv [c_i]$, is given by,

$$\boldsymbol{c} \equiv \underset{\boldsymbol{c}:\, \boldsymbol{c}^{\text{T}}\mathbf{1}=1}{\arg\min} \left\{ \big\| J_{\text{FPE}} - \mathsf{L}\big[J_{\text{FPE}}\big] \big\|^2 + \lambda^2 \sum_{n=1}^{M} c_n^2 \right\}, \tag{2.28}$$

where we added a regularisation term, governed by the regularisation constant, λ, that penalises large numerical values of the coefficients, c_n. Using equation (2.26), and leveraging the assumed linearity of the L-operator, we can rewrite (2.28) as,

$$\boldsymbol{c} = \underset{\boldsymbol{c}:\, \boldsymbol{c}^{\text{T}}\mathbf{1}=1}{\arg\min} \left\{ \left\| \sum_{n=1}^{M} c_n R_n \right\|^2 + \lambda^2 \sum_{n=1}^{M} c_n^2 \right\}, \tag{2.29}$$

where we defined the residuals, R_i, the difference between consecutive iterates as,

$$R_n \equiv J^{(M-n)} - J^{(M-n+1)}. \tag{2.30}$$

From equation (2.29) it is clear that we had to add the constraint, $\boldsymbol{c}^{\text{T}}\mathbf{1} = 1$, since otherwise only the trivial solution, $\boldsymbol{c} = \mathbf{0}$, could be obtained. Note that, in practice, each $J^{(n)}$ corresponds to a vector in parameter space, for instance, given by the different points in the model geometry, and thus $\|.\|$ is a norm on this parameter space. If this space is D-dimensional, then each R_n is a D-dimensional vector, and, $R \equiv [R_n]$, can be thought of as an $(M \times D)$-dimensional matrix. The minimisation problem in (2.29), given the constraint on the normalisation of $\boldsymbol{c}$, can be solved using the Karush-Kuhn-Tucker (KKT) theorem [99, 100], and yields,

$$\boldsymbol{c} \ = \ \boldsymbol{z} \ / \ (\mathbf{1}^{\mathrm{T}}\boldsymbol{z}), \quad \text{in which } \boldsymbol{z} \text{ is defined by, } \ \left(R^{\mathrm{T}}S^{\mathrm{T}}SR \ + \ \lambda^2\mathbb{1}\right) \ \boldsymbol{z} \ = \ \mathbf{1}, \tag{2.31}$$

where, $\mathbf{1}$, is an M-dimensional vector of ones and S is the $(D \times D)$-dimensional matrix that relates the norm in parameter space $\|.\|$ to the Euclidean norm $\|.\|_E$ via the relation $\|.\| = \|S\,.\|_E$. To get a better idea of this system that has to be solved (2.31), one can rewrite it as,

$$\sum_{b=1}^{M} \left(R_a \cdot R_b \ + \ \lambda^2\delta_{ab}\right) \ z_b \ = \ 1, \quad \text{for each } a \in \{1, \ldots, M\}, \tag{2.32}$$

where the dot indicates the inner product corresponding to the norm in the parameter space. For a typical radiative transfer application the dimension of the parameter space will be much larger than the number of iterations, $D \gg M$. Therefore, the computational cost of solving for the coefficients, $\boldsymbol{c}$, will be dominated by the evaluation of the inner products over the parameter space rather than solving the system (2.32). As a result, this acceleration scheme is computationally relatively cheap and usually only limited by the memory required to store the previous iterates.

The effects of different regularisation parameters on the fixed-point estimates are shown in Figure 2.2. A larger regularisation parameter clearly results in a more stable convergence behaviour, however, ultimately, it also leads to slower convergence. This can be understood, since, although regularisation improves the conditioning of the resulting linear system (2.31), it also drives the solution away from its optimal value, as it perturbs the optimisation problem. As a result, we see in Figure 2.2 that, the higher the regularisation parameter is, the lower convergence

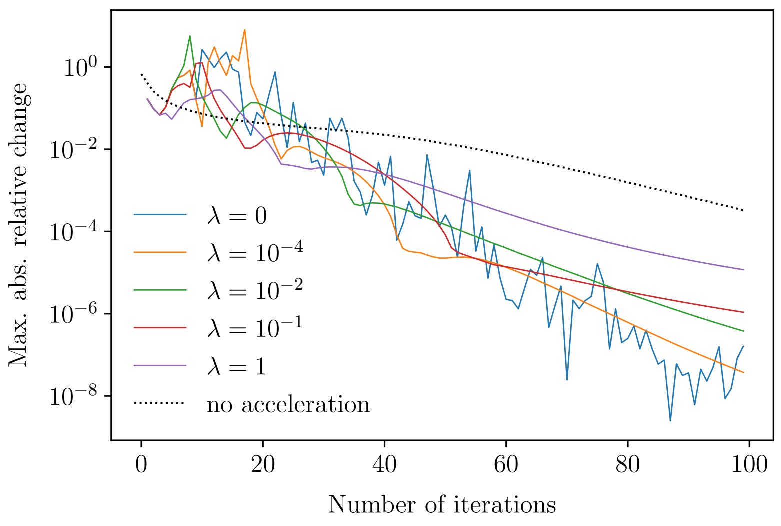

Figure 2.2: Convergence behaviour, represented by the maximum absolute relative change between consecutive iterations and the corresponding fixed-point estimates with different regularisation parameters, λ, computed with MAGRITTE for problem 1b of the van Zadelhoff benchmark [93] (see also Section 3.4).

is after 100 iterations. Therefore, a dynamic regularisation parameter should be preferred, for instance, defined in terms of the minimum absolute value in the matrix $R^{\text{T}}S^{\text{T}}SR$. However, although regularisation might help improve convergence in some applications and allows us to prove rigorous bounds (see e.g. [98]), we did not find significantly improved convergence in any of our radiative transfer applications.

Classical Ng-acceleration corresponds to the case where M is fixed and $\lambda = 0$. Figure 2.3 shows the convergence behaviour over 100 iterations with classical Ng-acceleration for several orders, M, which were recorded for problem 1b in the van Zadelhoff benchmark [93] (see also Section 3.4). Often Ng-acceleration is only applied after a certain level of convergence is already reached. This can be justified, since we assumed that the $\Lambda_{ij}[\mathbf{N}(\cdot)]$-operator can be approximated by a linear operator, which is only guaranteed close to the fixed-point. Hence, in the first few iterations, when iterates can still be far away from the fixed-point, the proposed fixed-point estimate is not guaranteed to be any good. After this, acceleration steps are usually applied periodically after a fixed number, M, of regular Lambda iterations. The dotted lines in Figure 2.3 show the effect of waiting for 20 iterations before applying Ng-acceleration. Nevertheless, this results in very

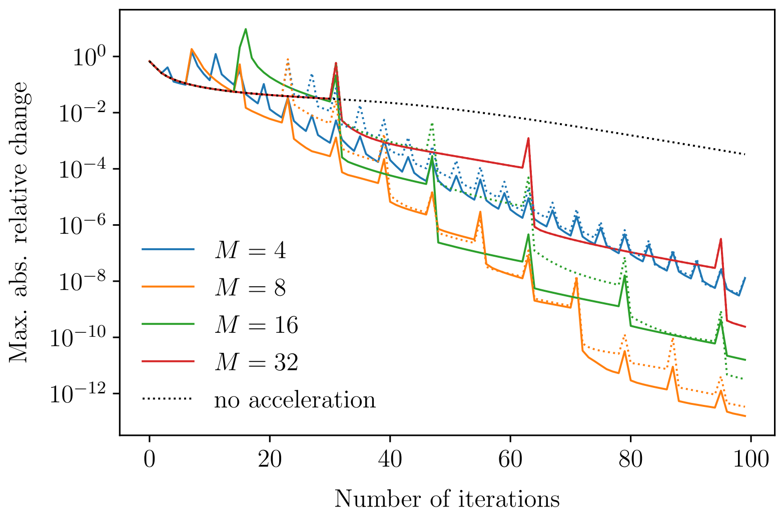

Figure 2.3: Convergence behaviour, represented by the maximum absolute relative change between consecutive iterations with classical Ng-acceleration for several orders, M, computed with MAGRITTE for problem 1b of the van Zadelhoff benchmark [93] (see also Section 3.4). The dotted lines show the minor effect of waiting for 20 iterations before turning on Ng-acceleration.

similar convergence behaviour. For this particular model, the best convergence is observed for $M = 8$, however, this is not in general optimal. The more iterates, the better the approximation, but the smaller the gain in accuracy over computational cost. Furthermore, the order, M, is mainly limited by the memory available to store the previous iterates. Since it is hard to express the desired effects in terms of what to do after how many iterations, we propose a dynamic acceleration approach.

Dynamic non-linear acceleration We propose to run the acceleration steps in parallel with the regular iteration scheme, i.e. perform regular iterations and compute every time the corresponding fixed-point estimate based on all available regular iterations, while checking the convergence of both the regular iterations and the resulting series of fixed-point estimates. The number of available iterations, m, is predefined by the memory capacity of the system and the model under consideration. When either the fixed-point estimates show better convergence than the regular Lambda iteration, or the maximum number of stored regular iterations is reached, the last estimate is injected into the regular iteration scheme. Then, the previous iterates are deleted, and starting from this fixed point estimate, the regular iterations are continued, again with the acceleration steps in parallel, storing the previous

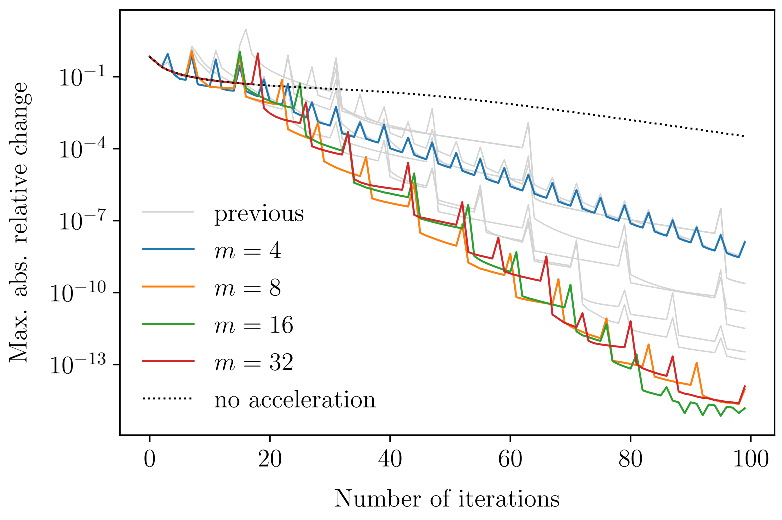

Figure 2.4: Convergence behaviour, represented by the maximum absolute relative change between consecutive iterations, with our new dynamic non-linear acceleration, for several storage maxima, m, computed with MAGRITTE for problem 1a of the van Zadelhoff benchmark [93] (see also Section 3.4). The grey lines show the previous results with Ng-acceleration from Figure 2.3.

regular iterates, and checking convergence. This allows us to use the fixed-point estimates whenever they show better convergence than the current regular iteration, hence allowing us to directly capitalise on any potential benefits.

Figure 2.4, shows the convergence behaviour of the new dynamic acceleration technique for several storage maxima, m. Most schemes show significantly better performance than their classical counterparts (shown in grey). Only for $m = 4$, the advantage is very minor. This can be understood, since, when only 4 previous iterates are available, it turns out that for this particular problem the convergence rate of the fixed-point estimates almost never drops below the convergence rate of the regular estimate, such that acceleration is just applied after every 4 regular iterations, effectively rendering this into classical Ng-acceleration of order $M = 4$. The other schemes ($m = 8, 16, 32$) all show very similar good convergence. Hence, we eliminated the need to tweak any parameters (e.g. number of iterations before acceleration, order of the acceleration scheme). The dynamic scheme automatically balances the regular and accelerated iterations in a way that outperforms any classical scheme with the same storage capacity. The only remaining parameter is, m, which should be maximised, given the memory constraints.

Finally, it should be emphasised that a small value for the maximum absolute relative difference between consecutive iterations, which we used as a proxy for convergence, does not guarantee that the iteration process has converged towards the true solution of the system. Therefore, it is always advised to check whether the obtained solution is indeed the true solution. Although there is no single method that can guarantee this, one can always gain more confidence in a solution, for instance, by starting the iteration process at different initial conditions and checking whether they always converge to the same solution. This is the method that we used to check the convergence for the examples in this thesis.

Throughout this chapter, we presented the mathematical ingredients that are required to model line radiative transfer. In the next chapter, we will demonstrate how these are implemented in our software library: MAGRITTE. In other words, we will present the lines of code, to decode what spectral lines encode.

Chapter 3

MAGRITTE: design & performance

The best code is the code that is never(theless) written.

– after Edsger W. Dijkstra

3.1 Introduction

The computational difficulties encountered in line radiative transfer, as discussed in Chapter 2, require a highly efficient solution method. In this chapter, we present the design and evaluate the performance of MAGRITTE, a modern software library for radiative transfer. This library was developed to test the ideas presented in this thesis, but also as a practical tool for astrophysics, in particular, for stellar wind research (see e.g. Chapter 4). The goal was to work towards a solver that can provide fast and reliable on-the-fly radiative transfer in hydro-chemical simulations. Since the main obstacle is the computational cost, there is a strong emphasis on performance.

But why did we build yet another radiative transfer solver? Admittedly, five years ago, there were already several 3D radiative transfer solvers available, see for instance [35–38, 41–43, 45, 48]. First of all, we wanted to go back to the classical formal methods and avoid Monte Carlo methods to maximise our control over the solver. The idea was that modern computer hardware had evolved enough to be able to benefit from the robustness of classical formal methods. We thus wanted a library that could leverage vector instructions, multi-core and multi-node parallelism, as well as accelerators. To achieve this in any existing code base usually requires a complete rewrite of all internal data structures, which encouraged us to develop our own with parallelisation and acceleration already in mind.

In retrospect, however, it probably would have been better to start from an existing radiative transfer library that already had a good front-end (i.e. data input, model setup, output), such as e.g. the recent LIGHTWEAVER [101], such that we could concentrate on the back-end. Furthermore, it probably also would have been better to rely more on existing back-end libraries, for instance for linear algebra, so we could focus our efforts on the problems specific to radiative transfer, adhering more to the advice of E. W. Dijkstra at the beginning of this chapter (see also Chapter 7).

Nevertheless, we developed a new software library for 3D radiative transfer, named MAGRITTE, for which the source code of the latest version can be found on GITHUB at `github.com/Magritte-code/Magritte`.

3.2 Design strategy

In the following, we describe the overall a posteriori design strategy for MAGRITTE. It should be noted that in the past five years the entire code base has been rewritten completely at least seven times. The design principles presented below should thus be read as the lessons learned throughout this design process, rather than as a set of predetermined principles used in its design.

3.2.1 Parallelisation & acceleration abstractions

One of the key motivations to build a new radiative transfer library was to be able to tailor its data structures such that they could leverage multi-core and multi-node parallelism, as well as different types of accelerators from several different vendors. The challenge, however, is to do this in a maintainable way, and avoid having to write an entire separate implementation for each programming model. The goal is thus to have a single source code that is portable between the different programming models. However, portability alone is not enough, since the resulting program should, of course, also still perform well within each programming model.

Following the example of the data-parallel mathematical object library GRID[1] by Boyle et al. [68], we identify two key steps to achieve (performance) portability: (1) abstract away all programming model-specific and vendor-specific interfaces

[1]See also `github.com/paboyle/Grid`.

behind a single internal interface, and (2) abstract away the memory layout of all data structures such that they can be tuned for each specific implementation. In this way, portability is achieved since the application can be written entirely in terms of the abstractions, and only the abstractions have to be implemented separately for each programming model.

In Magritte, we collected all parallelisation and acceleration abstractions in a separate header-only C++ library, that we named: ParAcAbs[2]. As such, they can readily be reused in other projects. ParAcAbs mainly consists of a set of custom data types and (C++) preprocessor macros for parallel `for`-loops.

ParAcAbs data types The data types include a `Vector`, `Matrix`, and `Tensor` structure, and their thread-private counterparts named: `VectorTP`, `MatrixTP`, and `TensorTP`. Furthermore, there is a special `Vector3D` structure with some additional functionality for geometric computations. All ParAcAbs data types can also be set and accessed in Python via PyBind11[3] [102]. The main idea behind the ParAcAbs data types is best illustrated with a (simplified) version of the `Vector` structure.

```
template <typename type>
struct Vector
{
    std::vector<type> vec;
    type* dat;
    type* ptr;

    ...

    void set_dat()
    {
        if (contextAccel()) dat = ptr;
        else                dat = vec.data();
    }

    type operator[](size_t id) {return dat[id];}
};
```

This is just a wrapper around the C++ standard library container `std::vector`, containing two extra pointers `dat` and `ptr`. From the definition of the `operator[]`, we see that the pointer, `dat`, points to the data that will be accessed when indexing the `Vector`. If there is another device, such as e.g. a GPU, the pointer, `ptr`, points

[2]See also `github.com/Magritte-code/Paracabs`.
[3]See also `github.com/pybind/pybind11`.

to the same data on that other device. The set_dat function sets the pointer, dat, either equal to the pointer of the std::vector or to the pointer ptr, depending on the context. The context is determined with the function contextAccel(), which returns true if the program is in a context which is to be executed on the accelerator, and false otherwise. Note that, to avoid overhead, the context is not checked at every data access, i.e. not in the operator[]. Instead, every time the context might switch, the set_dat function must be called to make sure the right pointers are used internally. The other functions, not included in the listing above, are mainly memory management functions, for instance, to allocate, de-allocate, and copy memory both for the host and the device.

The Matrix and Tensor structures both derive from the Vector structure and differ only in the number of indices. The corresponding thread-private structures have an additional internal index, representing the thread identifier, and provide the same functionality, with the only difference that, within the context of a parallel loop, each points to its own thread-private memory location. For example, a thread-private Vector, defined as VectorTP, is the same as a Matrix, but with one index indicating the thread identifier.

ParAcAbs macros Parallel for-loops are implemented in ParAcAbs as variadic macros in which the loop body is substituted in a capturing lambda. To illustrate how this works, we present below a (simplified) version of the accelerated_for-loop, as implemented in CUDA [69].

```
namespace pa = paracabs::accelerator;

#define accelerated_for(i, total, ... )                            \
{                                                                  \
    contextAccel() = true;                                         \
    auto l = [=, *this] __device__ (size_t i) mutable              \
    {                                                              \
        __VA_ARGS__;                                               \
    };                                                             \
    decltype(l)* l_ptr = (decltype(l)*) pa::malloc(sizeof(l));\
    pa::memcpy_to_accelerator(l_ptr, &l, sizeof(l));               \
    dim3 nblocks  = pa::nblocks();                                 \
    dim3 nthreads = pa::nthreads();                                \
    apply_l <<<nblocks, nthreads>>> (total, l_ptr);                \
    contextAccel() = false;                                        \
}
```

```
template <typename Lambda>
__global__ void apply_l (size_t stop, Lambda* l)
{
    size_t start  = blockIdx.x * blockDim.x + threadIdx.x;
    size_t stride =  gridDim.x * blockDim.x;

    for (size_t i = start; i < stop; i += stride)
    {
        lambda->operator()(i);
    }
}
```

The `accelerated_for`-loop has two named arguments, the first, `i`, representing the index that is looped over, and the second, `total`, represents the range of the index, such that `i` $\in \{0, \ldots,$ `total` $- 1\}$. When entering and leaving the `accelerated_for`-loop, we notice that the context is explicitly switched by setting `contextAccel()`. Furthermore, the remaining arguments of the variadic macro (indicated with `...`) are substituted (indicated as `__VA_ARGS__`) into the capturing lambda, named: `l`. All variables are captured by copy. The version given above is meant to be used within a class, since `*this` indicates to capture also the enclosing object by copy. The resulting lambda expression, `l`, thus yields a function that contains the body of the `accelerated_for`-loop. Since many variables can be contained in these objects, the lambda expression can become large and cannot be transferred to the device in a function argument, but rather has to be copied explicitly. Finally, the (pointer to the) lambda expression is passed to the CUDA-kernel, named `apply_l`, in which the lambda is iterated over in a strided loop. In the absence of an accelerator, the `accelerated_for`-loop is redefined as a simple multi-threaded loop in OpenMP, and in the absence of OpenMP, this is again redefined as a simple serial `for`-loop. In this way, by abstracting the behaviour of an accelerated `for`-loop, and falling back on other options depending on the available hardware or the preferred programming model, we can obtain portable code with minimal duplication. In ParAcAbs, a similar implementation is also available for SYCL.

Finally, we should note that the ParAcAbs abstraction layer presented above should only be regarded as a temporary design solution for performance portability. Hopefully, it can soon be replaced by a standard unified programming model.

3.2.2 Parallelisation & acceleration strategy

In the previous section, we demonstrated how parallelisation and acceleration can be implemented in a portable and yet maintainable way, by abstracting away all specifics behind an interface. The question remains, however, where which type of parallelism should be used in radiative transfer computations.

MAGRITTE is a formal solver (see also Section 1.4) that solves the radiative transfer equation (1.1) on a ray-by-ray basis, i.e. the radiation field is computed for each ray, through each point, and that for each frequency bin. The resulting computation of the radiation field can thus schematically be written as follows:

```
for (direction: directions)
{
    for (point: points)
    {
        Ray_pair ray_pair = trace (direction, point);

        for (frequency: frequencies)
        {
            solve_radiative_transfer (ray_pair, frequency);
        }
    }
}
```

The `trace` function traces an antipodal `ray_pair`, i.e. a straight line, through the model, originating from the given `point`, in the given `direction` and its antipode. Then, the `solve_radiative_transfer` function solves the radiative transfer equation along the given `ray_pair`, and for the given `frequency`. There are several possible strategies to parallelise and accelerate the above computation. We will first outline our entire strategy and then justify our particular choices.

The simplest yet effective parallelisation strategy for this computation assigns one type of parallelism (multi-node, multi-core, and vectorisation respectively) to each of the `for`-loops: The outer loop over the different directions is parallelised using MPI to be distributed over the multiple nodes in the system, the middle loop over the different points is parallelised using OPENMP to be split over the different cores inside each processing unit, and the inner loop over the different frequencies is vectorised.

Multi-node The reason to distribute the outer loop over the nodes in the system is twofold: On the one hand, simply to increase the computational throughput by executing the calculations concurrently, but, on the other hand, also to distribute the memory cost over the different nodes in the system. The radiation field is by far the largest data structure in this simulation. It is stored as one double-precision floating-point number for each direction, point, and frequency bin in the model. Every node has its own memory and can be made to store the radiation field only in the directions it is concerned with. As such, the memory cost per node can roughly be divided by the number of nodes. Furthermore, since the transfer equation is solved on a ray-by-ray basis, no communication is required between the nodes during an iteration, such that the communication overhead can be kept to a minimum. Only at the end of an iteration, reductions are required, for instance, to integrate over the different directions and compute the mean intensity. Finally, since a typical 3D model requires to take into account hundreds of directions, this strategy can (in principle) be scaled up to hundreds of nodes. The multi-node scaling of Magritte is discussed in Section 3.4.3.

Multi-core Given the distribution over the nodes as described above, the next level of parallelism in the software should be split over the next level of parallelism in the hardware, i.e. the different cores within each node. Before the rays are traced, there is no way to know which points in the geometry will be needed where in the computation. Therefore, it is favourable to have them all in the shared memory of each node, and not distributed, as we did with the directions. The multi-core scaling of Magritte is discussed in Section 3.4.3.

Vectorisation Solving the radiative transfer equation along a given ray pair requires the execution of exactly the same instructions for each frequency bin. The only thing that is different are the values for the radiative properties at each frequency. As a result, the computations for different frequencies are ideally suited for vectorisation, i.e. by grouping the operations for multiple frequency bins into vector (or SIMD) instructions. When the loop over the frequencies is written out explicitly, as done in the code sample above, it will be hard for any compiler to recognise this optimisation opportunity. Therefore, it is better to ensure that the frequency index is laid out contiguously in memory, and then abstract away the frequency index of each variable behind a vector type, such that operations between objects of this type, for instance,

```
vectorType<double> a, b, c;

a = b + c;
```

automatically translate to a loop over the frequencies, i.e.,

```
for (int f = 0; f < frequencies.size(); f++)
{
    a[f] = b[f] + c[f];
}
```

The latter will most likely be vectorised by the compiler, given the right compiler flags. To ensure this, one can also explicitly define the operators on `vectorType` variables in terms of the corresponding intrinsic vector instructions, such as done for instance in Grid [68]. In Magritte, although we successfully implemented the explicit vectorisation from Grid in an earlier version, we dropped it later on for compatibility reasons, and rely now on the compiler for automatic vectorisation. In the future, it could conveniently be reintroduced by implementing it, for instance, in the data types provided in ParAcAbs.

GPU acceleration Since GPUs are in essence large vector processors, i.e. able to execute vector instructions but on much larger vectors, the above considerations about vectorising over the frequencies can also be used to implement a GPU version of the solver. The most straightforward way to do this, is to offload the entire `solve_radiation_field` function to the GPU, and use the (now larger) vector instructions again to act on the frequency index[4]. The simplest way to target multiple GPUs per node, is to create multiple processes per node, each targeting a single GPU. It should be noted that, in this case, the performance of the node is most likely bounded by the data transfer between the CPU and GPU, and the number of available GPUs in the node. Performance can thus be enhanced by streamlining the data transfer, and by grouping computations to create more vector parallelism. We

[4]Following the CUDA nomenclature, this means taking the data that is meant for the different SIMD-lanes in the CPU and map it to the different SIMT-lanes in the GPU.

tried to do this in several ways, from grouping the computations for different rays, to offloading the entire for-loop over the different points. The main difficulty in creating additional vector parallelism that is not inherent to the computation is that it requires quite some orchestration of the computations on the GPU. Unfortunately, we did not yet manage do this in a way that significantly improves performance.

3.2.3 Test strategy

Proving the correctness of (scientific) software is in general very hard, since the resulting output usually depends on an extremely large number of parameters. Some of these parameters, such as the input data and the source code, we can control and test, as we describe below. However, we cannot strictly control every parameter, since, for instance, the source code is not written in machine code, but rather in a higher level language (such as PYTHON or C++) that is then translated (or compiled) into machine code. This translation often involves an optimisation step, which can change the order or form of the machine instructions. Although these optimisations are obviously restricted to guarantee the accuracy of the result up to some level, any change in the order or form of the instructions can still cause a numerical difference in the results, which furthermore might be amplified in the rest of the code. A single source code compiled at different levels of optimisation can thus yield different results, which makes it hard to strictly control the accuracy.

In an attempt to validate the correctness of our software, we therefore require a testing strategy. In MAGRITTE, we have three layers of tests to do that: sanity checks, unit tests, and integration or system tests, which we discuss in more detail below.

3.2.3.1 Sanity checks

Sanity checks are small checks throughout the code of properties that we assume to be true. They are performed on every execution of the relevant portion of code and can be implemented as simple `if`-statements, conditionally throwing an error or warning. In MAGRITTE, the model setup is usually provided by the user through a PYTHON script, which is quite error prone, since we make various assumptions, for instance, about the way in which the points, neighbours, and rays are stored. Therefore, when loading the input, we explicitly test these assumptions, to ensure the validity of the input. This type of test is often overlooked in discussions on testing strategies, but rightfully deserves its place here, since it has been by far the most helpful type of test in the development and use of MAGRITTE.

Unfortunately, this type of tests is usually unacceptable in performance critical parts of the code. However, in MAGRITTE, given the complexity of the part of the code that computes the Lambda operator, there is the option to store and perform sanity checks on the Lambda operator. However, this is not available in production versions of the code because of the resulting performance penalty.

3.2.3.2 Unit tests

Unit tests verify the validity of a small portion of code and typically concern just a single function. They are usually implemented as separate test programs that can be executed individually to test specific functionality. There are several frameworks that provide useful tools to facilitate the construction of unit tests, such as CATCH2[5] and GOOGLETEST[6], the latter of which is currently used in MAGRITTE.

A comment on test-driven development Test-driven development (TDD) is a software development process in which the requirements for the software are first formulated as a set of tests that the software should pass, before the actual code is written (see e.g. [103]). This is often already implemented on the lowest level of unit tests, such that the validity of the code is strictly controlled, even before it is written. This approach has many advantages, as tests are created concurrently with the code, resulting in a wide test coverage and hence justifiable confidence in the code. If the tests are designed properly, most bugs can be caught very early on in the development process, reducing the need for complex debugging of large portions of code. Nevertheless, despite the major benefits and its success in certain industries, test-driven development has proven to be more difficult to adopt, especially in scientific software engineering (see e.g. [104]). At least part of this can be attributed to a certain reluctance from the developers' side, since it usually requires quite a large change in workflow and indeed mentality. Moreover, a lot of scientific software is written by scientists, rather than trained software engineers, who are

[5]See also `github.com/catchorg/Catch2`.
[6]See also `github.com/google/googletest`.

often less aware of the available tools and development strategies. In addition, most of the examples and best practices are tailored to large projects involving multiple developers in an industrial setting, which is often quite different from the academic setting (although recently guidance for best practices has appeared, see e.g. [105]).

Although we heavily endorse the test-driven development strategy, this is not the way in which MAGRITTE was developed. As a result, the (unit) test coverage of the code is also rather poor. Despite several attempts to reinstate a test-driven approach at each major rewrite of the code base, it was never consistently maintained. Only in the later developed PARACABS library (see Section 3.2.1), tests were developed simultaneously with the code, but their utility was somehow limited, since any real test would require a variety of different hardware to run on. In retrospect, the reason why this worked for PARACABS and not for MAGRITTE can be attributed to the fact that for PARACABS there was a clear predefined interface that the library had to adhere to, e.g. the vector, matrix, and tensor operations, and the parallel `for`-loops. As a result, tests could be written based on that predefined and fixed expected behaviour. In contrast, at every major rewrite of MAGRITTE the internal structure of the code changed so dramatically that all tests had to be rewritten entirely, such that test-driven development implied a major development overhead. Hence, for instance here, it would have helped to have started development on top of a library that already had a good front-end, and using more existing libraries for the back-end, since this would have fixed the internal behaviour more. The inability to maintain a test-driven development strategy can thus in a sense be traced back to the tension in the dual raison d'être of MAGRITTE, namely: on the one hand being a framework to prototype new radiative transfer methods, while on the other hand being a useful tool for astrophysics. A test-driven development strategy only works well with the latter, whereas in the former the strategy is usually to develop and either succeed or fail as fast as possible.

To avoid these difficulties in the future development of MAGRITTE, apart from defining a clear interface for the front and back-end, we propose to have a much clearer separation between prototype and production code. This can be achieved with proper version control, e.g. with Git, by having a single strictly controlled production branch, which is fully tested, and using different less controlled branches for each prototype. Since this setup still bears the temptation of pushing untested code into production, it might be advised to separate real prototyping and production even more, for instance, by having a completely separate prototyping environment (or separate language), such as, for instance, in a Jupyter notebook [106].

3.2.3.3 Integration & system tests

Integration tests verify the validity of a combination of several smaller pieces of code, usually concerning the interplay of several functions, and thus test the aggregate of what would be covered by unit tests. System tests verify still bigger portions of code, for instance, testing the behaviour of an entire application. We combined integration and system tests here, since the function call trees in Magritte are not that deep, and therefore integration tests can also be considered system tests.

For the integration or system tests in Magritte we considered several semi-analytical and numerical problems. The details of these models and the performance of Magritte compared to other solvers is discussed in Section 3.4. Both models and results of these benchmarks are provided together with the source code of Magritte, such that its validity can be checked before it is used. Note, however, that it should be the responsibility of the developer, and not the user, to check their software before release. Therefore, we also employ continuous integration and deployment tools.

3.2.4 Continuous integration & deployment

Continuous integration (CI) and continuous deployment (CD) are modern practices in software development that aim to keep the integration and deployment cycle as short as possible, for instance, by integrating new code into the main source at least daily (see e.g. [107]). This avoids large conflicts between different versions of the code, simplifying and speeding up the development process. Although these practices are mostly aimed at large development teams, also smaller scale projects, such as Magritte, can benefit from their practices and tools. Below, we list some best practices, that can be found for instance in [107], that are typical for CI and CD, and we explain how we apply them in the development of Magritte.

Version control Proper version control, for instance using GIT, is an indispensable tool for any modern software project. It structures the development process and allows you to see or even revert back to previous versions when things go wrong. For MAGRITTE, we use a GIT repository, hosted online on GITHUB.

Automated builds The build process, i.e. the process of turning the source code into an executable, has a significant impact on the resulting application. For instance, the choice of compiler, compiler flags, linking libraries, etc., can all significantly affect the exact output and performance of the application. Furthermore, tests and documentation are often linked to a specific version of the source code, and thus should be rebuilt together with the source. Therefore, it is advised to have a single command that can build the entire application with the preferred settings together with the tests and documentation. In MAGRITTE, we use CMAKE[7] to orchestrate the build of the C++ library, the PYTHON library, and the tests. Optionally, this can also be used to trigger a build of the inline documentation with DOXYGEN[8].

Test on build Once the application is built, it must be tested. Using CMAKE, the test suite can easily be incorporated to execute at the end of the build process.

Build on commit To ensure the validity of the repository, it is important to build (and hence also test, see the previous point) the application each time a change is committed to the code repository. Locally, on the developer's machine, this can be automated, for instance, using GIT-hooks, which can trigger the execution of a command at each commit. For remote versions, for instance on GITHUB, one can use a service, such as JENKINS[9] or TRAVIS CI[10] that allows you to trigger a build and run tests on a server for each commit to a repository and get notified about their status. The latter is currently used in MAGRITTE. Furthermore, we also automatically update the online documentation, hosted on READTHEDOCS[11], at `magritte.readthedocs.io`. The documentation is part of the source code

[7]See also `cmake.org`.
[8]See also `www.doxygen.nl`.
[9]See also `www.jenkins.io`.
[10]See also `travis-ci.org`.
[11]See also `readthedocs.org`.

and written as ReStructuredText, which is built with Sphinx[12]. To incorporate the Doxygen-style C++ inline documentation into the online Sphinx-generated documentation, we used the convenient conversion package Breathe[13].

Now we have established how we designed and implemented the different parts of Magritte, we can focus on the particular methods that we implemented.

3.3 Methods

In this section, we highlight some of the methods used in Magritte. It is not an exhaustive list, but rather some highlights of important and perhaps peculiar aspects.

3.3.1 Discretisation

Discretisation is a key step in the numerical solution of any problem with continuous variables. It is crucial, since it puts a hard upper bound on the maximum achievable accuracy and largely determines the computational cost. In fact, proper discretisation is so important that we will discuss it in detail in a dedicated chapter: Chapter 5. Here, we only discuss the minimalist design choices that were made on how to represent the geometry of a model in Magritte.

Radiative transfer solvers are more often than not used in combination with other simulations, for instance, for making synthetic observations of a model. As a result, most of the discretisation is usually inherited from a previous simulation step, which is usually some sort of hydrodynamics model. There are various widely different discretisation schemes used in hydrodynamics, ranging form regular Cartesian to hierarchically refined grids, all the way to smoothed-particle and finite element methods. In order to accommodate for all of these types of input, we reduced the number of assumptions we make about the geometry to an absolute minimum. As a result, in Magritte, we only assume to know the locations of the points or cells in the geometry and their nearest neighbours. This is just enough information to trace a path through the model (see also Section 3.3.2 on ray-tracing). Furthermore, those two properties can easily be derived from any type of discretisation scheme. We

[12]See also `www.sphinx-doc.org`.

[13]See also `breathe.readthedocs.io`.

should note that our choice of representation of the model does not fully determine the geometric discretisation of the radiative transfer equation in Magritte. This is further determined by the ray-tracing algorithm, as described in Section 3.3.2.

In retrospect, this might not be the optimal choice, since model geometries can often be converted quite easily (and cheaply) form one format into another. Furthermore, as we will see in Chapter 5, a pre-processing step is often required anyway to optimise the geometry for radiative transfer computations. Therefore, a better approach would be to search for an optimal solver design and adapt the model geometry to that (see also Chapter 7). Nevertheless, although perhaps not optimal, the current design has already resulted in a useful tool that lead to interesting insights.

3.3.2 Ray-tracing

In order to solve the radiative transfer equation along a certain ray, the emissivity and opacity of the points that are encountered along that ray are required. Moreover, the path length that the ray traces through the neighbourhood of each point must be computed. Since we aimed to keep the ray-tracing algorithm as general and discretisation agnostic as possible, we choose to only assume the locations of points and their nearest neighbours (see also Section 3.3.1).

The ray-tracing algorithm in Magritte was originally inspired by 3D-PDR [42], but was significantly simplified and optimised, especially since the latter assumed that all ray paths could be stored in memory, an assumption that impeded scaling beyond geometries with more than 10^5 points. The idea behind the ray-tracing algorithm in Magritte is to walk along the ray from one point to the next and determine the path length through the neighbourhood of each point by projecting the points onto the ray. To determine which point is next, the set of all nearest neighbours of the current point is considered. From this set, the neighbour is chosen which is closest to the ray and whose projection on the ray lies farther than that of the current point. This procedure is then repeated until a point on the boundary is reached. Figure 3.1 shows a visual example of this algorithm. Once a pair of antipodal rays is traced, a straight line through the entire model can be formed along which the radiative transfer equation can be solved.

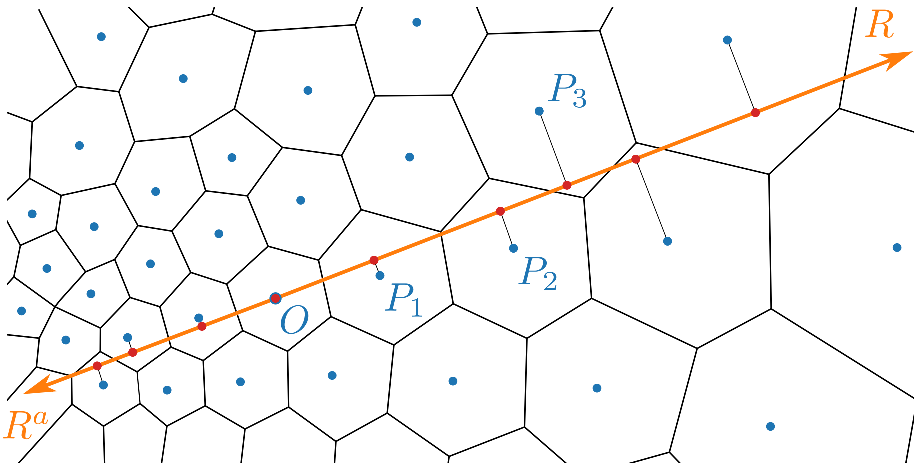

Figure 3.1: A visual representation of the ray tracing algorithm in Magritte for a ray, R, originating from a point O. The neighbourhood of each point is visualised by the corresponding Voronoi cell. The goal is to find which points are encountered along the ray. Clearly, the point O itself lies on the ray. The next point encountered is the neighbour of O that lies closest to the ray. We call this point P_1. Now the next point to be projected is the neighbour of P_1 that lies closest to the ray and is farther away from O than P_1. The last condition is there to ensure that one proceeds along the ray towards the boundary. This process is repeated until a boundary point is reached.

Especially in line radiative transfer, it is crucial to properly sample the velocity field, since too large a step in velocity from one point to the next can Doppler-shift a line directly from one wing to the other without capturing the effect of the core of the line. Therefore, in each step from one point to the next, the change in velocity along the ray is computed and checked for large variations. If the velocity field, and thus the resulting Doppler shift, changes too much the emissivity and opacity are interpolated between the points, such that the velocity steps are only a certain (user defined) fraction of the local line width. In this way, we avoid losing or improperly accounting for line contributions due to an inadequate sampling of the velocity field.

Ray-tracing in 1D (spherically symmetric) models The ray-tracing algorithm described above can also be adapted to work in 1D (spherical symmetric) models. The main complication is the way in which the geometry for these models is usually stored. Because of the spherical symmetry, it suffices to define spatial points in the interval, $[R_{\text{in}}, R_{\text{out}}]$, with R_{in} and R_{out}, respectively defined as the inner and outer radius of the model. A ray through this geometry will thus either move up from one layer to the next layer above until R_{out} is reached, or it will move down form one layer to the next layer below. This later case, however, is slightly more complicated, since the ray will travel down until a certain layer is reached corresponding to the impact parameter of the ray, after which the ray will travel up again (in terms of the distance from the centre) until again R_{out} is reached. Apart form this slight complication that is not encountered in the general 3D case, ray-tracing can go on exactly as before.

3.3.3 Solvers

As described in Chapter 2, there are two main equations that alternately need to be solved in the Lambda iteration approach to line radiative transfer: the radiative transfer equation (1.1), and the equation of statistical equilibrium (2.23). Next, we briefly describe the solvers that are used for each of these equations in MAGRITTE.

Radiative transfer Once a ray is traced (as described in Section 3.3.2), the radiative transfer equation has to be solved along that ray. In MAGRITTE, this is done in the second-order Schuster-Feautrier form that was introduced in Section 1.3.4. The Schuster-Feautrier equation is solved using an improved version of the tridiagonal matrix algorithm (or Thomas algorithm), as proposed by Rybicki & Hummer [91]. In addition, we use the improved boundary conditions, as proposed by Auer [108]. This is used as the default solver in MAGRITTE. Optionally, we also provide a fourth-order (or Hermitian) solver, based on an improved version of the discretised Schuster-Feautrier euqation by Auer [109]. The derivations and details about our particular optimised implementation of these solvers can be found in Appendix A.

Statistical equilibrium The statistical equilibrium equation (2.23) is a sparse linear system that can readily be solved with any standard (optionally sparse) linear equation solver. In MAGRITTE, we use the sparse LU-solver provided in the EIGEN[14] library [67]. Since solving the radiative transfer equation is orders of magnitude more computationally expensive than solving the statistical equilibrium equation, we did not optimise the latter.

[14]See also `eigen.tuxfamily.org`.

3.3.4 Imaging

Once the state of the medium and the corresponding radiation field are known, one can infer how the model would appear in observations, by making synthetic images. In Magritte, a synthetic observation viewed from a certain direction is constructed by considering the outgoing intensities at the endpoints of all rays in that direction. The positions of these endpoints are then projected on the plane orthogonal to the viewing direction. This results in a 2D point cloud in the image plane, each with a corresponding intensity. Finally, an image can be obtained by interpolating the intensities form the 2D point cloud onto a regular grid with the desired resolution. This interpolation can easily be achieved, for instance, with the `griddata` method in Scipy [110]. Since we use the data from every ray, the 2D point cloud usually highly over-samples the image, such that a simple (and fast) nearest neighbour interpolation often already suffices. Examples of the resulting synthetic observations created with Magritte can be found in Chapter 4.

3.4 Tests & benchmarks

To demonstrate the validity of our methods and to better understand their limitations, we have conducted a series of comparisons with analytical models and benchmarked against established radiative transfer solvers.

3.4.1 Semi-analytical tests

To assess the accuracy of the ray tracer and radiative transfer solver in Magritte, we first reproduce some semi-analytically solvable line radiative transfer models. This will help us later to better assess the uncertainties associated with the results of our simulations. We cannot overemphasise the importance of these analytical tests as they are the only way to obtain absolute measures of the accuracy (in contrast to the estimates for the uncertainty presented in Chapter 6).

3.4.1.1 Homogeneous Hubble-Lemaître models

As a first test, consider the transfer of a single line, $i \leftrightarrow j$, on a uniformly spaced grid with constant molecular abundance, temperature distribution, and velocity gradient.

The velocity distribution is thus given by the Hubble-Lemaître law,

$$v(r) \ = \ c\Delta\beta\, r, \tag{3.1}$$

where we scaled the velocity, within the velocity gradient $\Delta\beta$, with the speed of light, c. The boundary condition is given by incoming cosmic microwave background (CMB) radiation, i.e a black-body spectrum, B_ν, of temperature $T_{\text{CMB}} = 2.725$ K. If we assume LTE level populations, the line source function, $S_{\nu_{ij}}$, is spatially constant (since the temperature was assumed to be constant). In that case, one can find the mean intensity by directly integrating the transfer equation,

$$J_\nu(\boldsymbol{x}) \ = \ S_{\nu_{ij}} \ + \ \left(B_\nu \ - \ S_{\nu_{ij}}\right) \oint \frac{\text{d}\Omega}{4\pi}\, e^{-\tau_\nu(\boldsymbol{x},\hat{\boldsymbol{n}})}, \tag{3.2}$$

in which the optical depth, τ_ν, assuming Gaussian line profiles centred around, ν_{ij}, and with a line profile width, $\delta\nu_{ij}$, is given by,

$$\tau_\nu(\ell) \ = \ \frac{\chi_{ij}}{2\nu\Delta\beta}\left\{\text{Erf}\left[\frac{\nu-\nu_{ij}}{\delta\nu_{ij}}\right] \ - \ \text{Erf}\left[\frac{\nu\left(1-\Delta\beta\ell\right)-\nu_{ij}}{\delta\nu_{ij}}\right]\right\}, \tag{3.3}$$

where Erf is the error function, and $\ell(\boldsymbol{x},\hat{\boldsymbol{n}})$ is the distance from point $\boldsymbol{x}$ to the boundary, as measured along the ray in direction $\hat{\boldsymbol{n}}$. Since the Hubble-Lemaître velocity law is both translation and rotation invariant, only the total distance to the boundary appears in the expression for the optical depth. We consider two different cases: first, a truly one-dimensional case, considering the mean intensity flowing up and down a single ray, and second, the mean intensity in the full (spatially) three-dimensional yet spherically symmetric model.

Considering only a single ray in the interval, $[0, R]$, the mean intensity at a point $r \in [0, R]$, as expressed in equation (3.2), reads,

$$J_\nu(r) \ = \ S_{\nu_{ij}} \ + \ \frac{1}{2}\left(B_\nu \ - \ S_{\nu_{ij}}\right) \left[e^{-\tau_\nu(r)} + e^{-\tau_\nu(R-r)}\right], \tag{3.4}$$

in which the averaging integral over all directions reduces to the half sum of the intensity flowing up and down the ray.

In three dimensions, assuming a spherical boundary with radius, R, the mean intensity expressed in equation (3.2) reduces to,

$$J_{\nu}(r) \ = \ S_{\nu_{ij}} \ + \ \frac{1}{2}\left(B_{\nu} - S_{\nu_{ij}}\right) \int_{0}^{\pi} \text{d}\theta \ \sin\theta \ e^{-\tau_{\nu}(\ell(r,\theta))}, \tag{3.5}$$

in which the distance to the boundary, $\ell(r,\theta)$, is given by,

$$\ell(r,\theta) \ = \ r\cos\theta \ + \ \sqrt{R^2 - r^2\sin^2\theta}. \tag{3.6}$$

There is no analytic expression for the θ-integral in the mean intensity (3.5), however, it can easily be computed numerically.

Note that introducing the spherical boundary breaks the translation invariance of the problem. As a result, both solutions (3.5) and (3.4) depend on the radial distance from the centre of the boundary.

Although these are simple models, they can demonstrate some key issues in numerical radiative transfer modelling. In particular, both models can be used to directly assess the accuracy of the radiative transfer solver, and to test the sampling in velocity space. Especially in line radiative transfer it is crucial to properly sample the velocity field. This can be tested by adjusting the velocity gradient. By considering both the single ray and the full 3D model, we can also assess the quality of the spatial interpolations onto the rays.

In our test setup, we used a fictitious two-level species in a (radially) uniformly spaced grid, $[0,R]$, with $R = 495$ km, and a velocity gradient $c\Delta\beta = 0.01$ s^{-1}. The line data for the fictitious two-level species are summarised in Table 3.1. We assume a constant H_2 number density $n^{H_2} = 1.0 \times 10^{12}$ m^{-3}, and a constant fractional abundance of the fictitious two-level species $X \equiv n^{\text{fict}}/n^{H_2} = 10^{-4}$. For the level populations, we assume LTE with a constant temperature distribution $T = 45$ K. Moreover, we assume the gas has no turbulent velocity component, $v_{\text{turb}} = 0$ m/s. The corresponding 3D model is obtained from the 1D model by mapping each 1D grid point to a shell of 3D grid points uniformly distributed over a sphere. The model parameters for Magritte can be found in Table 3.2.

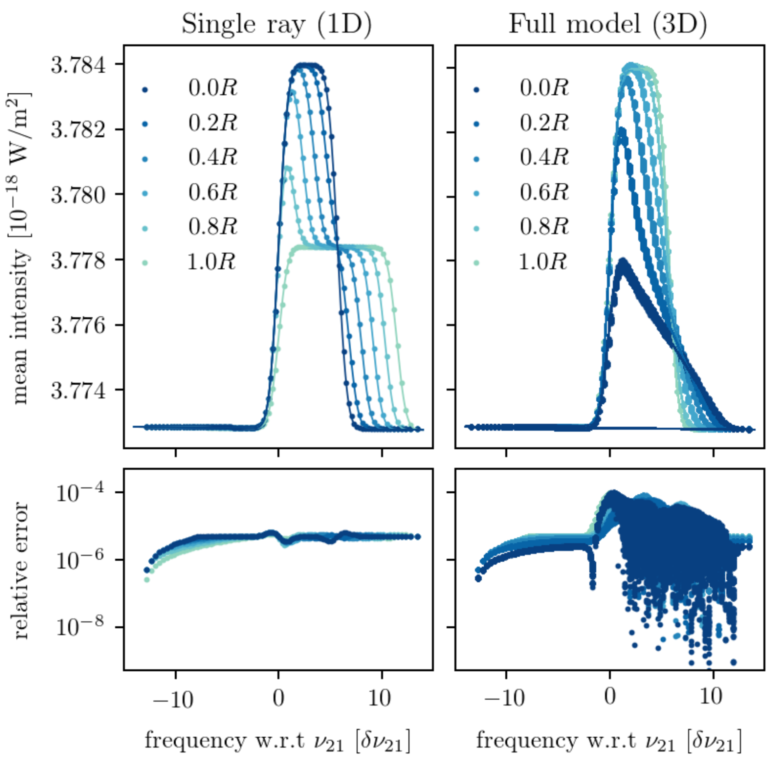

Figure 3.2: Comparison between MAGRITTE and the semi-analytical solution of the mean intensity as a function of frequency, evaluated at different radii in the Hubble-Lemaître model. The dots indicate results obtained with MAGRITTE and the lines represent the analytic results. Frequencies are expressed with respect to the line centre, $\nu_{21} \approx 179.88$ GHz, as a fraction of the line profile width, $\delta\nu_{21} \approx 519.03$ kHz. The relative error of two values is measured as twice the absolute difference over their sum.

Table 3.1: Line data of the fictitious two-level species. This is the same fictitious two-level species as used in problem 1 in the van Zadelhoff benchmark [93].

$E_2 - E_1$ [cm^{-1}]	g_2/g_1	A_{21} [s^{-1}]	$K_{21}^{\text{H}_2}$ [cm^3/s]
6.0	3.0	1.0×10^{-4}	2.0×10^{-10}

Figure 3.2 shows a comparison between the solution of MAGRITTE and the semi-analytical solutions (3.4 and 3.5) of the Hubble-Lemaître models. The numerical results obtained with MAGRITTE clearly agree with the analytic solution with a relative error well below 10^{-4} almost everywhere, where the relative error of two values is measured as twice the absolute difference over their sum.

3.4.1.2 Simple power-law density distribution

As a second test, consider the transfer of a single line, $i \leftrightarrow j$, on a logarithmically spaced grid, with a constant temperature distribution, with no velocity field, and a density distribution corresponding to the singular isothermal sphere, given by,

$$n^{\text{H}_2}(r) \ = \ \begin{cases} 0 & \text{for } r < R_{\text{in}} \\ n^{\text{H}_2}(R_{\text{in}}) \left(\frac{R_{\text{in}}}{r}\right)^2 & \text{for } r \geq R_{\text{in}} \end{cases} \tag{3.7}$$

where R_{in} is the inner radius of the model. The boundary condition is given by incoming CMB radiation, i.e. a black-body spectrum, B_ν, of temperature $T_{\text{CMB}} =$ 2.725 K. If we again assume LTE level populations, the line source function, $S_{\nu_{ij}}$, is spatially constant. As a result the mean intensity is again given by equation (3.2). To compute the optical depth, one needs to integrate the density distribution along every ray. Assuming a spherical boundary with radius, R, the optical depth reads,

$$\tau_\nu(r, \theta) \ = \ \frac{\chi_{ij} \ \phi_\nu^{ij} \ r}{\sin\theta} \left(\frac{\pi}{2} \ - \ \theta \ + \ \arccos\left(\frac{r \sin\theta}{R}\right) \ - \ f(r, \theta)\right), \tag{3.8}$$

where $f(r, \theta)$ accounts for rays going through the empty core ($r < R_{\text{in}}$), and reads,

$$f(r, \theta) \ \equiv \ \begin{cases} 2 \arccos\left(\frac{r \sin\theta}{R_{\text{in}}}\right) & \text{for } \ \theta < \theta_{\text{core}}, \\ 0 & \text{for } \ \theta \geq \theta_{\text{core}}. \end{cases} \tag{3.9}$$

Whether or not a ray passes through the empty core is determined by the direction of the ray at each radius, $\theta_{\text{core}} = \arcsin(R_{\text{in}}/r)$. We consider again both a truly one-dimensional single ray model and a full three-dimensional solution.

Considering only a single ray in the interval $[-R, R]$, the mean intensity reads,

$$J_\nu(r) \ = \ S_{\nu_{ij}} \ + \ \frac{1}{2}\left(B_\nu - S_{\nu_{ij}}\right) \ \left[e^{-\tau_\nu(r,0)} + e^{-\tau_\nu(r,\pi)}\right], \tag{3.10}$$

where the average over all directions reduces to half the sum of the intensity flowing up and down the ray. Note that both limits, $\theta \longrightarrow 0$, and, $\theta \longrightarrow \pi$, are well-defined.

In three dimensions, one can simply integrate over the entire solid angle to obtain the mean intensity,

$$J_\nu(r) \;=\; S_{\nu_{ij}} \;+\; \frac{1}{2}\left(B_\nu - S_{\nu_{ij}}\right) \int_0^\pi \mathrm{d}\theta \; \sin\theta \; e^{-\tau_\nu(r,\theta)}. \tag{3.11}$$

However, one should be careful in distinguishing between rays that do and rays that do not pass through the empty core of the model.

For this test, we used the same fictitious two-level species as before (Table 3.1) in a radially logarithmically spaced grid, $[R_{\text{in}}, R]$, with, $R_{\text{in}} = 1.0 \times 10^{13}$ m, and $R = 7.8 \times 10^{16}$ m. The H_2 number density just outside the empty core is $n^{H_2}(R_{\text{in}}) = 2.0 \times 10^{13}$ m^{-3}, and a constant fractional abundance of the fictitious two-level species $X \equiv n^{\text{fict}}/n^{H_2} = 10^{-6}$ is used. To obtain the level populations, we assume LTE with a constant temperature distribution $T = 20$ K. Furthermore, the gas has a turbulent velocity component, $v_{\text{turb}} = 150$ m/s. The 3D model is obtained from the 1D model by mapping each 1D grid point to a shell of 3D grid points uniformly distributed over a sphere. The model parameters for Magritte can be found in Table 3.2. This model setup is identical to problem 1b in [93]. However, here we are only interested in the resulting radiation field when the levels are in LTE (see also Section 3.4.2).

Figure 3.3 shows a comparison between the solution of Magritte and the semi-analytical solutions (3.10 and 3.11) of the simple models with a power-law density distribution. The numerical results obtained with Magritte clearly agree with the analytic solution. Only at the steep edges of the line, there is a larger relative error (~ 0.4), which can be attributed to the steepness of the model solution that is not properly resolved by the discrete grid.

3.4.2 Cross-code benchmarks

There are no analytic solutions for the full non-LTE line radiative transfer problem, so the only way to fully test the line radiative transfer module in Magritte is with benchmarks against established solvers. Although this does not prove the validity of our code, it is already reassuring to find the same results in different ways.

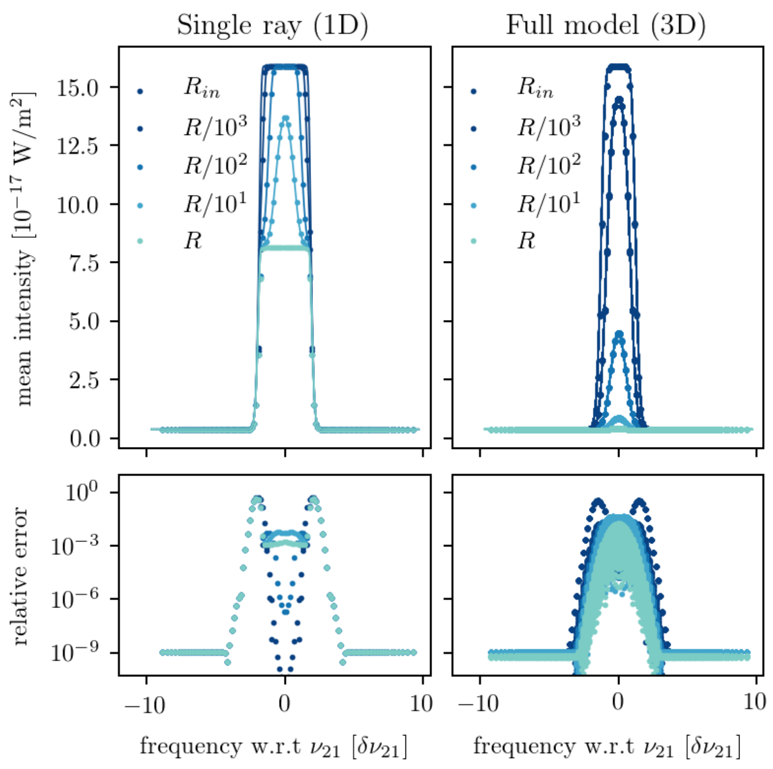

Figure 3.3: Comparison between MAGRITTE and the semi-analytical solution of the mean intensity as a function of frequency in a model with a simple power-law density distribution, evaluated at different radii. The dots indicate results obtained with MAGRITTE and the lines represent the analytic results. Frequencies are expressed with respect to the line centre, $\nu_{21} \approx 179.88$ GHz, as a fraction of the line profile width, $\delta\nu_{21} \approx 357.53$ kHz. The relative error of two values is measured as twice the absolute difference over their sum.

Table 3.2: MAGRITTE parameters for the semi-analytic test models.

model		(N_{shells})	N_{points}	N_{rays}	N_{q}
Hubble-Lemaître	1D	50	100	2	100
	3D	50	12 528	192	100
density distribution	1D	50	100	2	100
	3D	50	12 528	192	100

Table 3.3: MAGRITTE parameters for the benchmark models.

model	(N_{shells})	N_{cells}	N_{rays}	N_{q}
Problem 1 a/b/c/d	50	23 280	192	24
Problem 2 a/b	50	23 280	192	24

For the benchmarks we used the problems presented by van Zadelhoff et al. in [93], and compared our results with the publicly available version of the 1D Monte Carlo radiative transfer code RATRAN[15] [47]. Since MAGRITTE is intrinsically multidimensional, the 1D models were mapped to their 3D equivalents by mapping each 1D grid point to a shell of 3D grid points uniformly distributed over a sphere. The MAGRITTE parameters of these models can be found in Table 3.3.

3.4.2.1 Van Zadelhoff problem 1 a/b

The first test, referred to as problem 1 a/b in [93], considers a fictitious two-level species in a spherically symmetric cloud, without velocity field, with a constant temperature distribution, and a density distribution given by a power law. The entire model is thus defined analytically. This setup is essentially equivalent to the simple power-law density distribution model in Section 3.4.1.2. The only difference is that in problem 1a the relative molecular abundance, $X = 10^{-8}$, results in a low optical depth, whereas in problem 1b, $X = 10^{-6}$, yields a relatively high optical depth.

Figure 3.4 shows a comparison between the level populations for problem 1a/b obtained with MAGRITTE and RATRAN. Both are clearly in good agreement with each other, with relative errors well below 0.05, which can be attributed to the fact that MAGRITTE uses a 3D model instead of a manifestly spherically symmetric 1D model as used in RATRAN.

3.4.2.2 Van Zadelhoff problem 1 c/d

Since line radiative transfer critically depends on a proper sampling of the velocity field along the line of sight of each ray, it is worthwhile to test if this is properly accounted for. Therefore, we consider again benchmark problem 1 a/b from the previous paragraph, but this time with a non-zero velocity field. Although this test

[15]The source code can be found at `personal.sron.nl/~vdtak/ratran/frames.html`.

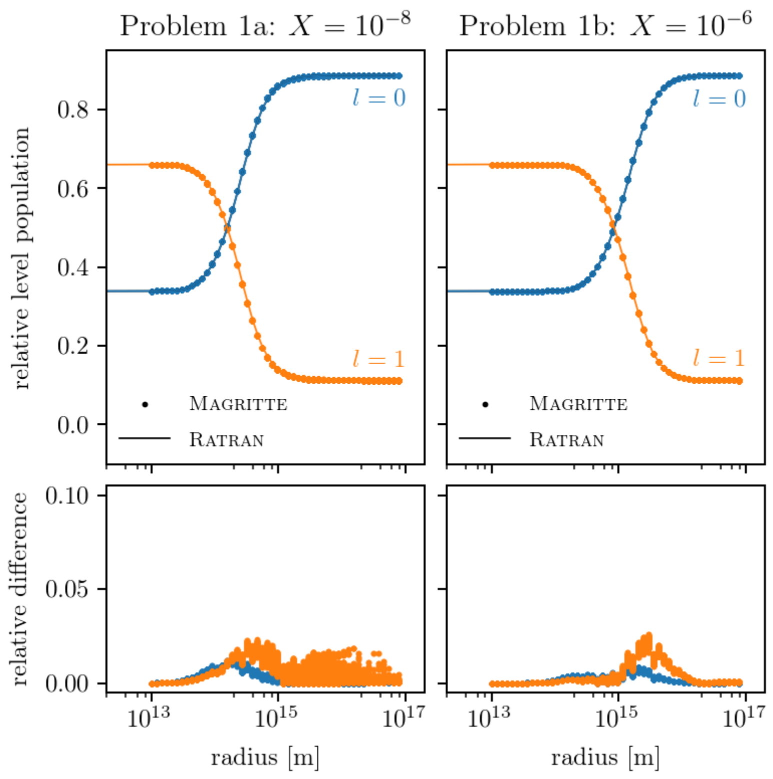

Figure 3.4: Comparison of the results for problem 1 a/b of the van Zadelhoff benchmark [93] obtained with Magritte (dots) and Ratran (lines). The relative difference of two values is measured as twice the absolute difference over their sum.

was not part of the van Zadelhoff benchmark [93], we can still compare our results with Ratran. We consider a velocity field that is pointing radially outward,

$$\mathbf{v}(r) \ = \ v_{\infty} \left(\frac{r - R_{\text{in}}}{R - R_{\text{in}}} \right)^{\gamma} \hat{\mathbf{r}}. \tag{3.12}$$

In the benchmarks below we used $\gamma = 0.5$, and since it is the same model setup as in Sections 3.4.1.2 and 3.4.2.1, the inner radius is $R_{\text{in}} = 1.0 \times 10^{13}$ m. Furthermore, we consider two different terminal velocities, $v_{\infty} = 10$ km/s, and $v_{\infty} = 50$ km/s.

Figure 3.5 shows a comparison between the level populations for problem 1 c/d obtained with Magritte and Ratran. Both are clearly in good agreement with each other. However, in order to obtain this result, we needed to increase the number of grid points in Ratran by a factor of 10 (resulting in 500 logarithmically spaced grid points). For any lower number of grid points, Ratran had difficulty properly sampling the velocity field and produced significantly different results from

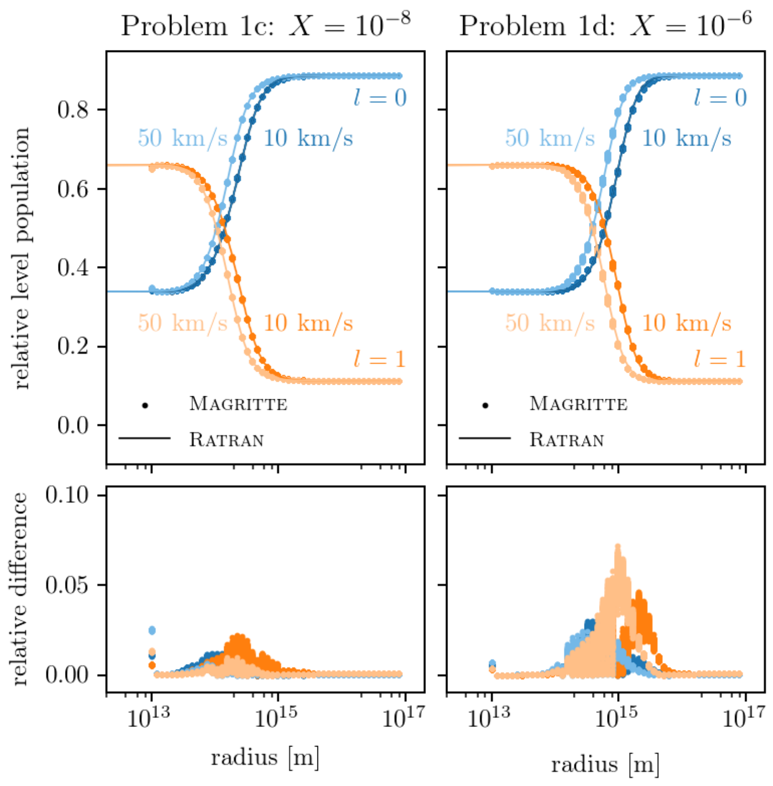

Figure 3.5: Comparison of the results for problem 1 c/d obtained with Magritte (dots) and Ratran (lines). The indicated velocities are the v_∞ for each model. The relative difference of two values is measured as twice the absolute difference over their sum.

Magritte. This clearly demonstrates the power of the automatic interpolation along a ray for large velocity gradients, as implemented in Magritte (see Section 3.3.2).

3.4.2.3 Van Zadelhoff problem 2 a/b

The third test has a more realistic setup and considers the lines of HCO^+ in a snapshot of an inside-out collapse model by Shu [111]. This is referred to as problem 2 a/b in [93]. The parameters describing the input model were taken from the website of the benchmark[16]. The model consists of 50 logarithmically spaced grid points. In each grid point the radial velocity, gas temperature, micro-turbulence, and both HCO^+ and H_2 abundances are given. Again there are two cases, one with a relatively low optical depth where the fractional HCO^+ abundance is, $X = 10^{-9}$, and one with a higher optical depth where the relative molecular abundance is, $X = 10^{-8}$.

[16]Benchmark website: `www.strw.leidenuniv.nl/astrochem/radtrans/`.

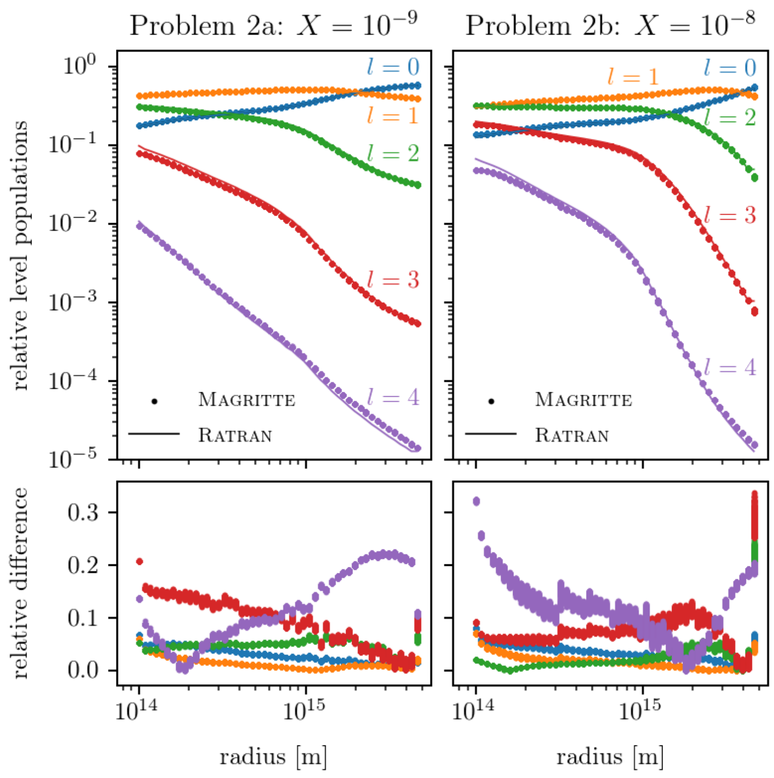

Figure 3.6: Comparison of the results of the first 5 levels (of 41) for problem 2 a/b of the van Zadelhoff benchmark [93] obtained with Magritte (dots) and Ratran (lines). The relative difference of two values is measured as twice the absolute difference over their sum.

Figure 3.6 shows a comparison between the results for problem 2 a/b obtained with Magritte and Ratran. Overall, both codes agree well with relative differences below 0.3 for the first five levels. These differences can again be attributed to the fact that Magritte uses a 3D model instead of a manifestly spherically symmetric 1D model, as used in Ratran. Furthermore, these results are comparable to what Brinch & Hogerheijde find in their Figure 10 for Lime [48], and what Rundle et al. find in their Figures 2 and 3 for Torus [112]. Furthermore, Rundle et al. report for $l = 0$ a relative deviation from the benchmark [93] of less than 5% [112], which is also comparable to what we find.

3.4.3 Performance scaling

Parallelising a program to distribute computations over N compute nodes or N processor cores, does not necessarily imply an N-fold speed-up. In fact, it only

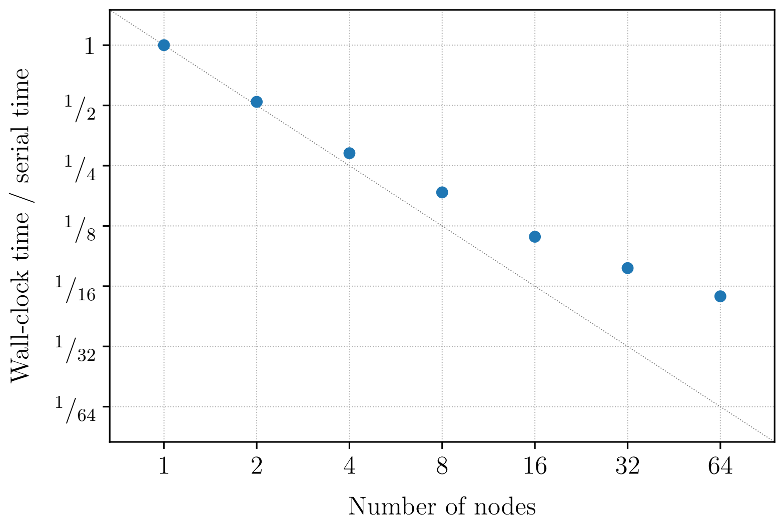

Figure 3.7: Strong scaling plot for the multi-node parallelism in MAGRITTE.

implies an upper bound on the achievable speed-up, since usually not every part of the program can be parallelised and there is overhead, since the work sharing has to be managed between the parallel components. Therefore, in order to understand the gains from parallelising a program, and gauge how efficiently it can utilise given computational resources, the scaling of the execution time has to be measured.

3.4.3.1 Multi-node

Figure 3.7 shows the scaling of the computation time of MAGRITTE for problem 1a in the van Zadelhoff benchmark [93], when using different numbers of compute nodes in a system. The scaling strongly depends on the load balancing, in this case, i.e. the balance between the computational cost of solving the transfer equation along the ray pairs, as they are distributed over the nodes. Since the model from problem 1a is spherically symmetric, load balancing should be almost perfect and cannot be the reason for the deviation from perfect scaling, observed in Figure 3.7. Here, the deviation is largely due to the work-load per node becoming too small for larger numbers of nodes, such that the relative contribution of the overhead increases. Remember that to measure strong scaling, one requires a model that can run decently on a single node, as well as on the maximum number of nodes. As a result, one often ends up with a rather artificial setup that would normally not be executed on such a large number of nodes.

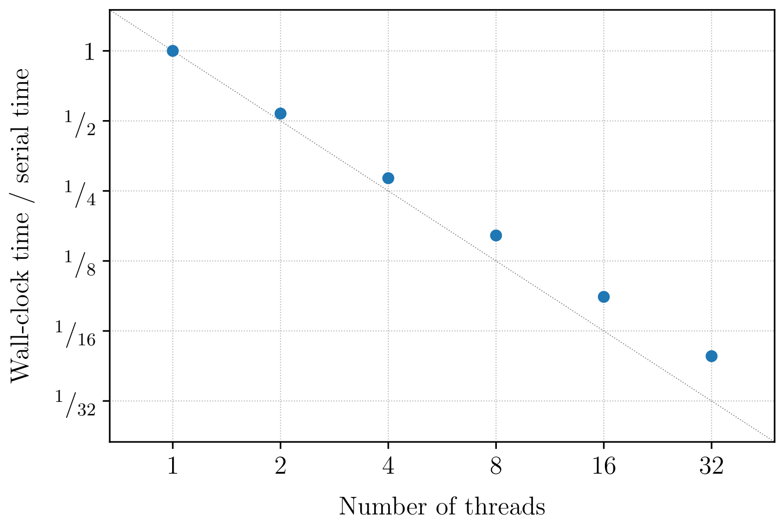

Figure 3.8: Strong scaling plot for the multi-core parallelism in MAGRITTE.

3.4.3.2 Multi-core

Figure 3.8 shows the almost perfect scaling of the computation time of MAGRITTE for problem 1a in the van Zadelhoff benchmark [93], when using different numbers of cores in a processor. Since the number of points will always be orders of magnitude larger than the number of cores in a processor, the load balancing will almost always be ideal. Since, furthermore, no communication is required, the overhead is minimal, resulting in almost perfect scaling.

Throughout this chapter, we described the design of MAGRITTE, its practical implementation, and demonstrated its excellent performance on classical benchmark problems. In the next chapter, we take this further and consider some examples of real scientific applications.

Chapter 4

Astrophysical applications

Everything we see hides another thing.
We always want to see, what is hidden by what we see.

– René Magritte

4.1 Introduction

The scientific method in astronomy is somewhat peculiar, since most theoretical models cannot be verified by direct experiments, but rather have to be inferred indirectly from observations. As a result, for any theoretical model, a key question is: how does it appear in observations? For observations of electromagnetic radiation, this is a question about radiative transfer, in particular, about how much radiation is emitted (or how much background radiation is absorbed) towards the observer.

With the advent of high-resolution spectroscopic imaging, for instance, using the Atacama Large (sub-)Millimetre Array (ALMA), we can make observations in unprecedented detail. The detailed images show us complexities that cannot longer be ignored in models, such as, the distinct non-spherical morphologies recently observed in the stellar winds around evolved stars by Decin et al. [113]. These complexities make it very difficult to interpret the data. For the observations in [113] alone, already tens of thousands of 3D hydro-chemical models are being created to interpret the data (see e.g. [114–116], and several forthcoming papers). To be able to compare these models with the data, in order to understand the observed phenomena, and to draw scientific conclusions, for instance, about the origins of these complex stellar winds, we need to know how they would appear in (synthetic) observations.

This can be done with Magritte, presented in Chapter 3. As other radiative transfer solvers were deemed insufficiently accurate and too computationally expensive for this task (see e.g. [28]), Magritte was especially designed to efficiently handle this dauntingly large and diverse set of models (see also Chapter 5).

In this chapter, we demonstrate how, Magritte can be applied, in particular, to create synthetic observations of these intricate hydro-chemical stellar wind models.

4.2 Synthetic observations

There is a hierarchy of several levels of detail that can be incorporated in synthetic observations, see e.g. [117]. At the most basic level, they can be just a simple estimate for the emission, based on some physical parameters of the model, whereas, on the most advanced level, they are the result of a full radiative transfer model, possibly even including instrumental effects. Here, we define a synthetic observation as an image for each frequency bin, of all the radiation that escapes the model in a certain direction, against the cosmic microwave background (see also Section 3.3.4). We do not include any re-scaling based on distance or possible instrumental effects.

In the following, we consider snapshots of hydrodynamics models of the stellar outflow produced by a mass-losing asymptotic giant branch (AGB) star, as it is perturbed by a companion. A detailed discussion of how these models with different discretisation schemes can be represented in Magritte will be given in Chapter 5.

4.2.1 AMR models

As a first example, we consider an MPI-AMRVAC [118] hydro-chemical model[1] of a mass-losing AGB star, that was kindly provided by Jan Bolte [119]. The AGB star has a mass of 1 $M_\odot$, an effective temperature of 2900 K, a radius of 0.9 AU, and a pulsation period of 1 year. The companion has a mass of 0.5 $M_\odot$, and follows a circular orbit around the AGB star, with a (constant) separation of 10 AU. The model uses a hierarchically refined Cartesian mesh, of which the cell centres (and their nearest neighbour lists) are used as input geometry for Magritte.

[1]The same model is used as an example in Figure 8 in [21]. For a movie of the hydrodynamics model and the synthetic observations, see `youtu.be/DQ3tj1EwMmM`.

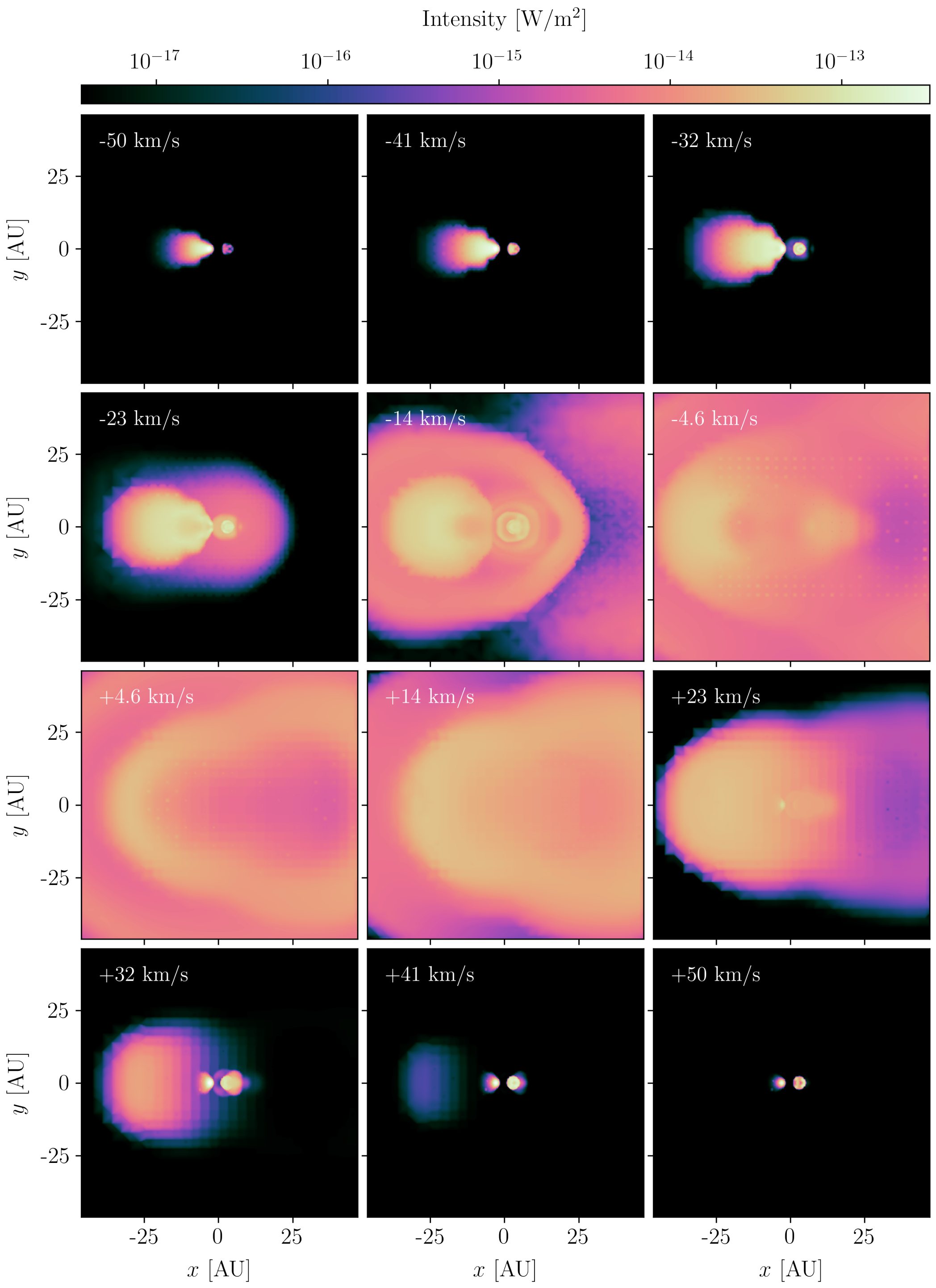

Figure 4.1: Channel maps for the CO($v = 0$, $J = 1-0$)-transition in an edge-on MAGRITTE synthetic observation of a companion-perturbed AGB wind model, simulated with MPI-AMRVAC.

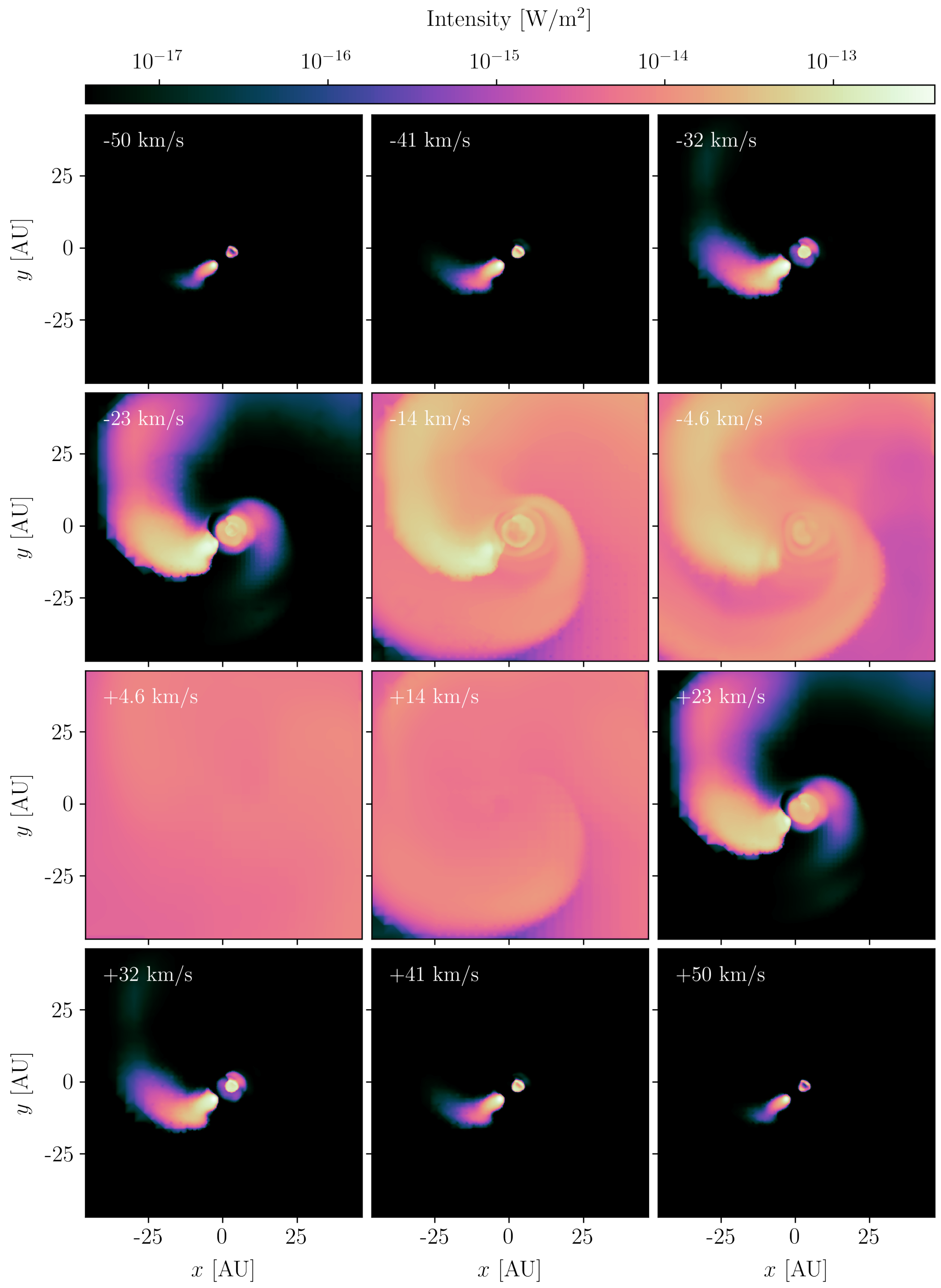

Figure 4.2: Channel maps for the CO(v = 0, J = 1 – 0)-transition in a face-on MAGRITTE synthetic observation of a companion-perturbed AGB wind model, simulated with MPI-AMRVAC.

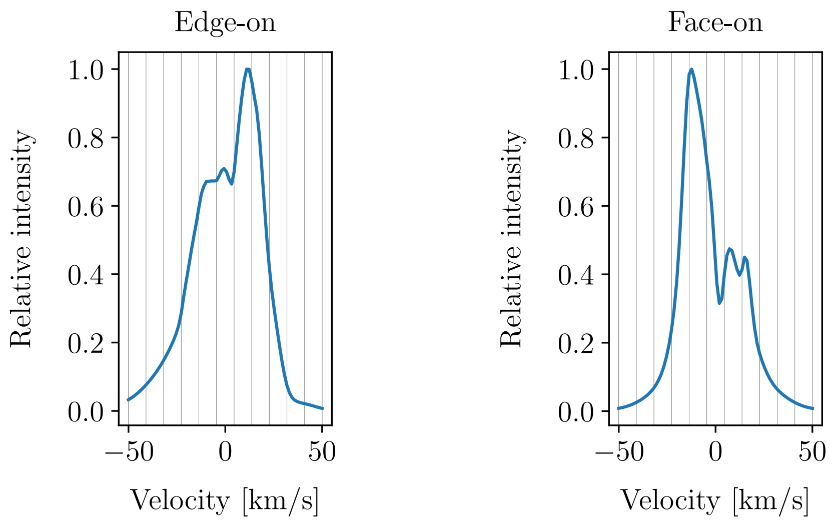

Figure 4.3: Spectral line for the CO($v = 0$, $J = 1 - 0$)-transition, obtained by integrating over the corresponding channel maps for the edge-on (*left*; Figure 4.1) and face-on (*right*; Figure 4.2) synthetic observations. The vertical grey lines indicate the velocities of the corresponding channel maps.

Figures 4.1 and 4.2 show the channel maps, created with Magritte, for the CO($v = 0$, $J = 1 - 0$)-transition in respectively an edge-on and a face-on view on the companion-perturbed AGB wind model. The synthetic observations are taken at time step, $t = 33.3$ years, in the hydro-chemical model. Figure 4.3 shows the corresponding spectral lines for these channel maps. These lines are obtained by integrating over the channel maps, without including any beam effects.

One can clearly distinguish the different spiral components in the synthetic observations. There is one clear spiral emanating behind the companion, caused by its gravity wake, and one fainter spiral emerging behind the AGB star, caused by its reflex motion. Furthermore, in the higher-velocity channel maps of the face-on view, one can neatly see the wind material glancing off the companion.

4.2.2 SPH models

As the next example, we consider two Phantom [120] hydrodynamics models of mass-losing stars, kindly provided by Silke Maes [115] and Jolien Malfait [116].

First, we consider the regular spiral outflow created by an AGB star of mass 1.5 $M_\odot$, an effective temperature of 3000 K, an effective radius of 1.267 AU, and a wind initiated at 20 km/s, with a mass-loss rate of 10^{-4} $M_\odot$/yr. The companion has a mass of 1 $M_\odot$, and resides on a circular orbit with a (constant) separation of 9 AU. This model is referred to as S90Fast in [115]. The model uses smoothed-particle hydrodynamics, of which the particles (and their nearest neighbour lists) can directly be used as input geometry in Magritte. Since no chemistry is included, we derive the H_2 density from the mass density (assuming H_2 accounts for all the mass), and assume a constant fractional abundance, $n^{\text{CO}}/n^{\text{H}_2} = 10^{-4}$, for the CO number density.

Figures 4.4 and 4.5 show the channel maps, created with Magritte, for the CO($v = 0$, $J = 1-0$)-transition in respectively an edge-on and a face-on view on the companion-perturbed AGB wind model. The synthetic observations are taken after 5 orbits of the companion in the hydrodynamics model. Figure 4.6 shows the corresponding spectral lines for these channel maps.

We can clearly see the regular spiral structure emerging in both synthetic observations. One distinct difference with the MPI-AMRVAC model from Section 4.2.1 is the symmetry along the line centre in frequency space. This is partially due to the much larger length-scale at which we are observing, and partially due to the much more radial nature of the velocity field in the model, see also [115].

The previous two examples show the applicability of Magritte for models that exhibit a morphology that can relatively easily be interpreted. To demonstrate the level of geometric complexity that can be modelled with Magritte, we present a final hydrodynamics model of a mass-losing AGB star with a companion in an eccentric orbit, modelled with Phantom. The AGB star has a mass of 1.5 $M_\odot$, an effective temperature of 3000 K, an effective radius of 1.267 AU, and a wind initiated at 10 km/s, with a mass-loss rate of 2×10^{-6} $M_\odot$/yr. The companion has a mass of 1 $M_\odot$, and resides on an elliptical orbit with a semi-major axis of 6 AU, and an eccentricity of 0.5. This model is referred to as v10e50 in [116]. Since no chemistry was included, the CO number density is again derived assuming a constant fractional abundance, $n^{\text{CO}}/n^{\text{H}_2} = 10^{-4}$, and assuming that H_2 is responsible for all the mass density.

Figures 4.7 and 4.8 show the channel maps, created with Magritte, for the CO($v = 0$, $J = 1-0$)-transition in respectively an edge-on and a face-on view on the

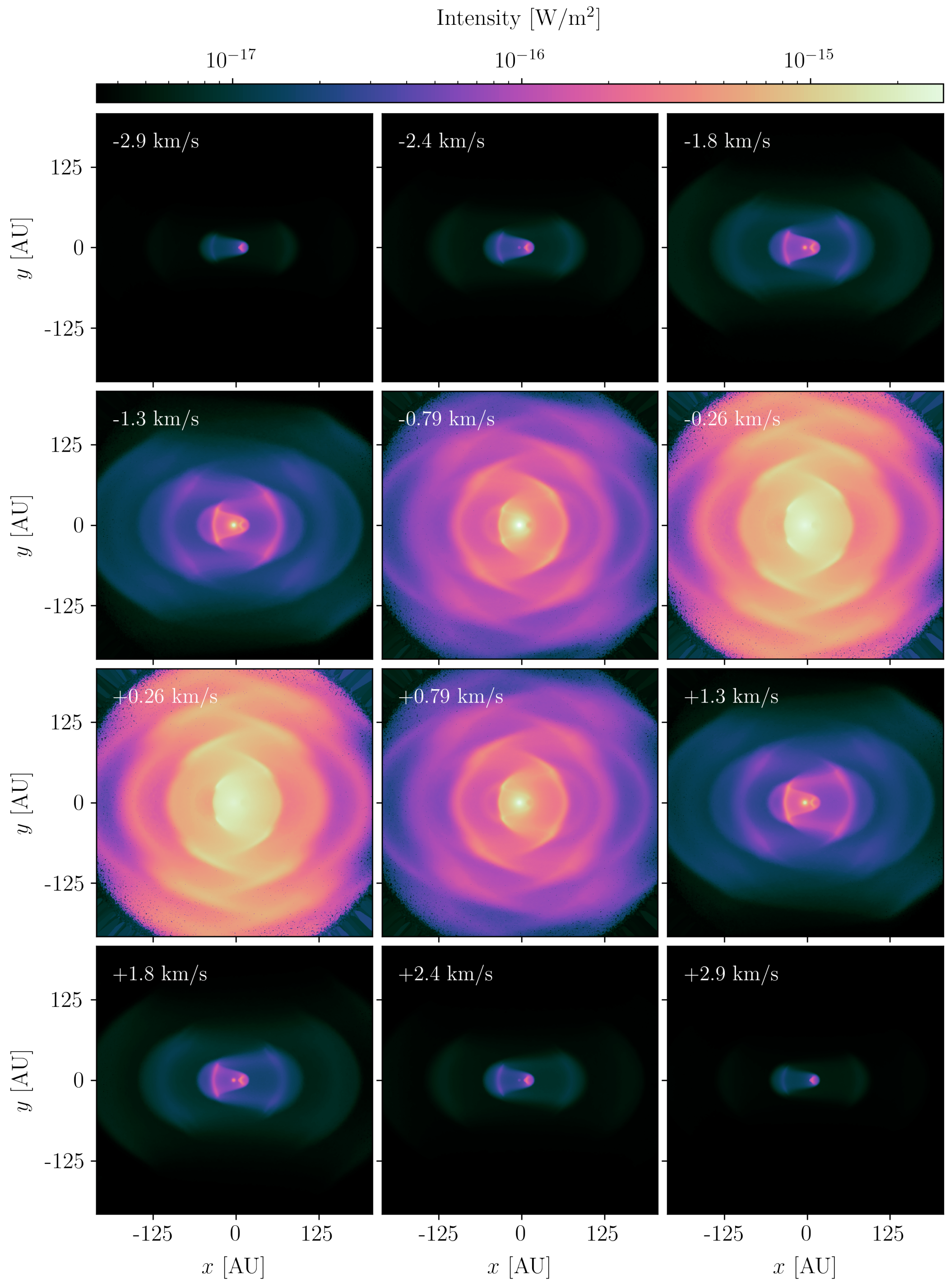

Figure 4.4: Channel maps for the CO($v = 0$, $J = 1-0$)-transition in an edge-on Magritte synthetic observation of a companion-perturbed AGB wind model, simulated with Phantom.

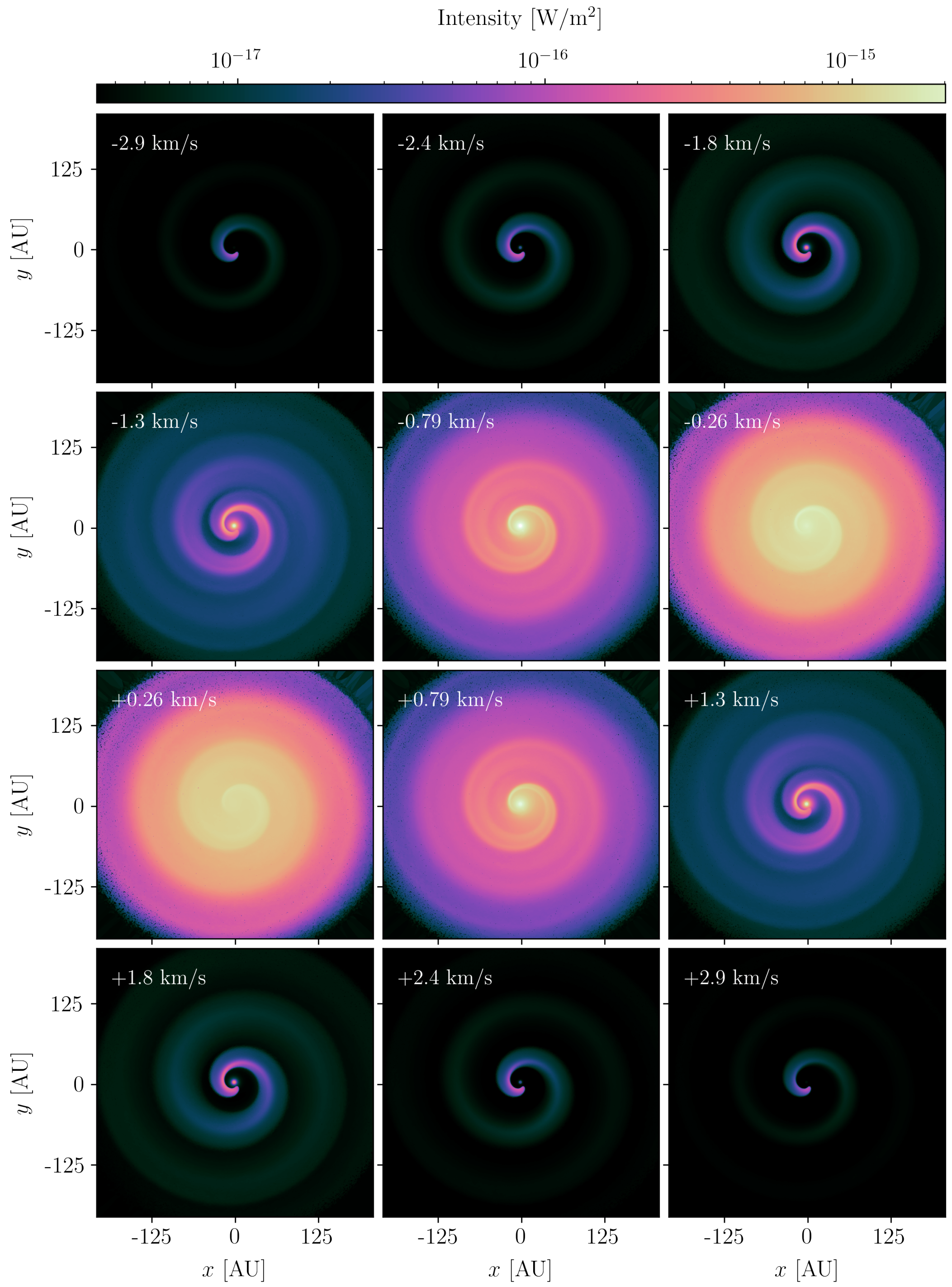

Figure 4.5: Channel maps for the CO($v = 0$, $J = 1 - 0$)-transition in a face-on Magritte synthetic observation of a companion-perturbed AGB wind model, simulated with Phantom.

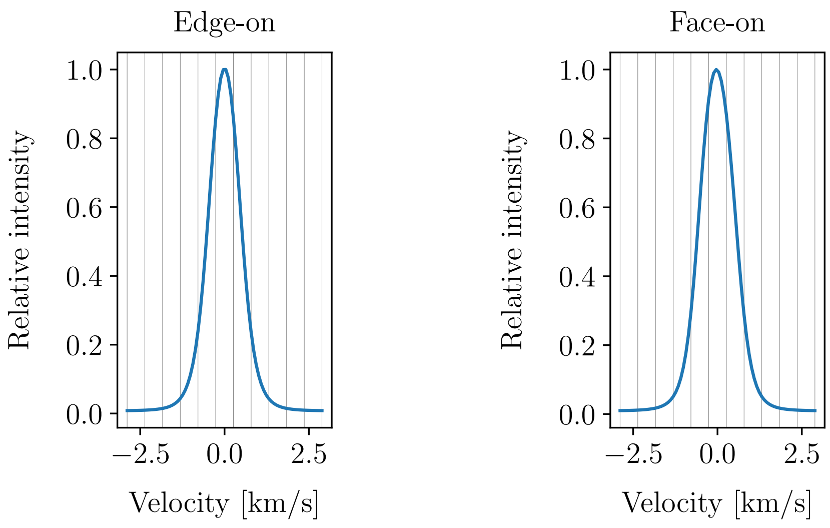

Figure 4.6: Spectral line for the CO($v = 0, J = 1-0$)-transition, obtained by integrating over the corresponding channel maps for the edge-on (*left*; Figure 4.4) and face-on (*right*; Figure 4.5) synthetic observations. The vertical grey lines indicate the velocities of the corresponding channel maps.

eccentric companion-perturbed AGB wind model. The synthetic observations are taken at time step, $t = 55.7$ years, in the hydrodynamics model. Figure 4.9 shows the corresponding spectral lines for these channel maps.

The frequency dependence is again very symmetric around the line centre, due to the radial nature of the velocity field, see also [116]. Nevertheless, the channel maps neatly demonstrate the amount of complexity and morphological richness that can be resolved with the more than one million particles in the model.

Synthetic observations for AGB wind models are a beautiful demonstration of the complexity that can be modelled with Magritte. However, they are only one of the many (astrophysical) applications in which Magritte can be applied.

4.3 Other applications

Radiative transfer models are an essential part of many astrophysical simulations. For instance, Lime [48], which is in a sense a predecessor of Magritte (see e.g. [28]), has been used for creating synthetic observations in a wide range of astrophysical contexts: ranging from stellar environments [51, 55, 121, 122], to the accretion disk around an active galactic nucleus [123], aiding with the discovery of giant water

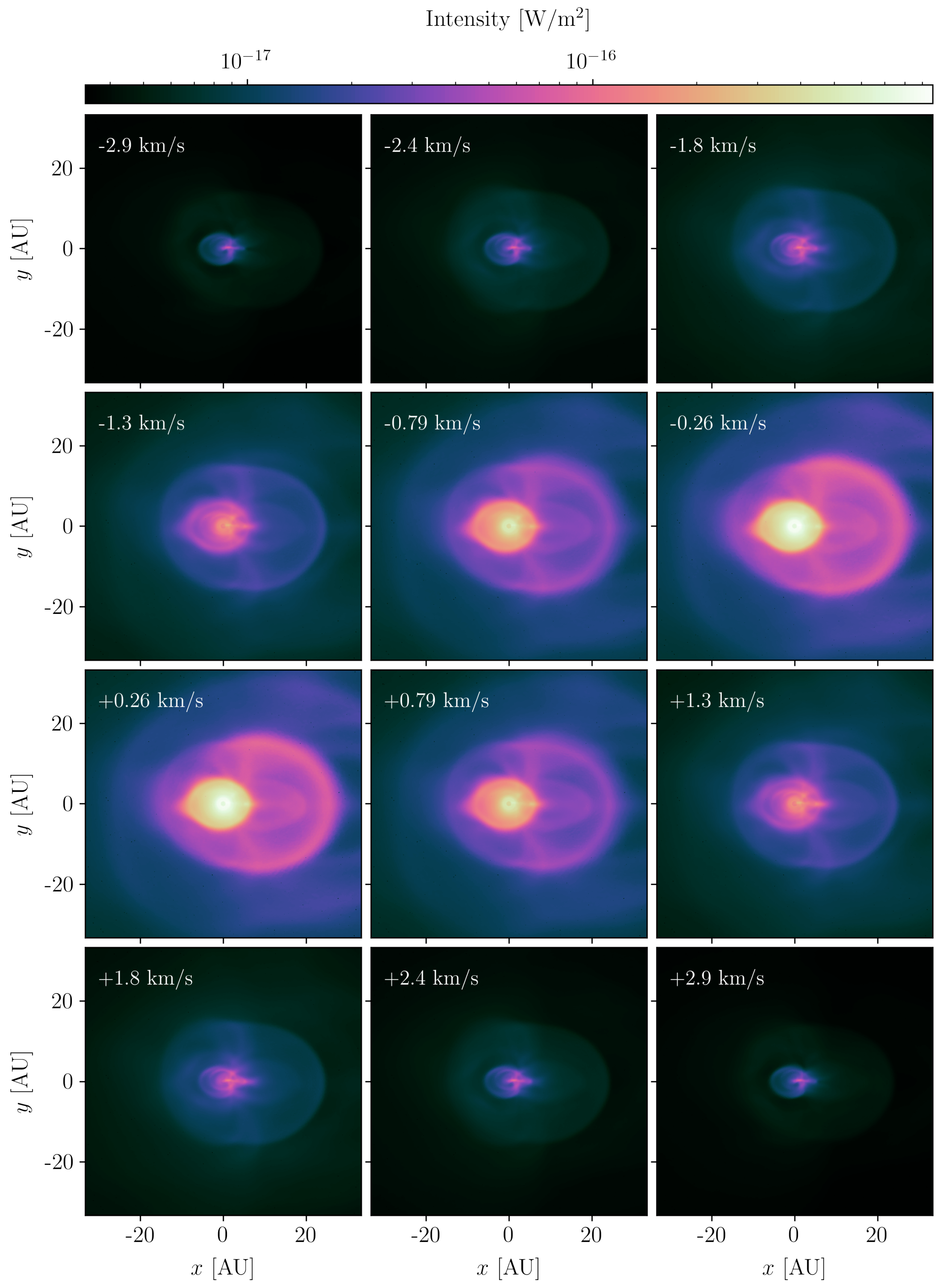

Figure 4.7: Channel maps for the CO(v = 0, J = 1 – 0)-transition in an edge-on MAGRITTE synthetic observation of the eccentric companion-perturbed AGB wind model, simulated with PHANTOM.

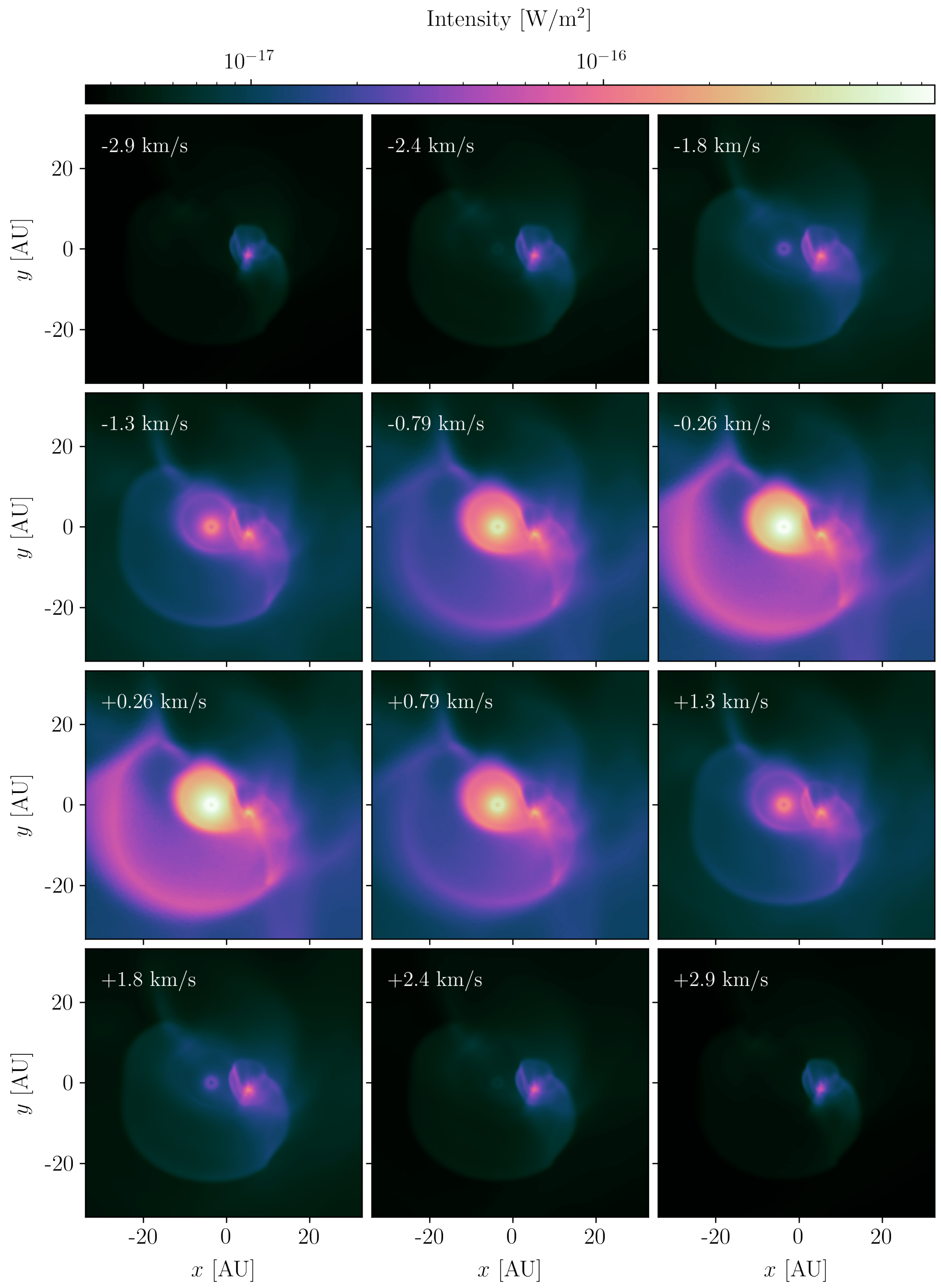

Figure 4.8: Channel maps for the CO($v = 0$, $J = 1 - 0$)-transition in a face-on MAGRITTE synthetic observation of the eccentric companion-perturbed AGB wind model, simulated with PHANTOM.

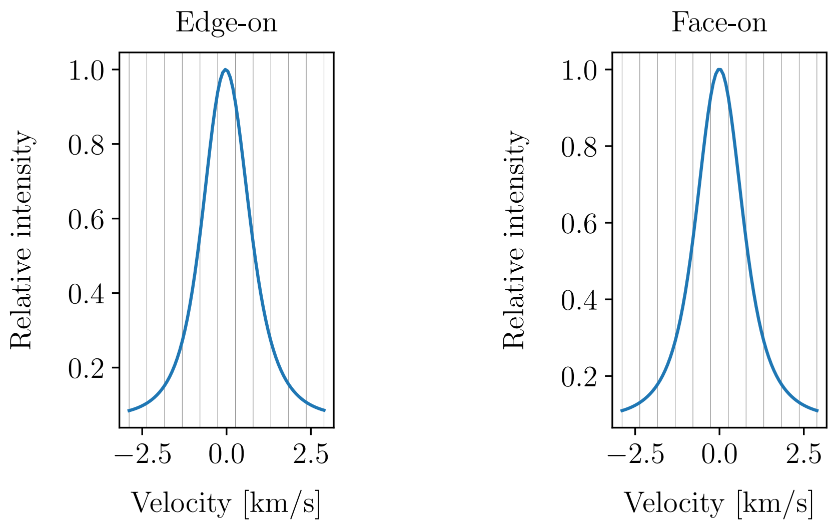

Figure 4.9: Spectral line for the CO($v = 0, J = 1-0$)-transition, obtained by integrating over the corresponding channel maps for the edge-on (*left*; Figure 4.7) and face-on (*right*; Figure 4.8) synthetic observations. The vertical grey lines indicate the velocities of the corresponding channel maps.

reservoirs in a forming planetary system [124], and complex organic molecules in a protoplanetary disk [125]. Since Magritte provides very similar functionality and improved performance (see e.g. [28]), one can easily envision its use in an equally broad array of astrophysical research.

Furthermore, although currently full 3D radiative transfer is often still highly approximated in most simulations, given the current performance of Magritte, it is starting to become feasible to incorporate on-the-fly radiative transfer, for example, in the heating, cooling, or photo-chemistry of hydro-chemical models. However, this is still work in progress, and would, currently, only be practical for a very limited number of models (see also Chapter 7). Meanwhile, reliable solvers, such as Magritte, also can be used to devise better approximations for the radiative transfer in these models, for instance, by emulation (see also Sections 1.5.2.1 and 7.2).

In this chapter, we established that Magritte can be a useful tool in a broad range of astrophysical research, and explicitly demonstrated this by making synthetic observations for several hydro-chemical AGB wind models. In the next chapter, we look in more detail at how these and other models are represented in Magritte.

Chapter 5

Discretisation

Triangles are my favourite shape.
Three points where two lines meet.
Toe to toe, back to back let's go, my love, it's very late.
'Til morning comes, ooh, let's tessellate.

– alt-j (Δ), *Tessellate*

5.1 Introduction

The first step in building a simulation is finding a proper representation for the simulated objects. Often, this comes down to finding an appropriate discretisation for all physical quantities. This is an essential step, since the number of elements in the discretisation will not only determine the direct memory cost of the model but also its computational cost and ultimately the maximal achievable accuracy of the simulation. Finding an appropriate discretisation scheme is thus a question of optimising the trade-off between accuracy and computational cost.

Well-established solvers for radiation transport, for instance, OpenMC[1] [126], Tripoli-4[2] [127], and MCNP[3] [128], which are used in industrial applications such as nuclear engineering and medical imaging, use surface-based or combinatorial representations as underlying geometrical models. These are constructed using computer-aided design (CAD) software and consist of combinations of well-defined shapes (e.g. cuboids, cylinders, spheres), of certain materials, with well-defined boundary surfaces. As a result, the radiation transport can be considered through

[1]See also `openmc.org`.
[2]See also `www.cea.fr/nucleaire/tripoli-4`.
[3]See also `laws.lanl.gov/vhosts/mcnp.lanl.gov`.

one material at a time and rays can be traced from one surface to the next, similar to the rendering techniques used in modern computer graphics, see e.g. [129]. The resulting highly accurate representations of the models allow for the highly accurate solutions that are required in these types of applications.

In contrast, in astrophysical and cosmological simulations, one is interested in the radiation field in fluids with continuously varying properties throughout the model. Therefore, the geometries of these models always have to be discretised before they can be numerically solved, leading to a first inescapable source of numerical error. The geometries that are used are often inherited from a previous simulation step. Typically, these radiative transfer solvers are used on top of a hydrodynamics solver to compute for instance the radiative pressure, or they are used to post-process snapshots of hydrodynamics simulations to produce synthetic observations. In those cases, the radiative transfer solver uses the same geometric model mesh as the hydrodynamics solver, although those meshes are usually only optimised for the latter.

Over the years, the spatial discretisation schemes used in hydrodynamics solvers have evolved from static structured meshes, to hierarchical meshes resulting from adaptive mesh refinement (AMR, [130]), to unstructured and dynamically evolving meshes [131]. Additionally, there are mesh-less smoothed-particle hydrodynamics (SPH) solvers that do not rely on a mesh, but rather evolve a set of particles with appropriate smoothing kernels [132, 133]. The spatial discretisation schemes used in radiative transfer simulations evolved accordingly from structured to unstructured meshes [134], and further to mesh-less schemes [28, 42]. For an assessment of the use of unstructured Voronoi meshes in (Monte Carlo) radiative transfer, see e.g. [135], and for the use of hierarchical octree and the more general k-d tree meshes see e.g. the work by Saftly et al. [136, 137].

In further contrast to the more industrial radiative transfer applications, in astrophysics and cosmology, we are almost never interested in a highly accurate solution of a specific model, but rather in understanding, for instance, the more general driving mechanisms that govern a set of models. For example, where in nuclear engineering it is crucial to be able to accurately describe the effect of adding a single fuel rod in a reactor, it is far less important in cosmology, for instance, to be able to describe the effect of one additional filament in the cosmic web. This allows us to optimise our models more aggressively, retaining only the essential features under investigation.

In this chapter, we will have a closer look at the discretisation of models in the MAGRITTE software library (see also Chapter 3), and we propose some heuristic methods to optimise the discretisation for radiative transfer computations.

5.2 Discretisation

The radiation field is a function of position, direction, and frequency. Therefore, in the following sections, we distinguish between the spatial, spectral, and directional discretisations in MAGRITTE.

5.2.1 Spatial discretisation

There are many different types of spatial discretisation schemes, each tailored to their specific use cases. Over the years, there has been a clear evolution from structured schemes, like e.g. regular Cartesian grids, to unstructured schemes, like e.g. Voronoi grids or smoothed particle hydrodynamics (SPH) discretisations.

Since we aimed to build a general-purpose library that can easily be integrated with other codes, we did not want to tie MAGRITTE to a certain discretisation scheme. Instead, we designed our algorithms such that they only require data that can easily be deduced from any discretisation scheme, and yet allow us to efficiently trace rays and solve the transfer equation. As a result, the ray-tracing algorithm used in MAGRITTE, presented in Section 3.3.2, only requires the positions of the cell centres[4] (or equivalently the positions of the particles in an SPH scheme) and the nearest neighbours lists for each cell (or particle). Hence, the input is effectively a point cloud complemented with nearest neighbour information. The boundary of the model can then be defined as the convex hull of the point cloud.

[4]We do not require a strict definition of the cell centre. If we define a cell as a unit in the discretisation of the spatial volume, then the cell centre may be any point in that volume. (We only use the cell centre to locate the cell.)

In principle, one could use a separate spatial discretisation to sample the density, velocity and temperature distributions of the model. In practice, however, one usually samples all three on the same discretisation. This is the case, especially for hydrodynamics computations, where all these parameters should be sampled equally well. However, for radiative transfer computations, especially when considering lines, it is essential to properly sample any changes in the velocity field along a certain line of sight. Since this effect depends on the velocity field in a certain direction it is difficult to fully take this into account in the spatial discretisation. In MAGRITTE, when detecting large changes in the velocity field along a ray, we make an appropriate interpolation on-the-fly during ray-tracing, without adjusting the mesh (see also Section 3.3.2). Therefore, the effective discretisation that is used by the solver in MAGRITTE is ultimately determined by the ray-tracer.

5.2.2 Spectral discretisation

The requirements for the spectral discretisation vary for different stages of the radiative transfer simulation. For instance, when determining the level populations, we are only interested in the radiation in the lines, whereas when computing spectra we require a proper frequency sampling over the full spectrum. To accommodate this, MAGRITTE can change its spectral discretisation throughout a simulation.

At the stage where the level populations are obtained, the frequency bins are distributed to suit the integration of the radiation field over the line. In MAGRITTE, these integrals are evaluated using quadrature formulae. Assuming a Gaussian line profile, the corresponding Gauss-Hermite quadrature for any frequency dependent function, $y(\nu)$, is given by,

$$\int_0^\infty \mathrm{d}\nu \; \phi_\nu^{ij}(\boldsymbol{x}) \; y_\nu \;\; \longrightarrow \;\; \sum_{n=1}^{N_\mathrm{q}} w_n \; y\left(\nu_{ij} + r_n \delta\nu_{ij}\right), \tag{5.1}$$

where N_q is the number of quadrature points and the quadrature weights read,

$$w_n \; = \; \frac{2^{N_\mathrm{q}-1} N_\mathrm{q}!}{\left(N_\mathrm{q} H_{N_\mathrm{q}-1}(x_n)\right)^2}, \tag{5.2}$$

in which H_{N_q-1} is the physicists' version of the Hermite polynomial and the r_n are the roots of the physicists' version of the Hermite polynomial $H_{N_q}(x)$ (see e.g. [138]). To be able to easily evaluate these quadratures in Magritte, we define a separate set of frequency bins for each point in the model, given by,

$$\left\{\nu_{ij} + r_n\,\delta\nu_{ij}(\boldsymbol{x}),\ \text{for each transition } ij \text{ and root } r_n\right\}, \tag{5.3}$$

possibly appended with additional frequency bins. Note that this set has to be different for each point, since it depends on the local line profile width, $\delta\nu_{ij}(\boldsymbol{x})$.

At the stage where spectra or images are created, extra frequency bins can be appended to the list above to improve the sampling of the spectrum.

5.2.3 Directional discretisation

Magritte is a ray-tracing code, i.e. the radiation field is determined by solving the radiative transfer equation along a set of rays (straight lines) originating from each point in the model. A ray can be defined by a point and a direction. The directions of the rays determine the solid-angular resolution, which is critical, for instance, in the computation of the mean intensity, and, when considering scattering, it implicitly discretises the possible scattering angles.

In general, there are no preferred directions. Therefore, initially, we discretise the directions uniformly. In 1D and 2D models this is trivial. In 3D, we determine the direction of a ray using the HEALPix[5] discretisation of the sphere [139]. Given a level of refinement, l, it discretises a unit sphere in $N_{\text{rays}} = 12 \times 4^l$ uniformly distributed pixels of equal area. For each pixel, there is an associated unit vector pointing from the origin of the sphere to the pixel centre. These unit vectors determine the directions of the rays in Magritte for 3D simulations. Hence, a directional average for a quantity, $y(\hat{\boldsymbol{n}})$, can be translated into an average over rays,

$$\frac{1}{4\pi}\oint \mathrm{d}\Omega\, y(\hat{\boldsymbol{n}}) \quad\rightarrow\quad \frac{1}{N_{\text{rays}}}\sum_{r=1}^{N_{\text{rays}}} y_r. \tag{5.4}$$

[5]See also `healpix.sourceforge.net`.

It should also be noted this does not correspond to any exact quadrature scheme, in contrast to the spectral discretisation. The uniform sampling also bears the danger of missing the contributions of very localised sources of emissivity or opacity. Furthermore, there might be situations in which there is one or more preferred directions, and one might better consider a non-uniform distribution of ray directions. Therefore, in Section 5.3.3, we investigate an adaptive directional weighting scheme, which is part of our more generally adaptive approach to ray-tracing.

5.3 Adaptive ray-tracing

Various methods have been devised to compute the radiation field in a given medium by solving the radiative transfer equation. MAGRITTE employs a ray-tracer and formal solver and thus solves the radiative transfer equation along a set of predefined rays through the model. The ray-tracer follows a straight line through the model and gathers the emissivities and opacities along that ray, which are then used to solve the radiative transfer equation. By stepping from one point to the next it indirectly determines the optical depth increments, and thus the step size, in the discretised transfer equation used by the solver. The first version of MAGRITTE employed a second-order radiative transfer solver [28, 33], but the current version also provides the more accurate (Hermitian) fourth-order scheme by Auer [109], with second and third-order boundary conditions respectively [108] (see also Appendix A). In the following sections, we describe three ways in which the rays can be adapted to a particular model to further improve the accuracy of the solver.

5.3.1 Adaptive velocity sampling

Since Doppler shifts can cause significant variations in emissivity and opacity within a frequency bin, it is crucial to properly sample the velocities encountered along a ray. This is even more important for line radiative transfer, where significant changes occur in particularly narrow frequency ranges. In Section 3.3.2, it was already discussed how MAGRITTE accounts for this by interpolating the velocity along a ray between two points if its change is too large. As we will see below, this can be extended by applying a similar approach to the optical depth increments.

5.3.2 Adaptive optical depth increments

In Magritte, the radiative transfer equation is solved in its second-order or Schuster-Feautrier form (Equation 1.16; see also Section 3.3.3 and Appendix A). The optical depth (within each frequency bin) is thus the relevant dependent variable. Assuming a proper sampling of the emissivity and opacity, the discretisation error will thus be determined mainly by the size of the optical depth increments. This means that a model mesh can perfectly sample the relevant radiative data, but nevertheless produce a large discretisation error. To resolve this, we adapt the optical depth increments as they are computed. In particular, we divide the interval on the ray between the (projected) points, n and $n+1$, in n_{inter} equal parts and linearly interpolate the emissivities and opacities on the sub-intervals. By defining the number of interpolations as,

$$n_{\text{inter}} \equiv \left\lceil \frac{\max\left\{\chi_n, \chi_{n+1}\right\} \Delta s_n}{\Delta \tau_{\max}} \right\rceil, \tag{5.5}$$

in which χ_n and χ_{n+1} denote the maximum opacities of the two consecutive points, and Δs_n is the distance increment along the ray, we can ensure that the optical depth increments along the ray are always smaller than a predefined value, $\Delta\tau_{\max}$.

It is important to note that the adaptive velocity sampling is still a separate process that happens before the adaptive optical depth increments are computed and that the former cannot be included in or replaced by the latter. For example, a large change in velocity along a ray which Doppler shifts a line from one wing to the other will not result in a particularly large optical depth increment, but will leave the line unaccounted for, increasing the error on the computed radiation field.

5.3.3 Adaptive directional discretisation

Many applications of radiative transfer computations require directional integrals over the radiation field to compute, for instance, the radiative heating or cooling, or to compute the radiative pressure. In those cases also the directions of the rays need to be discretised. Given a function, $y(\boldsymbol{x}, \hat{\boldsymbol{n}})$, the directional integral is discretised as,

$$\oint \text{d}\Omega \, y(\boldsymbol{x}, \hat{\boldsymbol{n}}) \quad \rightarrow \quad \sum_{r \in \mathcal{R}_i} w_{ir} \, y_{ir}, \tag{5.6}$$

where the i and r indicate the point and ray index respectively, and $\mathcal{R}_i$ is the set of rays originating from point i. The main difficulty is to determine an appropriate set of rays, $\mathcal{R}_i$, for each point. Once these are known, the corresponding weights, w_{ir}, can readily be computed.

Assuming that the points in the mesh properly sample the relevant distributions in the model, for each point in the mesh, the directions of the other points with respect to that point will properly sample the relevant directions for that point. Therefore, ideally, the discretisation of the directions for a point in the mesh would follow the distribution of the directions of the other points with respect to that point. Furthermore, since this discretisation has to be generated for every point in the mesh, the procedure cannot be too computationally demanding. Therefore, we propose a structured adaptively refining scheme based on HEALPix [139]. In Section 5.2.3, we already showed how HEALPix can be used to produce uniform directional discretisations. Now, by stitching together parts of the uniform HEALPix discretisations with different levels of refinement, we can obtain a locally refined directional discretisation adapted to the point density in the mesh.

Starting from a HEALPix discretisation with a minimal level of refinement, l_{min}, we can refine the pixels according to the distribution of the directions of the other points, until a predefined maximal level of refinement, say l_{max}, is reached. Figure 5.1 shows a Cartesian projection of an example of the resulting discretisation of the unit sphere. Since MAGRITTE considers pairs of antipodal rays, there is an antipodal symmetry in the directional discretisation.

To obtain the distribution of the directions of the other points with respect to the point under consideration, we draw a uniformly distributed sample of 10 000 points from the mesh and record the HEALPix pixels they belong to for each level of refinement from l_{min} up to and including l_{max}. In practice, we only need to

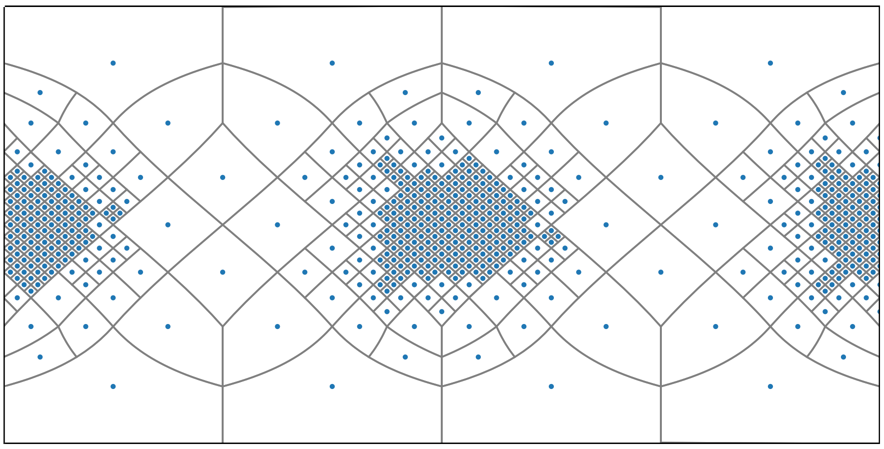

Figure 5.1: Example of the adaptive hierarchical discretisation of directions around a point, in Cartesian projection, using four different orders of the HEALPɪx scheme (from $l_{\text{min}} = 1$ to $l_{\text{max}} = 4$). More rays are traced in the directions with a higher mesh point density. The blue dots indicate the centres of the direction vectors and the grey lines delimit the HEALPix pixels.

compute the distribution at the highest level of refinement. The results for the other levels can be obtained by downgrading the resulting map, leveraging the nested ordering scheme in HEALPɪx [139]. To decide which pixels to refine, we order them according to the number of directions of other points belonging to them, and only refine the top half. To ensure that the same number of rays is traced for each point we fix the number of pixels that is refined at each level l to be 6×2^l. This implies that at each level (except l_{max}), half of the pixels is further refined.

Since HEALPɪx partitions the unit sphere in pixels of equal area, the weight corresponding to a pixel obtained with a level of refinement, $l(i,r)$, is the inverse of the number of pixels at that level,

$$w_{ir} = \frac{1}{12 \times 4^{l(i,r)}}, \tag{5.7}$$

where the level of refinement depends on the originating point, i, and the direction of the ray, r. This adaptive refinement scheme allows us to obtain a more efficient sampling for the directional dependence in the radiation field.

5.3.4 Other forms of adaptive ray-tracing

We should point out that our approach is quite different from the one proposed by Abel & Wandelt [140], who originally coined the term “adaptive ray-tracing” in the context of radiative transfer for cosmological simulations. Their idea, to split rays as they reach further away from their origin to obtain a more constant volume coverage, is ideal for direct solvers or short-characteristics methods but would be difficult to implement in our second-order solver. In contrast, our approach is more related to adaptive mesh refinement (AMR) methods, but applied on-the-fly along the ray and in the directional discretisation.

We were certainly not the first to propose a multi-resolution discretisation of the sphere using HEALPix. For instance, in [141] an hierarchical progressive survey (HiPS) scheme was proposed to represent astronomical data, using HEALPix tessellations of different orders. Moreover, in [142], an improved version of the HEALPix data structure was proposed, specifically tailored for a multi-resolution representation of large and sparse data sets. In our work, as our discretisation scheme is relatively simple, we only used the classical HEALPix scheme [139].

5.4 Spatial mesh construction & reduction

Over the years, many different algorithms have been devised to partition a given volume for use in a computer, see for instance the classic book by Thomson [143], the more recent treatment by George & Frey [144], and the references therein. Many of these algorithms have furthermore been implemented in various software libraries. For all models in this thesis we have used the open-source meshing library, called Gmsh[6], by Geuzaine & Remacle [145]. Gmsh provides various methods to generate a tetrahedral Delaunay mesh for a domain given a desired local element size distribution. Since the local element size is directly related to the local edge lengths of the tetrahedra, it allows us to control the step sizes along the rays traced through the domain. Although Magritte does not require a complete and fully consistent mesh (it only requires a point cloud and nearest neighbour information), the Delaunay

[6]See also `gmsh.info`.

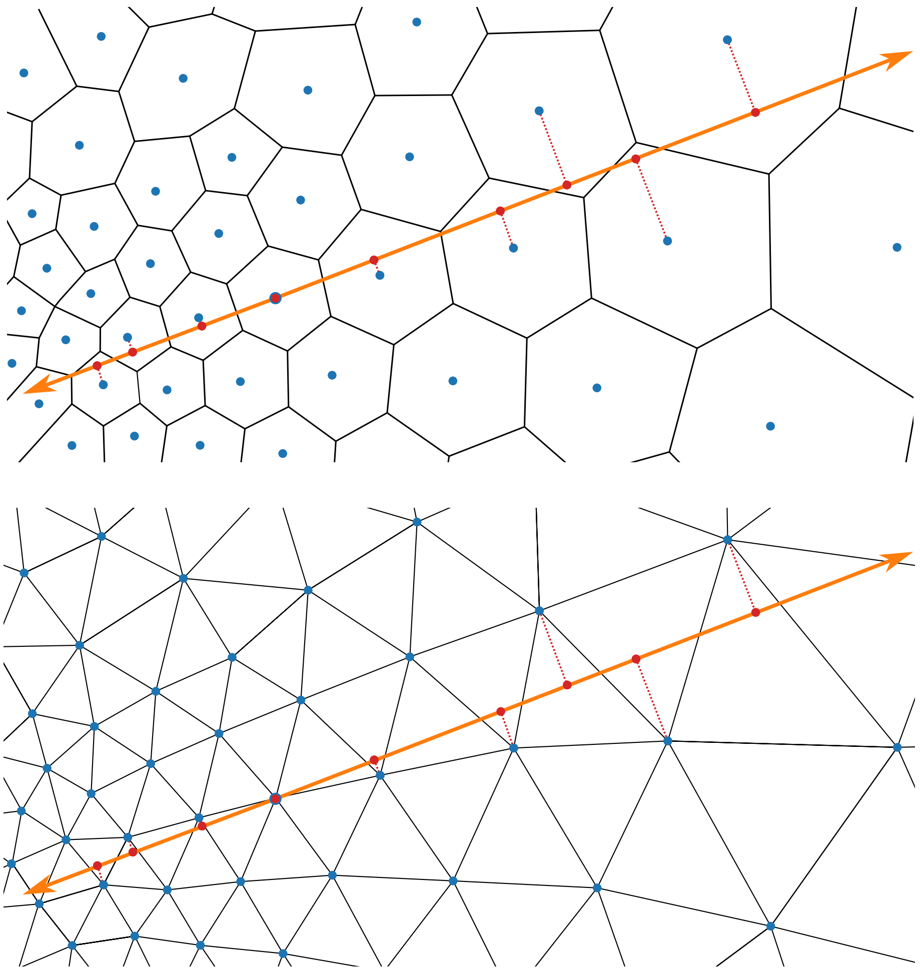

Figure 5.2: A ray traced through the Voronoi mesh *(top)* and its topologically dual Delaunay mesh *(bottom)*. Note that every distance increment along the ray is the projection of the edge of the Delaunay triangle connecting the traversed Voronoi cell centres. Hence, the lengths of the edges are a local upper limit for the step size encountered along the ray.

meshes, or their topologically dual Voronoi meshes, provide an excellent means to capture the complex morphologies typically encountered in radiative transfer simulations. Figure 5.2 shows a ray traced through a domain of Voronoi cells and the corresponding dual Delaunay tetrahedralisation. To use these meshes in Magritte, the Delaunay vertices (or Voronoi centres) can be used as the points and the nearest neighbours can be extracted from the edge list of the mesh, since every pair of nearest neighbours always shares an edge.

5.4.1 Meshing analytic models

Although many astrophysical objects are characterised by irregular structures that are difficult to describe with analytic models, it is nevertheless useful to study these analytic approximations, since they often make it easier to disentangle the effects of the various processes taking place. When discretising such a model, the key objective is to properly sample the functions that describe the model parameters. For simplicity we will restrict ourselves to one function, say $f(\boldsymbol{x})$, for which we want to optimise the mesh. We call this the tracer function and we assume it to be positive, i.e. $f(\boldsymbol{x}) \geq 0$. In radiative transfer computations the density distribution is often used for this purpose. Properly sampling a function for use in a differential equation solver means properly tracking its changes through the domain. A mesh that properly samples the tracer function will have small elements whenever the change in the tracer is large and vice versa. Therefore, in order to quantify the desired local element size distribution, we need to quantify the maximal local relative change in the tracer function. This is given by the norm of the gradient of its logarithm, which we will denote as,

$$Gf(\boldsymbol{x}) \ \equiv \ \max_{\hat{\boldsymbol{n}} \in S^2} \left\{ \hat{\boldsymbol{n}} \cdot \nabla f(\boldsymbol{x}) \ / \ f(\boldsymbol{x}) \right\} \ = \ \|\nabla \log f(\boldsymbol{x})\|. \tag{5.8}$$

This can be used in a map to obtain the desired element size distribution, $\ell(\boldsymbol{x})$, of the model mesh. We thus look for a continuous mapping that (at least roughly) maps $\max\{Gf(\boldsymbol{x})\}$ to $\min\{\ell(\boldsymbol{x})\}$, and maps $\min\{Gf(\boldsymbol{x})\}$ to $\max\{\ell(\boldsymbol{x})\}$.

Given the form of the radiative transfer equation one could argue that when constructing the mesh, one should strive to keep the optical depth increments as small as possible. However, this is not strictly required here, since Magritte will automatically limit the size of the optical depth increments by interpolating the optical properties where necessary. Furthermore, it was already shown, in the context of subdivision stopping criteria for adaptively refined meshes in [137], that it is far more important that a mesh allows us to accurately sample the model functions than to limit the encountered optical depth increments. Therefore, we use a linear mapping from $[\max\{Gf(\boldsymbol{x})\}, \min\{Gf(\boldsymbol{x})\}]$ to $[\min\{\ell(\boldsymbol{x})\}, \max\{\ell(\boldsymbol{x})\}]$. Any other mapping would have a larger local gradient in the desired local element size function, and would therefore make it harder to mesh.

Although GMSH has the option to construct meshes from an analytically defined element size distribution, it is often much simpler to provide the element size distribution evaluated on a background mesh. Therefore, we consider a regular Cartesian mesh that will be used as a background mesh. The resolution of the background mesh is determined by the smallest scales of the tracer function that we want to resolve, say ℓ_{min}. This is also the lower bound for the desired element size distribution. Similarly, we define an upper bound for the desired element size distribution, ℓ_{max}, which can be defined as a fraction of the size of the domain.

Once an appropriate background mesh is created with the desired element size distribution evaluated on it, GMSH can accordingly generate a mesh for a given domain. We demonstrate this method with an example in Section 5.5.1.

5.4.2 Re-meshing existing models

Since radiative transfer simulations are often only one of the many components in a larger simulation pipeline, the spatial discretisations that are used are often inherited from previous simulation steps. Therefore, the corresponding meshes are usually not tailored to the radiative transfer solvers, and, as a result, contain an exceedingly large number of elements. In this section, we present a simple heuristic algorithm to reduce the number elements in a given mesh, while preserving a proper sampling of a given tracer function or variable in the model, typically the density.

When re-meshing a given model, we need to know where the resolution of the original mesh is essential for the accurate representation of the model and where it (potentially) could be coarsened. This can again be quantified by the maximum relative change of a tracer function (or variable) at a point with respect to its neighbours, which can be expressed as an operator acting on the tracer function,

$$G f_i \ \equiv \ \max \left\{ \left| \frac{f_i - f_n}{f_i + f_n} \right|, \text{ for each neighbour } n \text{ of point } i \right\}. \tag{5.9}$$

By definition, $G f_i \in [0,1]$, so we can define a simple threshold value, G_{thres}, above

Table 5.1: Empirically determined parameters for the reduction algorithm.

Model type	f_{small}	f_{large}	G_{thres}
AMR	0.90	2.10	0.10
SPH	1.00	2.15	0.21

which the original local element size is deemed essential. We can now assign a desired local element size, $\ell(\boldsymbol{x})$, as a fraction of the original local element sizes, $L(\boldsymbol{x})$, where the fraction is determined by the local change in the tracer function,

$$\ell(\boldsymbol{x}) \ = \ L(\boldsymbol{x}) \begin{cases} f_{\text{small}} & \text{if } Gf(\boldsymbol{x}) \geq G_{\text{thres}} \\ f_{\text{large}} & \text{otherwise} \end{cases} \tag{5.10}$$

in which the local element sizes of the original mesh are given by,

$$L_i \ \equiv \ \text{mean}\Big\{ \|\boldsymbol{x}_i - \boldsymbol{x}_n\| \, , \text{ for each neighbour } n \text{ of point } i \Big\}. \tag{5.11}$$

The algorithm thus depends on three predefined parameters: f_{small}, f_{large}, and G_{thres}, for which the empirically determined values used for the applications in this thesis can be found in Table 5.1. Examples of this method are given in Section 5.5.2.

We should note that several variations are possible on the mapping to obtain the desired local element size distribution. However, due to the stochastic nature of the mesh construction process, the differences quickly blur, resulting in similar meshes. The particular map presented in (5.10) was chosen because it is a direct expression of our objective to coarsen the mesh where possible and retain the original mesh size where it is deemed essential for a proper representation of the model.

5.5 Applications

In this section we apply the mesh construction and reduction methods to a set of models inspired by analytic and numeric models of spiral-shaped stellar outflows (see also Chapter 4). These types of models are currently being developed to investigate the effect of companions on the shapes of the dust-driven winds of cool evolved stars, which have been found to deviate substantially from the originally assumed spherically symmetric wind model [113]. Understanding the origin of these features could help explain the morphological evolution towards the highly aspherical planetary nebula phase.

5.5.1 Meshing analytic models

As an example of an analytically defined model, we consider a stellar wind described by an Archimedean spiral with generic parameters, following the setup in [121]. Figure 5.3 shows two slices through the model, showing the density distribution as well as the underlying mesh generated with GMSH using the method described in Section 5.4.1. In this example, the density was used as a tracer function to determine the desired local element size distribution, with a minimum desired element size of $\ell_{\text{min}} = 15$ AU and a maximum desired element size of $\ell_{\text{max}} = 50$ AU. The regular Cartesian background mesh is defined in a cubic box of size $(1200\ \text{AU})^3$ and a resolution of 100^3 elements. The resulting mesh consists of 49 347 points, a modest number considering the relatively complex morphology. One can easily obtain even sparser meshes, either by increasing the minimum or maximum desired element sizes, or by applying the reduction method presented in Section 5.4.2. The latter technique will be demonstrated in the next section.

5.5.2 Re-meshing existing models

In the following, we consider four examples of how hydrodynamics models can be reduced, i.e. coarsened, before they are used as input in a radiative transfer solver. We consider two models based on an hierarchical octree mesh resulting from adaptive mesh refinement (AMR), and two models based on smoothed-particle hydrodynamics (SPH) simulation data.

5.5.2.1 AMR models

The basic idea of an octree discretisation of a 3D space is to locally subdivide an initial cubic cell into eight sub-cells until the desired local mesh size is achieved. Hydrodynamics solvers using adaptive mesh refinement often use an octree as the underlying geometric data structure. We consider late snapshots of two different hydrodynamics models of the intricate stellar outflow produced by a mass-losing

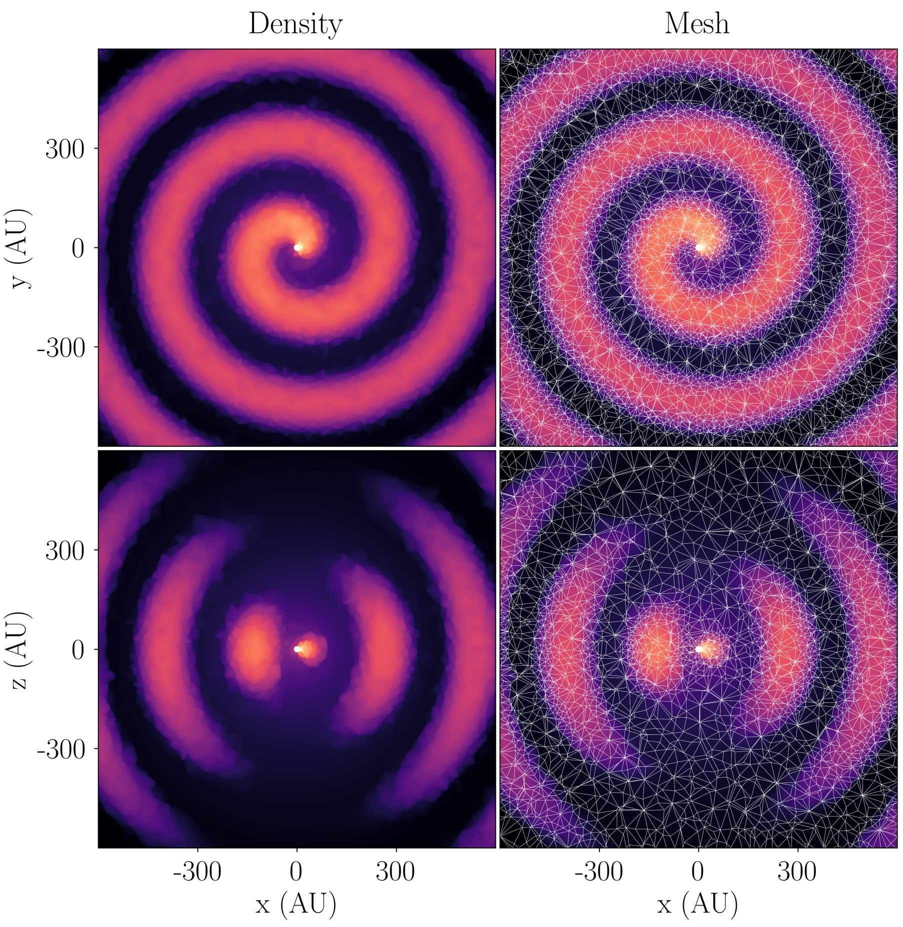

Figure 5.3: Model mesh for the analytic stellar wind model described by an Archimedean spiral. The top row shows slices through the centre along the xy-plane and the bottom row shows slices along the xz-plane.

asymptotic giant branch (AGB) star as it is perturbed by a companion, modelled using MPI-AMRVAC[7] [118]. We used the code with a Cartesian mesh and allowed for 8 levels of adaptive refinement. The hydrodynamic simulations were kindly provided by Jan Bolte. More details about the models and their astrophysical interpretation can be found in [119] (see also Chapter 4).

The first example, shown in Figure 5.4, contains a relatively regular spiral outflow. Reducing the model using the algorithm described in Section 5.4.2, the resulting reduced mesh contains about 10.2 times fewer points than the original one. This results in a speedup factor of 18.3 for the computation of the radiation field. In the second example, shown in Figure 5.5, we consider a more erratic spiral outflow.

[7]See also `amrvac.org`.

Table 5.2: Properties of the original and reduced meshes and the resulting speedup that is achieved in computing the radiation field.

Model	N_{original}	N_{reduced}	reduction	speedup
AMR regular	642 048	62 984	10.2	18.3
AMR erratic	627 712	75 137	8.4	12.6
SPH regular	916 601	82 554	11.1	35.0
SPH erratic	820 471	76 660	10.7	34.1

This leads to a reduced mesh containing 8.4 times fewer points than the original one, which results in a 12.6-fold speedup. The parameters of the reduction algorithm can be found in Table 5.1 and the properties of the original and reduced meshes are summarised in Table 5.2.

In both examples, one can see in the reduced meshes some artefacts of the levels of refinement in the original meshes. This is due to the fact that the desired element sizes in the algorithm are determined by the original element sizes.

To quantify the quality of the reduced meshes we compute the radiation field for both the original and reduced models using Magritte and calculate the absolute relative difference between the results. The solution on the reduced mesh can be mapped to the original by barycentric interpolation on the reduced mesh to each point in the original mesh. This can readily be done using the `LinearNDInteprolator` in Scipy [110]. The absolute relative difference between the results can then be computed by point-wise dividing the absolute difference by the result on the original mesh. In order to gauge the overall distribution of the errors, Figure 5.6 presents the cumulative density distribution of the relative errors. The more than 10% of points with an error below 10^{-3} in the AMR erratic model are due to the fact that the reduced mesh at small resolutions still closely resembles the original one.

Since the internal geometric data structure in Magritte consists of a point cloud with nearest neighbour information, the hierarchical octree mesh produced by MPI-AMRVAC cannot be used directly as input for Magritte. However, a natural way to map the octree mesh to a point cloud is to associate all cell data with the cell centre, then use the cell centres as points in Magritte, and extract the nearest neighbour information from the hierarchical octree.

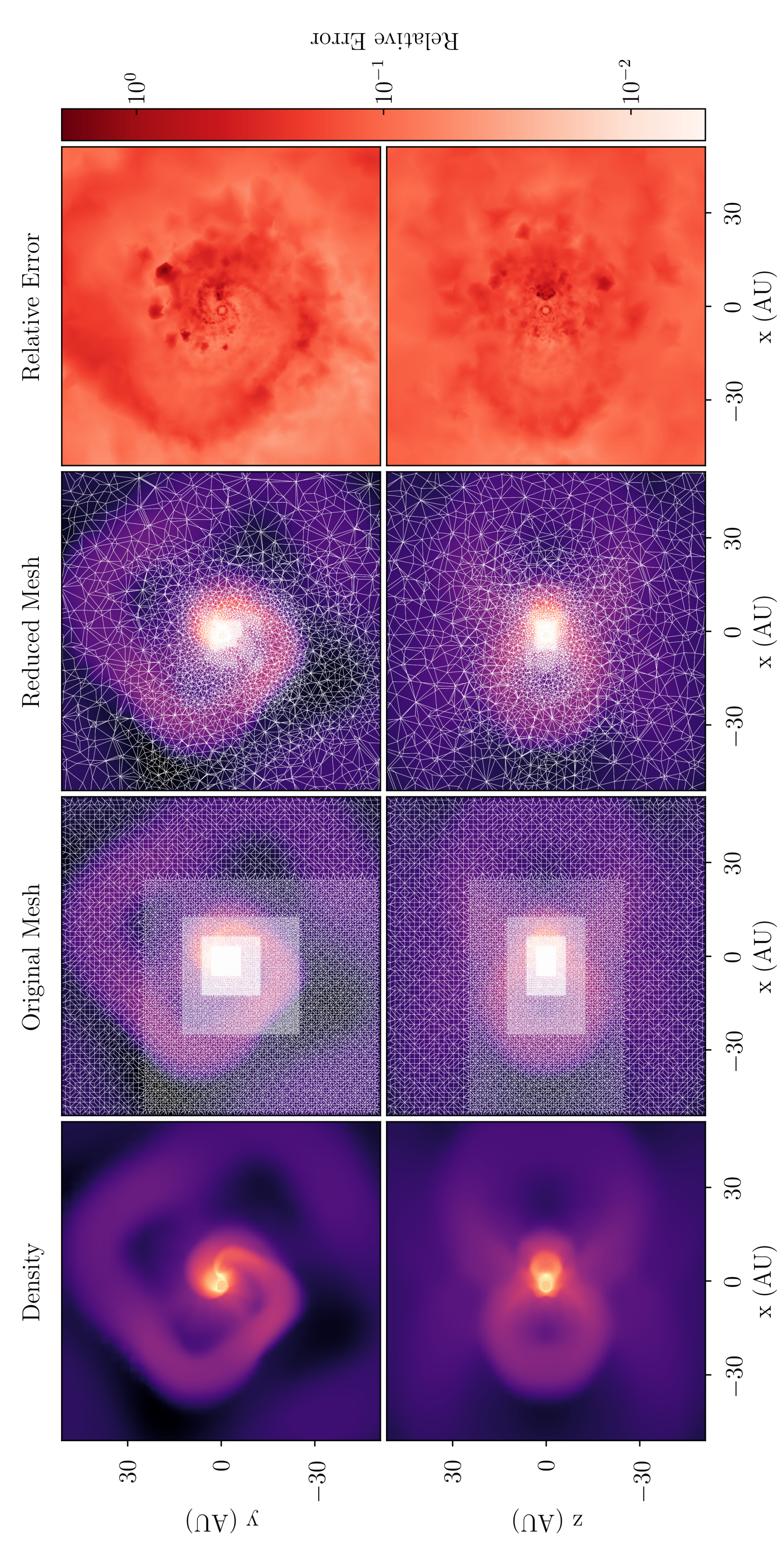

Figure 5.4: Comparison between the original and reduced model mesh for the octree version of the regular spiral model. The properties of the meshes can be found in Table 5.2. The top row shows slices through the centre along the xy-plane and the bottom row shows slices along the xz-plane. The relative error in the rightmost column is computed as the average over all directions and frequency bins of the absolute relative difference between the radiation field computed on the original mesh and the radiation field computed on the reduced mesh when interpolated to the original.

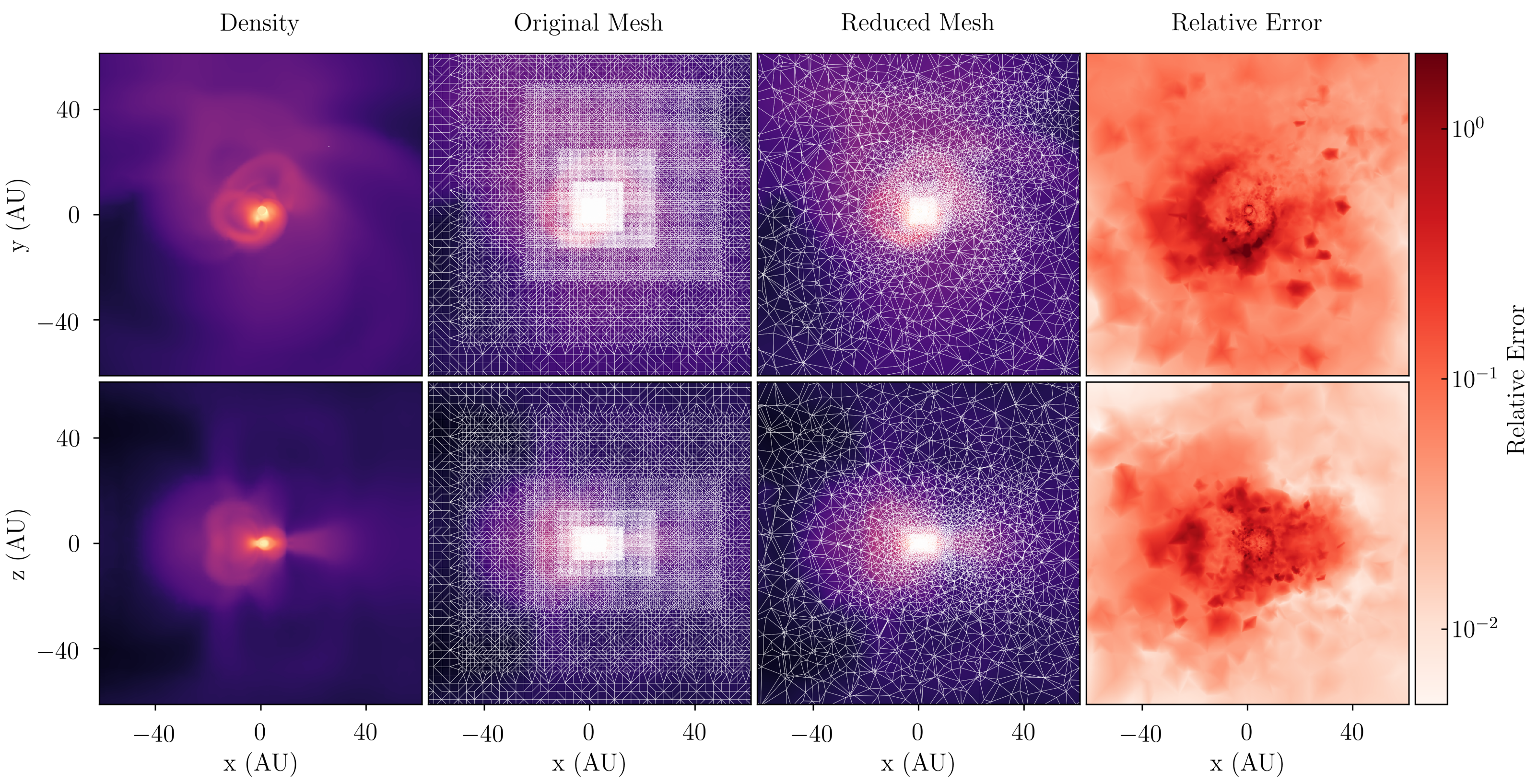

Figure 5.5: Comparison between the original and reduced model mesh for the octree version of the erratic spiral model. The properties of the meshes can be found in Table 5.2. The top row shows slices through the centre along the xy-plane and the bottom row shows slices along the xz-plane. The relative error in the rightmost column is computed as the average over all directions and frequency bins of the absolute relative difference between the radiation field computed on the original mesh and the radiation field computed on the reduced mesh when interpolated to the original.

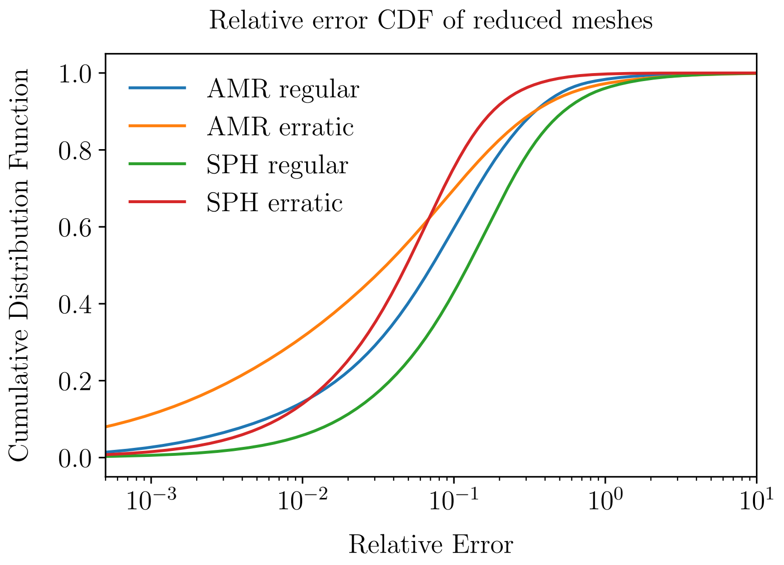

Figure 5.6: Cumulative distribution function of the relative errors with respect to the original meshes for the different models, computed for 1000 bins. The properties of the different meshes can be found in Table 5.2.

Figures 5.7 and 5.8 show a comparison between a dense regular directional discretisation containing $12 \times 2^8 = 3072$ rays and our adaptive scheme containing only 552 rays for a point half way along the z-axis looking down on the xy-plane of the AMR models. The adaptive scheme allows for 3 levels of refinement, from $l_{\text{min}} = 1$ up until $l_{\text{max}} = 4$. Although the results on the coarser adaptive discretisation clearly shows some differences with the finer regular one, the overall relative errors are limited, as can be seen from the cumulative distribution of the relative errors between the dense regular and adaptive models, shown in Figure 5.9. More than 60% of all the rays originating from all the points show a relative error below 10%, and more than 80% show a relative error below 20%. Note that about 20% of the points have a relative error below 10^{-3}. This is due to the fact that the maximally refined directions in the adaptive scheme have the same order, and hence exactly the same rays, as in the dense regular scheme, yielding exactly the same results.

5.5.2.2 SPH models

In smoothed-particle hydrodynamics (SPH) simulations, rather than defining the physical quantities on a mesh, the model is described by a number of particles with definite properties and smoothing kernels describing their proliferation. We again

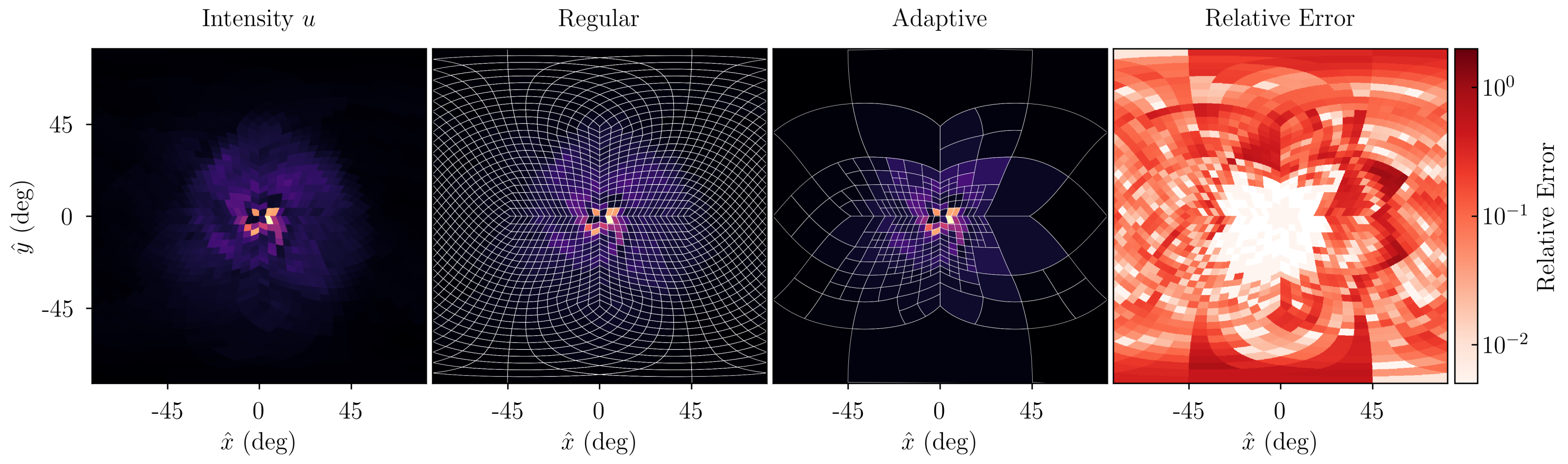

Figure 5.7: Comparison between a regular and an adaptive discretisation of the directions for a point located at $(x, y, z) = (0, 0, 42)$ AU in the regular spiral AMR model. The point and rotation are chosen such that the viewing angle resembles the slice in the top row of Figure 5.4. Since the mean intensity along a ray (u) is symmetric, each plot shows only half of a Cartesian projection.

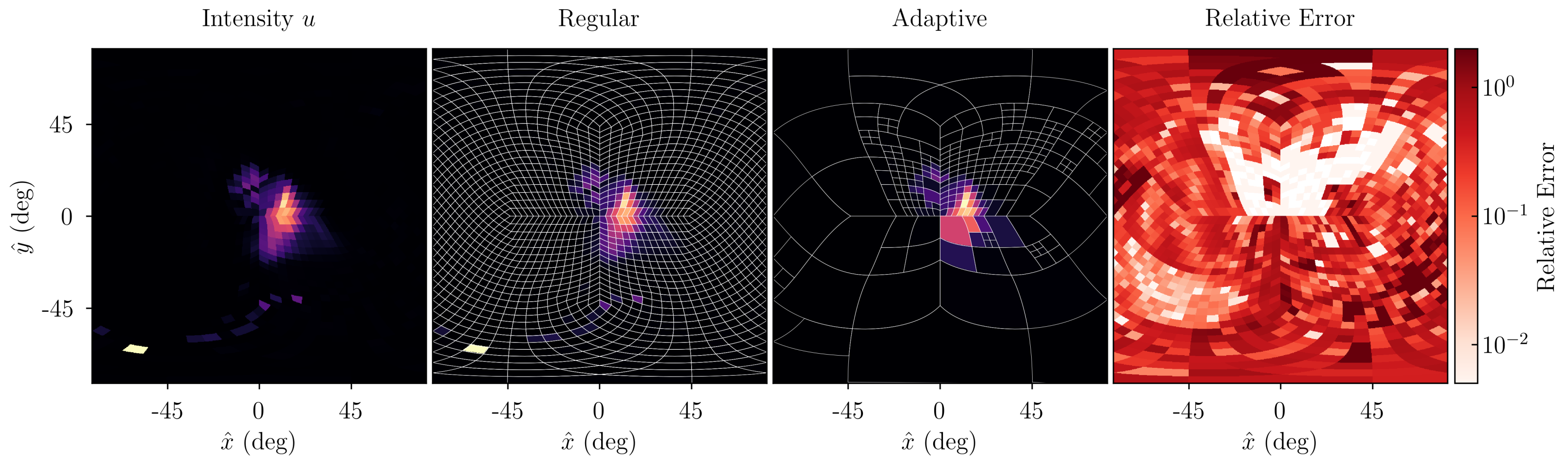

Figure 5.8: Comparison between a regular and an adaptive discretisation of the directions for a point located at $(x, y, z) = (0, 0, 6)$ AU in the erratic spiral AMR model. The point and rotation are chosen such that the viewing angle resembles the slice in the top row of Figure 5.5. Since the mean intensity along a ray (u) is symmetric, each plot shows only half of a Cartesian projection.

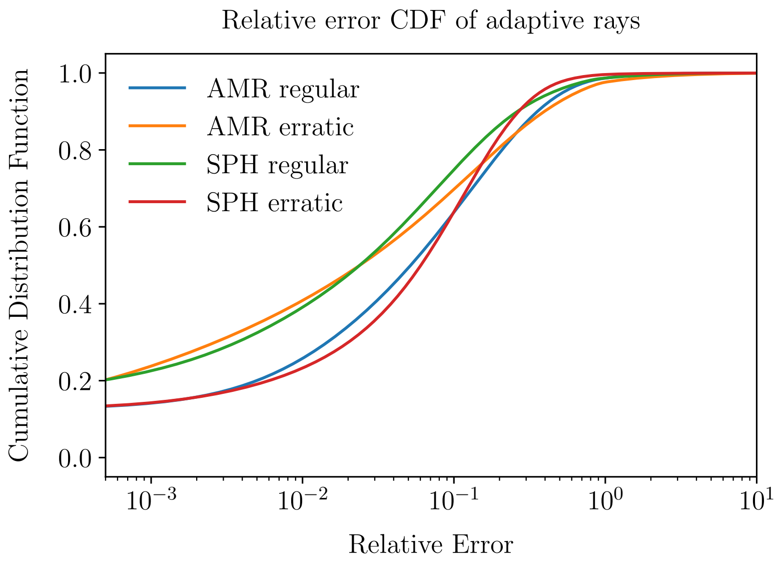

Figure 5.9: Cumulative distribution function of the relative errors of the adaptively ray-traced models with respect to the regular ones, computed for 1000 bins. The reduced meshes (see Table 5.2) were used as spatial discretisation.

consider late snapshots of two hydrodynamics models of the intricate stellar outflow produced by a mass-losing asymptotic giant branch (AGB) star as it is perturbed by a companion, this time modelled using the smoothed-particle hydrodynamics code PHANTOM[8] [120]. The hydrodynamics simulations were kindly provided by Silke Maes and Jolien Malfait. More details about these models and their astrophysical interpretation can be found in [115, 116]. These are the same models that we used to create synthetic observations of in Chapter 4.

The first example, shown in Figure 5.10, describes a very regular spiral outflow. Reducing the model using the algorithm described in Section 5.4.2, the resulting reduced mesh contains 11.1 times fewer points than the original. This results in a speedup factor of 35.0 for the computation of the radiation field. In the second example, shown in Figure 5.11, we consider a much more erratic spiral outflow. Despite the complex morphology, the reduced mesh still contains 10.7 times fewer points than the original, resulting in a 34.1-fold speedup. The parameters of the reduction algorithm can be found in Table 5.1 and the properties of the original and reduced meshes are summarised in Table 5.2.

[8]See also `phantomsph.bitbucket.io`.

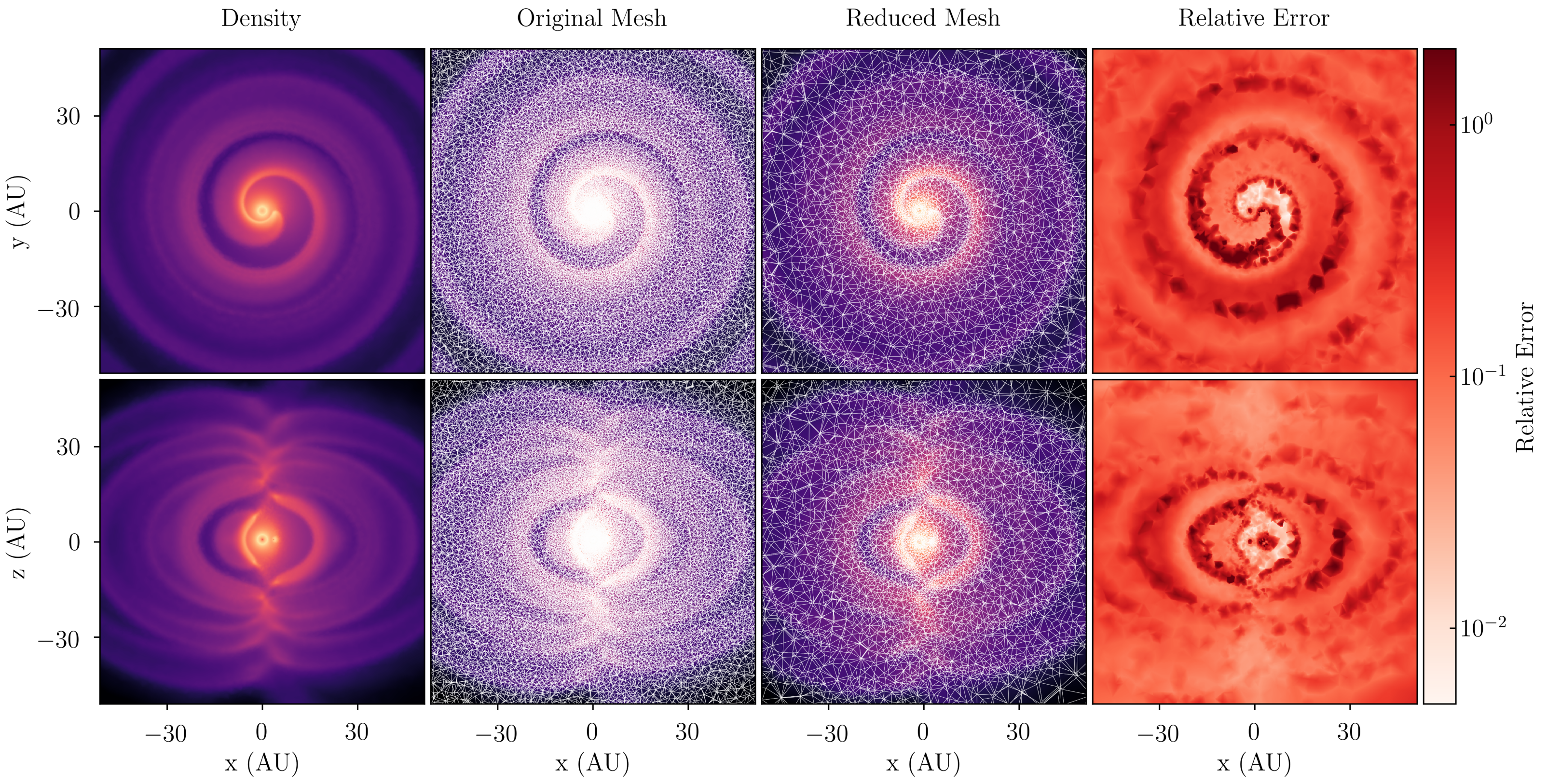

Figure 5.10: Comparison between the original and reduced model mesh for the SPH version of the regular spiral model. The properties of the meshes can be found in Table 5.2. The top row shows slices through the centre along the xy-plane and the bottom row shows slices along the xz-plane. The relative error in the rightmost column is computed as the average over all directions and frequency bins of the absolute relative difference between the radiation field computed on the original mesh and the radiation field computed on the reduced mesh when interpolated to the original.

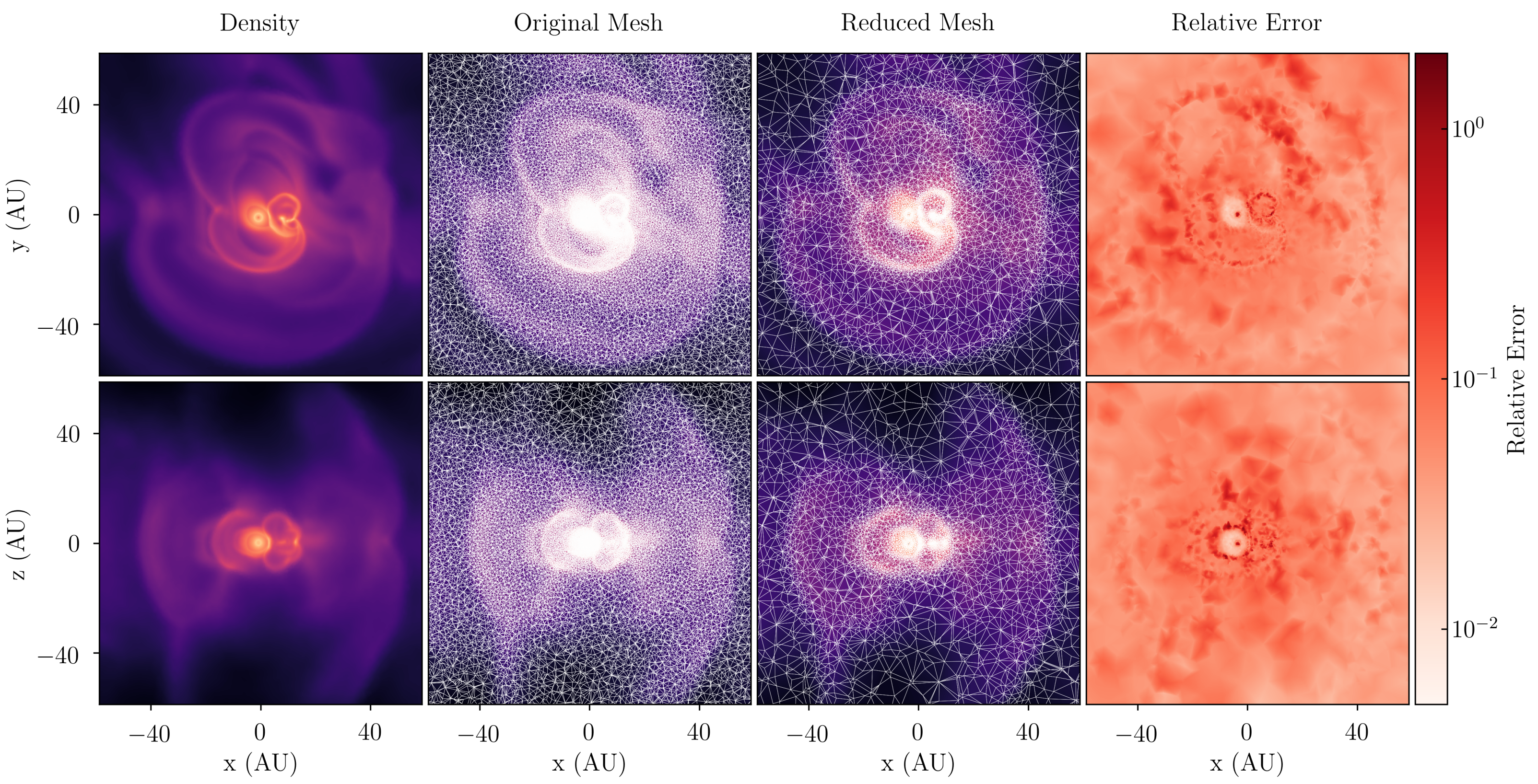

Figure 5.11: Comparison between the original and reduced model mesh for the SPH version of the erratic spiral model. The properties of the meshes can be found in Table 5.2. The top row shows slices through the centre along the xy-plane and the bottom row shows slices along the xz-plane. The relative error in the rightmost column is computed as the average over all directions and frequency bins of the absolute relative difference between the radiation field computed on the original mesh and the radiation field computed on the reduced mesh when interpolated to the original.

The quality of the meshes can again be quantified by comparing the results of a radiative transfer computation using Magritte between the original and reduced meshes. The results of the reduced mesh can be mapped to the original one in the same way as with the AMR models. Figure 5.6 shows the cumulative distribution of the relative errors between the original and reduced meshes. The SPH erratic model shows a relative error below 10% for about 90% of its points. This can be attributed to the fact that the original SPH model already had a higher sampling that follows the morphology more closely. Since the desired element sizes are based on this sampling, the resulting meshes are of a higher quality, explaining the observed lower relative errors.

The point cloud structure of an SPH data set maps naturally to the internal geometric data structure of Magritte, and thus can be used as such. However, it should be noted that in this way we do not account for the SPH smoothing kernels in the radiative transfer computations.

Figures 5.12 and 5.13 show a comparison between a dense regular directional discretisation containing $12 \times 2^8 = 3072$ rays and our adaptive scheme containing 552 rays for a point half way along the z-axis looking down on the xy-plane of the SPH models. The adaptive scheme allows for 3 levels of refinement from $l_{\text{min}} = 1$ up until $l_{\text{max}} = 4$. In accordance with the results for the AMR models, the results for the SPH models on the coarser adaptive discretisation clearly shows some differences with the finer regular one, while the overall relative errors are again limited, as can be seen from the cumulative distribution of the relative errors between the dense regular and adaptive models shown in Figure 5.9. More than 60% of all the rays originating from all the points show a relative error below 10%, and more than 80% show a relative error below 20%. Note also here that about 20% of the points have a relative error below 10^{-3}, which can be attributed to the fact that the maximally refined directions in the adaptive scheme have the same order and hence exactly the same rays as in the dense regular scheme, yielding again exactly the same results.

In this chapter, we demonstrated how typical input models for radiative transfer simulations can successfully be reduced to speed up computations by more than an

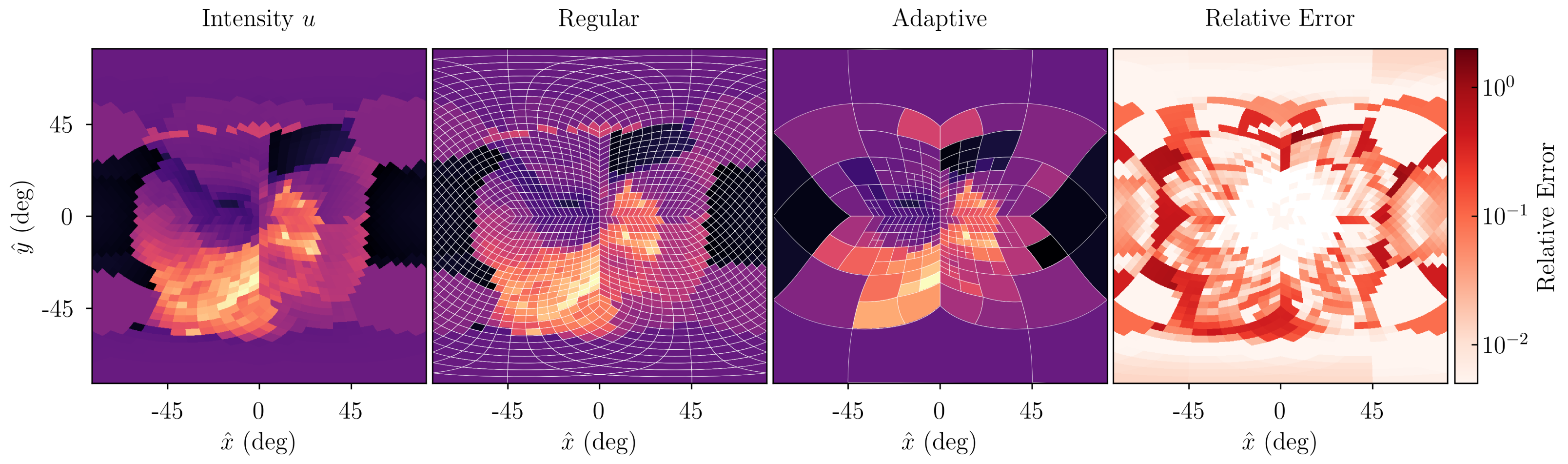

Figure 5.12: Comparison between a regular and an adaptive discretisation of the directions for a point located at $(x, y, z) = (0, 0, 81)$ AU in the regular spiral SPH model. The point and rotation are chosen such that the viewing angle resembles the slice in the top row of Figure 5.10. Since the mean intensity along a ray (u) is symmetric, each plot shows only half of a Cartesian projection.

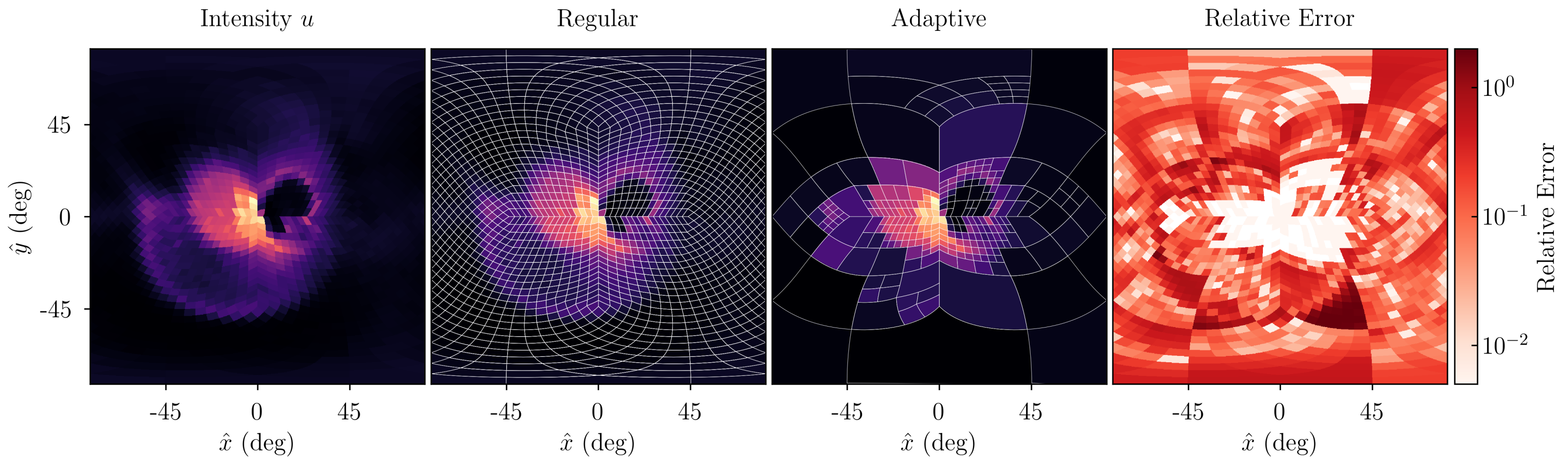

Figure 5.13: Comparison between a regular and an adaptive discretisation of the directions for a point located at $(x, y, z) = (0, 0, 18)$ AU in the erratic spiral SPH model. The point and rotation are chosen such that the viewing angle resembles the slice in the top row of Figure 5.11. Since the mean intensity along a ray (u) is symmetric, each plot shows only half of a Cartesian projection.

order of magnitude with only limited loss in accuracy. However, the methods are rather heuristic and we cannot provide any guarantees for their success. Therefore, in the next chapter, we present a probabilistic framework to put these methods on a stronger mathematical footing.

Chapter 6

Probabilistic solution methods

Some men went fishing in the sea with a net, and upon examining what they caught, they concluded that there was a minimum size to the fish in the sea.

– Arthur S. Eddington

6.1 Introduction

Throughout this thesis, we already discussed several methods to accelerate radiative transfer computations. Some of them should (at least in principle) not affect the accuracy of the results, such as e.g. acceleration of convergence, parallellisation, and GPU offloading, while others, such as e.g. the particular choice of solver and the mesh reduction methods involve a trade-off between computational speed and accuracy. In this chapter, we present a probabilistic mathematical framework to study and optimise this trade-off between computational speed and accuracy.

Considering the mesh reduction techniques presented Chapter 5, one can argue that they are wasteful, since they discard most of the data that is contained in the original model. One of the goals of this chapter will be to present a method in which all data can be taken into account, while still effectively reducing the size of the radiative transfer problem that needs to be solved. At the same time, this method will provide a measure for the uncertainty that is introduced by reducing the system, as a measure of the information lost due to compressing the model. Furthermore, we will attempt to relate this measure of uncertainty to the absolute uncertainty on radiative transfer computations, to become, unlike the fishermen in Eddington's parable, aware of the effects of our solution methods on our results.

6.2 Probabilistic methods

We present a probabilistic numerical approach to radiative transfer simulation by formulating the solution of the radiative transfer equation as a linear regression problem. We start at the very beginning to gradually introduce the concepts and ideas in our notation. For a more comprehensive introduction, see e.g. [146, 147].

6.2.1 Linear regression

The aim of a linear regression model is to approximate (or fit) a function, f, with a linear combination of basis functions, ϕ_i, based on data in the form of function evaluations, i.e. pairs of the form $(x_i, y_i \equiv f(x_i))$. Given a set of N_b basis functions, $\{\phi_i\}$, and a set of N_d data points, $\{(x_d, y_d)\}$, the approximation can either be expanded in terms of the basis function or in terms of the data, resulting respectively in a primal and dual formulation.

6.2.1.1 Primal formulation

In the primal formulation, the function approximation, $\tilde{f}(x)$, is modelled as a linear combination of the basis functions,

$$\tilde{f}(x) \;=\; \sum_{i=1}^{N_\mathrm{b}} w_i \, \phi_i(x) \;\equiv\; \boldsymbol{w}^\mathrm{T} \boldsymbol{\phi}(x), \tag{6.1}$$

where we defined the weight vector, $\boldsymbol{w}$, and basis function vector, $\boldsymbol{\phi}$. The appropriate weights, w_i, can be found, for instance, by minimising the regularised and weighted mean squared error between the model and the data,

$$\mathrm{RMSE}(\boldsymbol{w}) \;\equiv\; \sum_{d=1}^{N_\mathrm{d}} \frac{1}{\sigma_d^2} \left(\boldsymbol{w}^\mathrm{T} \boldsymbol{\phi}(x_d) - y_d \right)^2 + \sum_{i=1}^{N_\mathrm{b}} \left(\frac{w_i}{\lambda_i} \right)^2 . \tag{6.2}$$

The factors, σ_d^{-2}, weight the contributions of the different data points to the mean error, and are summarised in the diagonal matrix $\boldsymbol{\sigma} \equiv \mathrm{diag}(\sigma_d)$. We also added a regularisation term, characterised by the diagonal matrix $\boldsymbol{\lambda} \equiv \mathrm{diag}(\lambda_i)$, penalising the size of the components of the weight vectors, which will guarantee the existence of a unique solution, as we will see below. If we define the design matrix, $\Phi_{di} \equiv \phi_i(x_d)$, and the data vector pair, $(\boldsymbol{x},\boldsymbol{y})$, equation (6.2) can conveniently be rewritten as,

$$\mathrm{RMSE}(\boldsymbol{w}) \equiv \left(\boldsymbol{\sigma}^{-1}\left(\boldsymbol{\Phi}\boldsymbol{w}-\boldsymbol{y}\right)\right)^2 + \left(\boldsymbol{\lambda}^{-1}\boldsymbol{w}\right)^2, \tag{6.3}$$

in which the square of a vector, $\boldsymbol{a}$, is defined as $(\boldsymbol{a})^2 \equiv \boldsymbol{a}^{\mathrm{T}}\boldsymbol{a}$. Minimising this error function by demanding a vanishing gradient with respect to the weights, $\boldsymbol{w}$, yields,

$$\left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\sigma}^{-2}\boldsymbol{\Phi}+\boldsymbol{\lambda}^{-2}\right)\boldsymbol{w}_{\mathrm{min}} = \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\sigma}^{-2}\boldsymbol{y}, \tag{6.4}$$

such that the resulting function approximation reads,

$$\tilde{f}(x) = \boldsymbol{y}^{\mathrm{T}}\boldsymbol{\sigma}^{-2}\boldsymbol{\Phi}\left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\sigma}^{-2}\boldsymbol{\Phi}+\boldsymbol{\lambda}^{-2}\right)^{-1}\boldsymbol{\phi}(x). \tag{6.5}$$

The inverse is guaranteed to exist as long as the regularisation term is non-zero, i.e. $\lambda_i \neq 0, \forall i \in \{1,\ldots,N_{\mathrm{b}}\}$[1]. Note that a $(N_{\mathrm{b}}\times N_{\mathrm{b}})$-dimensional linear system must be solved to obtain the approximate solution, and hence the computational cost of the primal formulation is mainly determined by the number of basis functions, N_{b}.

6.2.1.2 Dual formulation

In the dual formulation, the function approximation, $\tilde{f}(x)$, is modelled as a linear combination of (evaluations of a kernel function on) the data,

$$\tilde{f}(x) = \sum_{d=1}^{N_{\mathrm{d}}} v_d\, k(x_d,x) \equiv \boldsymbol{v}^{\mathrm{T}}k(\boldsymbol{x},x), \tag{6.6}$$

in which the kernel is defined in terms of the basis functions,

$$k(x,x') \equiv \sum_{i=1}^{N_{\mathrm{b}}} \phi_i(x)\lambda_i^2\phi_i(x') = \boldsymbol{\phi}(x)^{\mathrm{T}}\boldsymbol{\lambda}^2\boldsymbol{\phi}(x'). \tag{6.7}$$

In case of an uncountably infinite set of basis functions this can readily be rewritten in an integral form. From this definition of the kernel, it can be seen that the weights of the primal and dual formulation are related as, $\boldsymbol{w} = \boldsymbol{\lambda}^2\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{v}$. Also here,

[1]This can be seen e.g. from the singular value decomposition of the matrix that is to be inverted.

the appropriate weights can be obtained by minimising the regularised and weighted mean squared error. In terms of the new weights, $\boldsymbol{v}$, this error reads,

$$\mathrm{RMSE}(\boldsymbol{v}) \equiv \left(\boldsymbol{\sigma}^{-1}\left(\boldsymbol{\Phi}\boldsymbol{\lambda}^2\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{v} - \boldsymbol{y}\right)\right)^2 + \left(\boldsymbol{\lambda}\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{v}\right)^2. \tag{6.8}$$

Minimising this error function by demanding a vanishing gradient with respect to the new weights, $\boldsymbol{v}$, then gives,

$$\left(\boldsymbol{\Phi}\boldsymbol{\lambda}^2\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\sigma}^{-2}\boldsymbol{\Phi}\boldsymbol{\lambda}^2\boldsymbol{\Phi}^{\mathrm{T}} + \boldsymbol{\Phi}\boldsymbol{\lambda}^2\boldsymbol{\Phi}^{\mathrm{T}}\right)\boldsymbol{v}_{\mathrm{min}} = \boldsymbol{\Phi}\boldsymbol{\lambda}^2\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\sigma}^{-2}\boldsymbol{y}. \tag{6.9}$$

Note that $\boldsymbol{\Phi}\boldsymbol{\lambda}^2\boldsymbol{\Phi}^{\mathrm{T}}$ might not be invertible and thus equation (6.9) might not have a unique solution. However, we can always pick the uniquely solvable system that will also minimise (6.8), by ignoring the extra factor, $\boldsymbol{\Phi}\boldsymbol{\lambda}^2\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\sigma}^{-2}$, yielding,

$$\left(\boldsymbol{\Phi}\boldsymbol{\lambda}^2\boldsymbol{\Phi}^{\mathrm{T}} + \boldsymbol{\sigma}^2\right)\boldsymbol{v}_{\mathrm{min}} = \boldsymbol{y}. \tag{6.10}$$

The resulting function approximation then reads,

$$\tilde{f}(x) = \boldsymbol{y}^{\mathrm{T}}\left(\boldsymbol{\Phi}\boldsymbol{\lambda}^2\boldsymbol{\Phi}^{\mathrm{T}} + \boldsymbol{\sigma}^2\right)^{-1} k(\boldsymbol{x}, x). \tag{6.11}$$

Note that the inverse is guaranteed to exist as long as $\sigma_i \neq 0, \forall i \in \{1, \ldots, N_{\mathrm{d}}\}$. In this case, a $(N_{\mathrm{d}} \times N_{\mathrm{d}})$-dimensional linear system must be solved to obtain the approximate solution, and thus, in contrast to the primal formulation, the computational cost of the dual formulation is determined mainly by the number of data points, N_{d}.

It should be noted that the dual formulation can also be constructed directly from a given kernel without any link to a set of basis functions. In particular, since the design matrix always appears in the combination, $\boldsymbol{\Phi}\boldsymbol{\lambda}^2\boldsymbol{\Phi}^{\mathrm{T}} = k(\boldsymbol{x}, \boldsymbol{x})$, it can always be replaced by its equivalent kernel expression.

6.2.1.3 Primal versus dual formulation

One can show that the primal (6.5) and dual (6.11) solutions are equal. This requires,

$$\boldsymbol{\sigma}^{-2}\boldsymbol{\Phi}\left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\sigma}^{-2}\boldsymbol{\Phi} + \boldsymbol{\lambda}^{-2}\right)^{-1} = \left(\boldsymbol{\Phi}\boldsymbol{\lambda}^2\boldsymbol{\Phi}^{\mathrm{T}} + \boldsymbol{\sigma}^2\right)^{-1}\boldsymbol{\Phi}\boldsymbol{\lambda}^2, \tag{6.12}$$

as proved in Appendix B.1. The only, yet key, difference between both formulations is thus the size of the linear system that needs to be solved to obtain the solution. In particular, if there are more data points than basis functions, then the primal formulation will result in the smallest linear system, whereas if there are more basis functions than data points, then the dual formulation is preferred.

6.2.1.4 Solving linear PDEs as linear regression

Numerically solving linear operator equations, such as, for instance, linear partial differential equations (PDEs), can be viewed as a linear regression problem. Say we want to numerically solve a PDE,

$$\begin{aligned} \mathscr{L} f(x) &= g(x), \ \ x \in D \\ \mathscr{B} f(x) &= h(x), \ \ x \in \partial D \end{aligned} \tag{6.13}$$

on a domain, D, with boundary, ∂D, where the PDE and boundary conditions are determined respectively by the linear operators $\mathscr{L}$ and $\mathscr{B}$. Suppose that for the numerical solution the domain is discretised to $\tilde{D}$, and that $\boldsymbol{a}$ is a vector containing the points in $\tilde{D}$. Furthermore, suppose that the boundary is discretised to $\partial\tilde{D}$, and that $\boldsymbol{b}$ is a vector containing the points in $\partial\tilde{D}$. The data can then be split as,

$$(\boldsymbol{x}, \boldsymbol{y}) = \left(\begin{pmatrix} \boldsymbol{a} \\ \boldsymbol{b} \end{pmatrix}, \begin{pmatrix} g(\boldsymbol{a}) \\ h(\boldsymbol{b}) \end{pmatrix} \right) \equiv \left(\begin{pmatrix} \boldsymbol{a} \\ \boldsymbol{b} \end{pmatrix}, \begin{pmatrix} \boldsymbol{g} \\ \boldsymbol{h} \end{pmatrix} \right). \tag{6.14}$$

Similarly, the design matrix, $\boldsymbol{\Phi}$, which has evaluations at different data points on its rows, can correspondingly be split as,

$$\boldsymbol{\Phi} = \begin{pmatrix} \mathscr{L}\boldsymbol{\phi}(\boldsymbol{a}) \\ \mathscr{B}\boldsymbol{\phi}(\boldsymbol{b}) \end{pmatrix} \equiv \begin{pmatrix} \boldsymbol{\Phi}_{\mathrm{L}} \\ \boldsymbol{\Phi}_{\mathrm{B}} \end{pmatrix}. \tag{6.15}$$

In this new notation, the key matrices appearing in the primal and dual formulation can conveniently be rewritten. In the primal formulation this yields,

$$\boldsymbol{\Phi}^{\mathrm{T}} \sigma^{-2} \boldsymbol{\Phi} + \lambda^{-2} = \boldsymbol{\Phi}_{\mathrm{L}}^{\mathrm{T}} \sigma^{-2} \boldsymbol{\Phi}_{\mathrm{L}} + \boldsymbol{\Phi}_{\mathrm{B}}^{\mathrm{T}} \sigma^{-2} \boldsymbol{\Phi}_{\mathrm{B}} + \lambda^{-2}, \tag{6.16}$$

while in the dual formulation this results in,

$$\boldsymbol{\Phi}\lambda^2\boldsymbol{\Phi}^{\mathrm{T}} + \sigma^2 = \begin{pmatrix} \boldsymbol{\Phi}_{\mathrm{L}}\lambda^2\boldsymbol{\Phi}_{\mathrm{L}}^{\mathrm{T}} + \sigma^2 & \boldsymbol{\Phi}_{\mathrm{L}}\lambda^2\boldsymbol{\Phi}_{\mathrm{B}}^{\mathrm{T}} \\ \boldsymbol{\Phi}_{\mathrm{B}}\lambda^2\boldsymbol{\Phi}_{\mathrm{L}}^{\mathrm{T}} & \boldsymbol{\Phi}_{\mathrm{B}}\lambda^2\boldsymbol{\Phi}_{\mathrm{B}}^{\mathrm{T}} + \sigma^2 \end{pmatrix}. \tag{6.17}$$

Equivalently, in terms of the kernel, equation (6.17) can be written as,

$$k(\boldsymbol{x},\boldsymbol{x}) + \sigma^2 = \begin{pmatrix} \mathscr{L}_1\mathscr{L}_2 k(\boldsymbol{a},\boldsymbol{a}) + \sigma_{\mathrm{L}}^2 & \mathscr{L}_1\mathscr{B}_2 k(\boldsymbol{a},\boldsymbol{b}) \\ \mathscr{B}_1\mathscr{L}_2 k(\boldsymbol{b},\boldsymbol{a}) & \mathscr{B}_1\mathscr{B}_2 k(\boldsymbol{b},\boldsymbol{b}) + \sigma_{\mathrm{B}}^2 \end{pmatrix}, \tag{6.18}$$

which elegantly separates the bulk, the boundary, and their cross-correlations.

As with linear regression, numerically solving the PDE can now be formulated as a minimisation problem and can be solved both in the primal and dual formulation. This is in close analogy with the Galerkin or finite element method, see e.g. [148].

6.2.2 Bayesian linear regression

The framework of linear regression can also be derived in a Bayesian or probabilistic setting. Here we consider a stochastic function, $F(x)$, and are interested in the distribution of this function as it is conditioned on observations of evaluations, $(\boldsymbol{x},\boldsymbol{y})$, of that function, i.e. our goal is to find $p(F(x)\,|\,\boldsymbol{y})$. To simplify notation, we write, $|\boldsymbol{y}$, to denote conditioning on the data, whereas we actually mean, $|(\boldsymbol{x},\boldsymbol{y})$.

6.2.2.1 Bayesian primal formulation

Given a linear model in the primal formulation with corresponding weights, $\boldsymbol{w}$, a zero-mean Gaussian error on the observed function evaluations, denoted by the stochastic variable, $\boldsymbol{\mathcal{Y}}$, results in a Gaussian likelihood given by,

$$p\left(\boldsymbol{\mathcal{Y}}\,|\,\boldsymbol{w}\right) = \mathcal{N}\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x}), \sigma^2\right) = \mathcal{N}\left(\boldsymbol{\Phi}\boldsymbol{w}, \sigma^2\right). \tag{6.19}$$

Note that we reused the variable σ^2 and reinterpreted it as the variance on the data, allowing for deviations from the mean value, $\boldsymbol{\Phi}\boldsymbol{w}$, predicted by the model, given the weights, $\boldsymbol{w}$. We will see below that both interpretations are indeed compatible.

Furthermore, we assume a zero-mean Gaussian prior on the stochastic weights, $\mathcal{W}$,

$$p(\mathcal{W}) = \mathcal{N}\left(\mathbf{0}, \lambda^2\right). \tag{6.20}$$

Note that we reused the variable λ^2 and reinterpreted it as the variance of the prior on the weights. Using the relations given in Appendix B.2, we can infer the implied distribution of the weights conditioned on the data,

$$p(\mathcal{W} | \boldsymbol{y}) = \mathcal{N}\left(\boldsymbol{\mu}_{w|y}, \boldsymbol{\Sigma}_{w|y}\right), \tag{6.21}$$

in which the mean vector, $\boldsymbol{\mu}_{w|y}$, and covariance matrix, $\boldsymbol{\Sigma}_{w|y}$, are defined as,

$$\boldsymbol{\mu}_{w|y} \equiv \boldsymbol{\Sigma}_{w|y} \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\sigma}^{-2} \boldsymbol{y}, \tag{6.22}$$

$$\boldsymbol{\Sigma}_{w|y} \equiv \left(\boldsymbol{\lambda}^{-2} + \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\sigma}^{-2} \boldsymbol{\Phi}\right)^{-1}. \tag{6.23}$$

Since the stochastic function, $F(x)$, is a linear mapping of the weights, that is, $F(x) = \mathcal{W}^{\mathrm{T}} \boldsymbol{\phi}(x)$, the corresponding conditioned distribution reads,

$$p\big(F(x) | \boldsymbol{y}\big) = \mathcal{N}\left(\mu_{\text{primal}}(x),\ \sigma^2_{\text{primal}}(x)\right), \tag{6.24}$$

in which the mean, $\mu_{\text{primal}}(x)$, and variance, $\sigma^2_{\text{primal}}(x)$, are defined as,

$$\mu_{\text{primal}}(x) \equiv \boldsymbol{\mu}^{\mathrm{T}}_{w|y} \boldsymbol{\phi}(x), \tag{6.25}$$

$$\sigma^2_{\text{primal}}(x) \equiv \boldsymbol{\phi}(x)^{\mathrm{T}} \boldsymbol{\Sigma}_{w|y} \boldsymbol{\phi}(x). \tag{6.26}$$

Substituting equation (6.22), we rediscover the primal solution (6.5) as the mean of the resulting conditioned primal distribution,

$$\mu_{\text{primal}}(x) = \boldsymbol{y}^{\mathrm{T}} \boldsymbol{\sigma}^{-2} \boldsymbol{\Phi} \left(\boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\sigma}^{-2} \boldsymbol{\Phi} + \boldsymbol{\lambda}^{-2}\right)^{-1} \boldsymbol{\phi}(x). \tag{6.27}$$

Furthermore, we now also have a measure for the spread in possible approximations, which can be derived from the variance of the conditioned distribution,

$$\sigma^2_{\text{primal}}(x) = \boldsymbol{\phi}(x)^{\mathrm{T}} \left(\boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\sigma}^{-2} \boldsymbol{\Phi} + \boldsymbol{\lambda}^{-2}\right)^{-1} \boldsymbol{\phi}(x). \tag{6.28}$$

This allows us to predict an approximation for the function, f, which we aim to fit, based on the data, $(\boldsymbol{x}, \boldsymbol{y})$, and moreover provide a confidence level for the result.

6.2.2.2 Bayesian dual formulation

A similar argument can be made for the dual formulation and is typically encountered in the context of Gaussian processes, see e.g. [147]. Since we assumed both the weights and errors in the data to follow a (multivariate) Gaussian distribution, we know that the function values and the data follow a joint Gaussian distribution,

$$p\left(\begin{bmatrix} F(x) \\ \boldsymbol{y} \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} 0 \\ \boldsymbol{0} \end{bmatrix}, \begin{bmatrix} k(x,x) & k(x,\boldsymbol{x}) \\ k(\boldsymbol{x},x) & k(\boldsymbol{x},\boldsymbol{x}) + \boldsymbol{\sigma}^2 \end{bmatrix}\right), \tag{6.29}$$

where we reused the definition of the kernel (6.7). Using the relations given in Appendix B.3, the posterior distribution can be obtained by conditioning on the data, $(\boldsymbol{x}, \boldsymbol{y})$, yielding,

$$p\big(F(x)\,|\,\boldsymbol{y}\big) = \mathcal{N}\left(\mu_{\text{dual}}(x),\ \sigma^2_{\text{dual}}(x)\right), \tag{6.30}$$

in which the mean, $\mu_{\text{dual}}(x)$, and variance, $\sigma^2_{\text{dual}}(x)$, are respectively defined as,

$$\mu_{\text{dual}}(x) \equiv k(x,\boldsymbol{x}) \left(k(\boldsymbol{x},\boldsymbol{x}) + \boldsymbol{\sigma}^2\right)^{-1} \boldsymbol{y}, \tag{6.31}$$

$$\sigma^2_{\text{dual}}(x) \equiv k(x,x) - k(x,\boldsymbol{x}) \left(k(\boldsymbol{x},\boldsymbol{x}) + \boldsymbol{\sigma}^2\right)^{-1} k(\boldsymbol{x},x). \tag{6.32}$$

Rewriting this in terms of the design matrix, we rediscover the dual solution (6.11) as expectation value of the conditioned distribution,

$$\mu_{\text{dual}}(x) = \boldsymbol{\phi}(x)^{\mathrm{T}} \boldsymbol{\lambda}^2 \boldsymbol{\Phi}^{\mathrm{T}} \left(\boldsymbol{\Phi} \boldsymbol{\lambda}^2 \boldsymbol{\Phi}^{\mathrm{T}} + \boldsymbol{\sigma}^2\right)^{-1} \boldsymbol{y}. \tag{6.33}$$

Similarly, we can also obtain a measure for the spread in possible approximations, which can be derived from the variance of the conditioned distribution,

$$\sigma^2_{\text{dual}}(x) = \boldsymbol{\phi}(x)^{\mathrm{T}}\left(\lambda^2 - \lambda^2\boldsymbol{\Phi}^{\mathrm{T}}\left(\boldsymbol{\Phi}\lambda^2\boldsymbol{\Phi}^{\mathrm{T}} + \sigma^2\right)^{-1}\boldsymbol{\Phi}\lambda^2\right)\boldsymbol{\phi}(x). \tag{6.34}$$

Again, we can predict an approximation for the function, f, which we aim to fit, based on the data, $(\boldsymbol{x}, \boldsymbol{y})$, and moreover provide a confidence level for the result.

6.2.2.3 Bayesian primal versus Bayesian dual formulation

As in the non-probabilistic case, we notice that in the primal formulation a $(N_{\mathrm{b}} \times N_{\mathrm{b}})$-dimensional linear system needs to be solved, while in the dual formulation it is a $(N_{\mathrm{d}} \times N_{\mathrm{d}})$-dimensional linear system, both to for the mean and for the variance.

We already showed in the non-probabilistic case (see Appendix B.1) that the primal and dual solutions are equal. Using the Woodburry matrix identity, we can now also verify that the variances in the primal and dual formulation are equal, since,

$$\left(\lambda^{-2} + \boldsymbol{\Phi}^{\mathrm{T}}\sigma^{-2}\boldsymbol{\Phi}\right)^{-1} = \lambda^2 - \lambda^2\boldsymbol{\Phi}^{\mathrm{T}}\left(\boldsymbol{\Phi}\lambda^2\boldsymbol{\Phi}^{\mathrm{T}} + \sigma^2\right)^{-1}\boldsymbol{\Phi}\lambda^2. \tag{6.35}$$

Therefore, we can conclude that the duality also holds in the probabilistic sense, i.e.,

$$\mathcal{N}\left(\mu_{\text{primal}}(x), \sigma^2_{\text{primal}}(x)\right) = \mathcal{N}\left(\mu_{\text{dual}}(x), \sigma^2_{\text{dual}}(x)\right), \tag{6.36}$$

and thus that both formulations are equivalent and can be used interchangeably.

It should be noted that our choice of Gaussian priors in this description was only motivated by computational convenience, and that it is not ideal. For instance, the Gaussian distribution always assigns a non-zero probability, also to negative values of a variable. For many physical quantities that are only positive, such as density or temperature, this is not desirable as it can lead to non-physical results. However, bearing in mind these dangers, the Gaussian distribution is a good first approximation for the uncertainties in our variables. For a similar treatment using student-t distributed priors, see e.g. [149].

6.2.3 Uncertainty quantification

Quantifying uncertainties is itself an approximate endeavour. After all, the exact solution, f, is required in order to determine the error, ε, that is made in an approximation, $\tilde{f}$, since,

$$f(x) = \tilde{f}(x) + \varepsilon(x). \tag{6.37}$$

Although it is possible to obtain highly accurate estimates for the errors in particular models (see e.g. [150]), it is crucial to note that any form of practical on-the-fly uncertainty quantification will always only be an approximation of the true error. Just as the quality of the approximation strongly depends on the estimation method, so does the quality of the error.

6.2.3.1 Probabilistic numerical paradigm

Following the probabilistic numerical paradigm [151, 152], we aim to quantify uncertainties by modelling the distribution over all possible functions conditioned on the data. In particular, we will use the expectation of the conditioned distribution as our function approximation,

$$\tilde{f}(x) \equiv \mathbb{E}\left[F(x) \,|\, \boldsymbol{y}\right]. \tag{6.38}$$

As a result, we can estimate the expected squared error in this approximation with the variance of the conditioned distribution,

$$\tilde{\varepsilon}^2(x) \equiv \mathbb{V}\left[F(x) \,|\, \boldsymbol{y}\right], \tag{6.39}$$

where we used that fact that the stochastic function, $F(x)$, and the corresponding stochastic error, $\mathcal{E}(x)$, ought to be related as,

$$F(x) = \tilde{f}(x) + \mathcal{E}(x), \tag{6.40}$$

and the definition of the variance, which implies that,

$$\mathbb{V}\left[F(x) \,|\, \boldsymbol{y}\right] = \mathbb{E}\left[\left(F(x) - \tilde{f}(x)\right)^2 |\, \boldsymbol{y}\right] = \mathbb{E}\left[\mathcal{E}(x)^2 \,|\, \boldsymbol{y}\right]. \tag{6.41}$$

Assuming that the probabilistic model, $F(x)\,|\,\boldsymbol{y}$, is an adequate model for the actual function, $f(x)$, the variance thus quantifies the expected squared error in the function approximation. Note that in our particular case, where the posterior is a Gaussian, the variance does not depend on the function values, $\boldsymbol{y}$, in the data but only on the locations, $\boldsymbol{x}$, at which the function was evaluated.

Based on the variance in the dual formulation (6.32), one can derive an upper and lower bound on the squared error,

$$0 \;\leq\; \tilde{\varepsilon}^2(x) \;\leq\; k(x,x), \tag{6.42}$$

where, in the left inequality, we used the fact that the variance has to be positive, and in the right inequality that $k(\boldsymbol{x},\boldsymbol{x}) + \boldsymbol{\sigma}^2$ is a positive definite matrix, such that the second term in (6.32) is negative. It might seem odd to have an error measure that is bounded from above. However, one should note that it is not an upper bound on the actual error, but only an upper bound on the variance of the distribution of possible solutions. Furthermore, we should note that the upper bound can always be scaled using the prior covariance, $\boldsymbol{\lambda}$.

This probabilistic interpretation of numerical methods has a long and rich history (see e.g. [153]) that can be traced all the way back to 1896 with the work of Poincaré [154]. With some noteworthy exceptions in the eighties (see e.g. [155]), these ideas only rather recently gained more traction, see e.g. [151, 152, 156–158]. Our goal will be to tailor these ideas to radiative transfer simulations and apply them in a practical setting. First, however, we need to properly understand the approximate error measure (6.39), and identify what exactly it measures.

6.2.3.2 The limit of perfect data: $\boldsymbol{\sigma} \to \boldsymbol{0}$

In order to isolate the effects of the numerical scheme from the errors in the data, consider the limit of perfect data, i.e. $\boldsymbol{\sigma} \to \boldsymbol{0}$. The primal and dual solutions in this limit are respectively given by,

$$\begin{aligned}\mu_{\text{primal}}(x) &\rightarrow \boldsymbol{y}^{\mathrm{T}}\boldsymbol{\Phi}\left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\phi}(x), &(6.43)\\ \mu_{\text{dual}}(x) &\rightarrow \boldsymbol{y}^{\mathrm{T}}\left(\boldsymbol{\Phi}\lambda^2\boldsymbol{\Phi}^{\mathrm{T}}\right)^{-1}\boldsymbol{\Phi}\lambda^2\boldsymbol{\phi}(x). &(6.44)\end{aligned}$$

Note, however, that the inverses above do not necessarily exist. In particular, if $N_{\mathrm{d}} > N_{\mathrm{b}}$, the singular value decomposition shows that $\boldsymbol{\Phi}\lambda^2\boldsymbol{\Phi}^{\mathrm{T}}$ must be singular and thus only the primal formulation remains, whereas if $N_{\mathrm{d}} < N_{\mathrm{b}}$, it follows that $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ must be singular and thus only the dual formulation remains[2]. As a result, in the limit of perfect data, $\boldsymbol{\sigma} \rightarrow \mathbf{0}$, if $N_{\mathrm{d}} \neq N_{\mathrm{b}}$, the duality between the two formulations ceases to exist and only the formulation with the smallest corresponding linear system will have a unique solution.

Moreover, note that in this limit the variance of the primal formulation always vanishes, such that there is no probabilistic interpretation when $N_{\mathrm{d}} > N_{\mathrm{b}}$, i.e. when only the primal formulation remains. In the dual formulation, the variance in this limit is given by,

$$\sigma^2_{\text{dual}}(x) = \boldsymbol{\phi}(x)^{\mathrm{T}}\left(\lambda^2 - \lambda^2\boldsymbol{\Phi}^{\mathrm{T}}\left(\boldsymbol{\Phi}\lambda^2\boldsymbol{\Phi}^{\mathrm{T}}\right)^{-1}\boldsymbol{\Phi}\lambda^2\right)\boldsymbol{\phi}(x), \tag{6.45}$$

which evidently only makes sense if the inverse of $\boldsymbol{\Phi}\lambda^2\boldsymbol{\Phi}^{\mathrm{T}}$ exists, which requires that $N_{\mathrm{d}} \leq N_{\mathrm{b}}$. In the particular case that $N_{\mathrm{d}} = N_{\mathrm{b}}$, demanding that $\boldsymbol{\Phi}\lambda^2\boldsymbol{\Phi}^{\mathrm{T}}$ is invertible implies that $\boldsymbol{\Phi}$ is invertible, such that also in this case the variance vanishes. Hence, in the limit of perfect data, there is only a probabilistic interpretation if $N_{\mathrm{d}} < N_{\mathrm{b}}$, assuming that the inverse for $\boldsymbol{\Phi}\lambda^2\boldsymbol{\Phi}^{\mathrm{T}}$ exists.

Intuitively this can be understood form the fact that we model the variance with the same basis functions as we use to model the function approximation. If we assume the data to be exact and if $N_{\mathrm{d}} \geq N_{\mathrm{b}}$, the contributions of all basis functions are fixed by the data and there are no undetermined degrees of freedom that can cause a spread in the resulting distribution conditioned on the data.

[2]Note, however, that the existence of the inverses of $\boldsymbol{\Phi}\lambda^2\boldsymbol{\Phi}^{\mathrm{T}}$ and $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ still depends on the choice of basis functions and the positions of the data points.

These considerations seem to imply that in order to learn about the true error, neglecting the error induced by the data, we should use as many basis functions as possible to capture the remaining freedom in the result. However, as the following example will show, indefinitely adding basis functions is also not advisable.

6.2.3.3 Example: Fourier basis

As an example, consider the set of the first $N_\mathrm{b} = 2N+1$ real Fourier basis functions, $\{1\} \cup \{\sin(\omega_n x)\}_{n=1}^{N} \cup \{\cos(\omega_n x)\}_{n=1}^{N}$, where we defined $\omega_n \equiv 2\pi n/L$, with L the size of the domain that we are interested in. Given these basis functions, the primal representation of the function approximation (6.5) corresponds to the (truncated) Fourier series of the function that we are looking for. If we denote the entry in $\boldsymbol{\lambda}$ corresponding to the constant with λ, the entries corresponding to the sines with λ_n, and the entries corresponding to the cosines with λ'_n, the resulting kernel reads,

$$\begin{aligned} k(x,x') \;=\; \lambda^2 \;&+\; \sum_{n=1}^{N} \left(\frac{\lambda_n^2 + \lambda_n'^2}{2}\right) \cos\left(\omega_n (x - x')\right) \\ &+\; \sum_{n=1}^{N} \left(\frac{\lambda_n^2 - \lambda_n'^2}{2}\right) \cos\left(\omega_n (x + x')\right). \end{aligned} \tag{6.46}$$

Typically, one would expect that modes corresponding to the same length scale, i.e. same n, would have similar weights, such that $\lambda_n \approx \lambda'_n$, and hence the second summation in (6.46) vanishes. This approximately renders the kernel into a radial basis function, $k(x,x') \approx K(x-x')$.

Now consider the simplest case, when all $\lambda_n = \lambda'_n = \lambda$. The kernel is then a radial basis function and can be computed explicitly,

$$k(x,x') \;=\; \frac{\lambda^2}{2}\left(1 + \frac{\sin\left(\pi(2N+1)(x-x')/L\right)}{\sin\left(\pi(x-x')/L\right)}\right), \tag{6.47}$$

which is commonly known as (one half plus) the Dirichlet kernel. The kernel attains its maximum on the diagonal, $k(x,x) = \lambda^2(N+1)$, and oscillates and decays away from there. The dual solution (6.11) is a linear combination of these radial basis functions centred (and thus peaking) around the data points.

Note that, as the number of basis functions increases, $N \to \infty$, the kernel becomes more narrow and peaked, and in the limit tends towards a delta distribution. Since the kernel centred around a data point represents the influence of that data point on the solution, this implies that with increasing N, the effect of each individual data point on the solution decreases and becomes ever more confined to a shrinking region around each data point. Similarly, with increasing N, the variance (or error estimate) in between data points will increase. Increasing the number of basis functions, while keeping the number of data points fixed, will imply that the basis functions are ever less constrained by the data, a problem commonly known as over-fitting.

One possible solution is regularisation, for instance by damping the higher modes in the kernel with $\boldsymbol{\lambda}$, making it less peaked in the limit of large N. Note that the entries of the regularisation vector, $\boldsymbol{\lambda}$, appear as the Fourier coefficients of the kernel (6.46). This shows the crucial interplay between the choice of basis functions and regularisation. However, note that by re-scaling the basis functions, $\phi_i \to \lambda_i \phi_i / c$, the regularisation vector can always be cast into the form $\boldsymbol{\lambda} = [\lambda_i] \to c\mathbf{1}$.

This example clearly illustrates that indefinitely adding basis functions without proper regularisation will not allow us to have a better error estimate, as it will always at the same time worsen the fit, thus artificially increasing the error (6.39).

6.2.3.4 Interpretation in terms of data-(de)compression

From a function approximation point of view, we are essentially approximating the true error function, $\varepsilon(x)$, in the space spanned by the same basis functions as we use for the original function approximation, $\tilde{f}(x)$, whereas one would expect the true error to live in the orthogonal complement of that space. Therefore, although the probabilistic error estimate (6.39) is some measure of the uncertainty in the model, it is far from exact. So what does it quantify? In Appendix B.6, we show that the probabilistic error estimate (6.39) can be obtained as the minimum of a (rather contrived) error measure, thus providing a framework to view it as a true error. However, it can be easier, and somehow more accurate, to think of the probabilistic error estimate, $\tilde{\varepsilon}(x)$, as a measure of the (de)compression of the data, $(\boldsymbol{x}, \boldsymbol{y})$, when it is represented by the model, as defined by the basis functions, $\{\phi_i\}$.

This point of view, allows us to use the probabilistic framework to formalise the model reductions presented in Chapter 5. However, first, we demonstrate how the probabilistic methods can be applied to radiative transfer computations.

6.3 Probabilistic radiative transfer

We can now apply the probabilistic numerical approach to the particular case of radiative transfer and show how the classical solution schemes arise as particular instances of the general probabilistic method. For a comprehensive overview of the different classical solution schemes, see e.g. [26].

6.3.1 Method of characteristics

For simplicity, consider the radiative transfer equation (1.1), in the absence of scattering and neglecting any frequency dependency[3], so it can then be written as,

$$\mathscr{L} I(s) \ = \ \eta(s), \tag{6.48}$$

with, s, the position on the ray, and the linear differential operator, $\mathscr{L}$, defined as,

$$\mathscr{L} \ \equiv \ \chi(s) \ + \ \partial_s . \tag{6.49}$$

The Green's function for this linear operator, $\mathscr{L}$, is given by,

$$G(z,s) \ = \ \Theta(s-z) \ e^{-\tau(z,s)}, \tag{6.50}$$

in which Θ is the Heaviside function (which is 1 for positive arguments, 0 otherwise), and τ is the optical depth, as defined in (3.3). Remember that, by definition of the Green's function, we have that,

$$\mathscr{L}_s G(z,\, s) \ = \ \delta(s-z). \tag{6.51}$$

[3]This is a significant simplification, since a frequency dependency in moving media would require us to consider the transfer equation in the co-moving frame (see Section 1.3.6). Although this can also perfectly be captured in this formalism, we omit it here for simplicity.

Using this Green's function one can (at least formally) solve the radiative transfer equation, as in the method of characteristics.

6.3.1.1 Classical method of characteristics

The method of characteristics solves the transfer equation from its formal solution (see Section 1.3.3), which is based on the Green's function. Given the boundary condition, $I(s_0) = I_0$, at boundary point, s_0, one finds, in accordance with (1.6),

$$I(s) \ = \ I_0 \, e^{-\tau(s_0,s)} \ + \ \int_{s_0}^{s} \mathrm{d}s' \, \eta(s') \, e^{-\tau(s',s)}. \tag{6.52}$$

The integral is then evaluated using a (local) interpolation for the emissivity, η.

At this point, a distinction is often made between short and long characteristic methods depending on the location of the point s_0 in the underlying discretisation. In short characteristic methods, s_0 is taken to be the previous point in the discretisation, while for long characteristics, it is the boundary of the computational domain. For our intents and purposes this distinction does not matter, so we continue with the general formulation as in (1.6), in which s_0 can be any point in the discretisation.

The emissivity is often interpolated using a linear scheme. Using, for instance, the dual formulation from Section 6.2.1.2, given a kernel, κ, the interpolant (6.11) for the emissivity, can be written as,

$$\eta(s) \ \equiv \ \boldsymbol{\eta}^{\mathrm{T}} \mathsf{K}^{-1} \kappa(\boldsymbol{a}, s), \tag{6.53}$$

where we defined the matrix, $\mathsf{K} \equiv \kappa(\boldsymbol{a},\boldsymbol{a}) + \sigma_{\mathrm{L}}^2$, and $\boldsymbol{a}$ is the vector of positions at which the values for η are given. One particularly popular kernel is the one corresponding to the basis of Lagrange polynomials, since they trivially satisfy the interpolation property. Substituting (6.53) in the formal solution (6.52) yields,

$$I(s) \ = \ I_0 \, e^{-\tau(s_0,s)} \ + \ \boldsymbol{\eta}^{\mathrm{T}} \mathsf{K}^{-1} \int_{s_0}^{s} \mathrm{d}s' \, \kappa(\boldsymbol{a}, s') \ e^{-\tau(s',s)} \tag{6.54}$$

such that the integrals can now be evaluated on the (analytically) known kernel function, κ, thus effectively solving the transfer equation.

6.3.1.2 Probabilistic method of characteristics

Now we show how the method of characteristics can be derived as a probabilistic regression problem in the dual formulation by choosing a particular type of kernel, or equivalently, by choosing a particular set of basis functions.

Given the Green's function (6.50) for the differential operator (6.49) in the radiative transfer equation, consider a kernel of the form (see also Appendix B.4),

$$k(z,s) \ = \ \int_{-\infty}^{+\infty} \text{d}s' \int_{-\infty}^{+\infty} \text{d}z' \ \kappa(s',z') \ G(z',z) \, G(s',s), \tag{6.55}$$

in which $\kappa(s',z')$ is another kernel of which we only demand that it does not correlate the region $s > s_0$ with $s < s_0$. This implies a block diagonal kernel of the form,

$$\begin{aligned} \kappa(z,s) \ &\equiv \ \Theta(s_0 - z)\Theta(s_0 - s) \, \kappa(z,s) \\ &\quad + \ \Theta(z - s_0)\Theta(s - s_0) \, \kappa(z,s). \end{aligned} \tag{6.56}$$

If we now assume that, $\forall a \in \boldsymbol{a} : a > s_0$, and we assume no error on the boundary condition, one can show (see appendix B.5) that the dual solution reads,

$$\tilde{f}_{\text{dual}}(s) \ = \ I_0 \, e^{-\tau(s_0,s)} \ + \ \boldsymbol{\eta}^{\text{T}} \mathsf{K}^{-1} \int_{s_0}^{s} \text{d}s' \ \kappa(\boldsymbol{a},s') \ e^{-\tau(s',s)} \tag{6.57}$$

with the corresponding variance (or error measure) given by,

$$\begin{aligned} \tilde{\sigma}^2_{\text{dual}}(s) \ = \ &\int_{s_0}^{s} \text{d}s' \int_{s_0}^{s} \text{d}z' \ G(z', s) \, G(s', s) \\ &\qquad \times \Big(\kappa(s', z') \ - \ \kappa(\boldsymbol{a}, s')^{\text{T}} \mathsf{K}^{-1} \, \kappa(\boldsymbol{a}, z') \Big) \, . \end{aligned} \tag{6.58}$$

Note that the probabilistic solution (6.57) is equivalent to the classical solution for the method of characteristics (6.54). Furthermore, we can identify the expression between brackets in the variance, i.e.,

$$\kappa(s', z') \ - \ \kappa(\boldsymbol{a}, s')^{\text{T}} \mathsf{K}^{-1} \, \kappa(\boldsymbol{a}, z'), \tag{6.59}$$

as the resulting variance corresponding to the interpolation scheme (6.53) that was used for interpolating the emissivity. Hence, the variance (6.58) is the convolution of the variance in the interpolation of the emissivity (6.59) with the Green's function.

6.3.2 Second-order method

The classical second-order or Schuster-Feautrier method was already described in Section 1.3.4, and its numerical implementation is described in Appendix A. To cast it into a probabilistic method, we identify in (1.16) the linear operator equation,

$$\mathscr{L} u(\tau) \ = \ S(\tau), \tag{6.60}$$

in which the differential operator, $\mathscr{L}$, is defined as,

$$\mathscr{L} \ \equiv \ 1 \ - \ \partial_{\tau}^{2}, \tag{6.61}$$

and, u, the mean intensity up and down the ray is defined as in (1.7). As described in Appendix A, equation (6.60) is then discretised and solved as a tridiagonal linear system of equations (see A.6). Calling this tridiagonal matrix, T, we can write,

$$\mathsf{T} \boldsymbol{u} \ = \ \boldsymbol{S}. \tag{6.62}$$

The boundary conditions are incorporated by modifying T and $\boldsymbol{S}$, see Section A.1.2. Considering perfect data (see Section 6.2.3.2), and assuming the resulting design matrix, $\boldsymbol{\Phi}$, to be invertible, the probabilistic solution can be written as,

$$\tilde{f}(x) \ = \ \boldsymbol{\phi}(x)^{\mathrm{T}} \boldsymbol{\Phi}^{-1} \boldsymbol{y}. \tag{6.63}$$

Comparing equations (6.62) and (6.63), and assuming interpolating basis functions, i.e. $\phi_i(x_d) = \delta_{id}$, we observe that to reproduce the second-order method, we require, $\mathsf{T} = \boldsymbol{\Phi}$, and the appropriate basis functions can be derived from (6.62) and (A.6).

Assuming perfect data, there is no probabilistic interpretation for the second-order method, since the variance will vanish everywhere (see also Section 6.2.3.2). In fact, this is true in general, whenever the resulting design matrix, $\boldsymbol{\Phi}$, is invertible.

As a result, one could conclude that nothing can be gained with the probabilistic approach for this type of solution methods, and that it is only relevant for the method of characteristics. However, when reducing models in the spirit of the mesh reduction techniques presented in Chapter 5, such that the corresponding number of basis functions becomes smaller than the number of data points, all types of solution methods can benefit from a probabilistic approach.

6.4 Probabilistic mesh reduction

In order to tie the ideas of mesh reduction presented in Chapter 5 to the probabilistic framework, we define a set of basis functions that is related to the data points. A particular choice, that relates to smoothed-particle hydrodynamics, is to pick a radial basis function, ψ, and define the set of basis functions as $\{\psi\,(\|x - x_d\|)\}$, such that there is one (radial) basis function for each data point, x_d. Reducing the model can then be achieved by disregarding the basis functions corresponding to the points one wants to eliminate. If the resulting system is solved in the primal formulation (6.5), the data for that point is still taken into account, while the computational cost to solve the system reduces together with the number of basis functions. Moreover, the effect of disregarding those basis functions can be captured by the variance (6.39). Moreover, the variance can be used to identify which points could be disregarded. This idea that probabilistic methods could be used to optimise the problem design is not new, and was already suggested, for instance, in [157]. However, some comments on the practicality of this approach are in order here.

First of all, when using radial basis functions (RBFs), the resulting solution method is practically equivalent to the well-established RBF-PDE solution methods, see e.g. [159] and the references therein. The key difference, however, is the probabilistic interpretation and the corresponding confidence intervals that now can be derived. A well-known issue for RBF-PDE solvers is the (ill)conditioning of the resulting linear system that needs to be solved [160, 161]. This is a challenge for the practical implementation of probabilistic mesh reduction, but, perhaps, also an opportunity to study the conditioning issues now also with probabilistic tools.

Second, as already alluded to in [157], we note that computing the variance (6.39) is computationally expensive. In particular, computing the variance is more expensive than computing the approximation (6.38), since the former requires the full matrix inverse while the latter only requires the solution of the corresponding linear system. This further complicates a practical implementation of the probabilistic reduction method in which the variance is used in the reduction criterion.

Finally, it should be noted that, although the probabilistic framework indeed allows us to formulate the reduction method from Section 5.4.2 in a more rigorous way, the more interesting question is to not only focus on removing points from the model, but to optimise the set of basis functions more generally, i.e. given the data, search for the minimal set of basis functions that still yields acceptable results. This is a key question that should be addressed in future work.

6.5 Future work

The probabilistic numerical approach promises opportunities for optimisation, while also providing a rigorous framework for uncertainty quantification. However, many details still have to be worked out to be able to fully utilise its power in practical simulations. In the following, we point out interesting directions for future research.

6.5.1 Optimisation

The probabilistic numerical method provides us with a mathematical framework to describe the trade-off between the computational speed and accuracy of a model, i.e. the speed can directly be inferred from the size of the corresponding linear system, while the accuracy can be gauged with the error measure (6.39). Since radiative transfer models often rely on a previous simulation step, the available data, $(\boldsymbol{x}, \boldsymbol{y})$, can be considered as a given. The remaining optimisation question is to find the minimal set of basis functions, $\{\phi_i\}$, that still solves the model to an acceptably accurate degree. There already exists a wealth of research in this direction generally known as sparse representation (see e.g. [162] and the references therein). In particular, recently, efficient algorithms have been developed to solve exactly the type of linear equations encountered in the probabilistic numerical method [163] (see also [158]). A key next step towards will be to make the probabilistic numerical method practically feasible, by applying these new algorithms.

6.5.2 Error estimates for finite element methods

As already pointed out, the probabilistic numerical method is very akin to finite element methods. In a sense, the probabilistic numerical method can be thought of as a finite element method that is augmented with a probabilistic interpretation. Finite element methods have already successfully been applied to (astrophysical) radiative transfer problems, see e.g. [44–46]. Furthermore, there is a wealth of research on so-called a posteriori error estimates for finite element methods, see e.g. [164, 165] and the references therein, and see [45] for an application to radiative transfer. Many of these error estimates are much cheaper to compute than the probabilistic one (6.39), but are more difficult to interpret and use in a statistical setting. For instance, when comparing models, or when combining the effects of various sources of uncertainties. It would be interesting to see whether a probabilistic interpretation could also be given for these cheap error estimates, or conversely, whether these could be used to define a computationally cheaper probabilistic error estimate.

6.5.3 Non-linear models

In this thesis, we limited ourselves to function approximations that are linear in the model parameters (weights). This has the clear advantage that it allows for explicit analytic solutions that can, at least in principle, be obtained by direct computations. Nevertheless, when the corresponding linear systems become large and sparse, one must eventually resort to iterative solvers, for instance minimising the mean squared error functions (6.3) or (6.8). At that point, the methods that are used to solve the system can equally well be employed to solve non-linear models, making the requirement of linearity somewhat obsolete[4]. One particularly interesting type of non-linear model to consider, is a neural network, i.e. an N-layered function of the form: $\boldsymbol{y}^{(l+1)} = \phi^{(l)}(\mathbf{A}^{(l)}\boldsymbol{y}^{(l)} + \boldsymbol{b}^{(l)})$, in which the input and output are respectively given by: $\boldsymbol{x} = \boldsymbol{y}^{(1)}$ and $\boldsymbol{y} = \boldsymbol{y}^{(N)}$, the $\phi^{(l)}$ are non-linear (activation) functions,

[4]Linearity remains crucial for the probabilistic interpretation, since a non-linear function does not necessarily preserve the distribution of its input, although approximations are still possible.

and the superscripts indicate the index of each layer. The matrices, $\mathbf{A}^{(l)}$, and vectors, $\boldsymbol{b}^{(l)}$, are the weights that need to be obtained (or learned) by minimising the error function. Neural networks are often good approximators[5] and there exist efficient algorithms to solve the error minimisation problem, see e.g. [146]. Using a neural network as an underlying model makes the solution method (Section 6.2.1.4) equivalent to the one used in physics-informed neural networks (PINNs; see also Section 1.5.2.2, [75–77]), with the mean squared error as the loss function. PINNs have already successfully been applied to radiative transfer [78], and recently also methods have been developed for uncertainty quantification on their results [169]. It would be interesting to see whether these uncertainties can also be interpreted in a probabilistic context. Furthermore, there exists a wealth of research in optimising neural network architectures. One particularly interesting direction would be to explore neural architecture search (NAS; [170]) methods, that search for the optimal architecture for a given problem. This is closely related to our quest for optimal basis functions (Section 6.5.1), and might lead to adaptive algorithms to find them.

6.5.4 Error estimates, preconditioning & ALI

As we have seen in Section 1.3.5 and Chapter 2, solving for the radiation field given the state of the medium is often only one step in an iterative process of finding a consistent solution for the radiation field and the state of the medium. Furthermore, in Section 2.3.2, we have seen how approximate solutions for the radiation field can help accelerate the convergence of this process by preconditioning the system with an approximated Lambda operator. As recently pointed out in [171], there is a clear link between finite element methods for error estimation and preconditioning of the corresponding linear system. Perhaps these finite element methods or their probabilistic equivalents can be used to obtain cheap but reliable approximated Lambda operators. Moreover, it should be possible to formulate the entire Lambda iteration process in a probabilistic setting, which might lead to more reliable convergence criteria, and hence more reliable results.

[5]Several neural network architectures can be shown to be universal approximators [166–168], and from the numerous examples it is clear that good approximation can already be achieved with a relatively limited number of weights, making them very efficient and thus practical approximators.

Chapter 7

Summary & Conclusions

Challenge Everything!

– The whispering voice from EA games

7.1 Summary

Throughout this thesis, we investigated the computational aspects of simulating the transport of electromagnetic radiation, with a particular focus on how to optimise the trade-off between computational speed and accuracy.

In Chapter 1, we introduced the radiative transfer problem, formulated it in terms of the radiative transfer equation, and discussed the different solution methods, with an emphasis on numerical solvers. Furthermore, we sketched the landscape of modern computing for which these solvers should be designed. We introduced the two key features that can be exploited on modern hardware: (1) the several layers of parallelism, and (2) hardware acceleration, and pointed out the modern evolution towards learning algorithms in software.

In Chapter 2, we focused on the radiative transfer of atomic and molecular lines. This requires an iterative solution method, known as Lambda iteration, to obtain a consistent coupling between the radiation field and the state of the medium. Lambda iterations are known for their slow convergence, but there are several methods to accelerate this. We review the classical scheme for accelerated Lambda iteration by Rybicki & Hummer [91], and the acceleration of convergence method by Ng [95]. For the latter, we demonstrate an improved dynamic scheme that does not depend on spurious parameters, but only on the amount of memory available in the system.

In Chapter 3, we presented the design strategy, tests, and benchmarks for our newly developed radiative transfer library, Magritte. In particular, we highlighted the modular way in which we implemented multi-core and multi-node parallelism as well as GPU acceleration, using the abstraction library ParAcAbs, that we specifically designed for that purpose. Magritte employs a ray-tracer and a second-order formal solver that can handle both structured and unstructured meshes as well as smoothed-particle data, making it a versatile tool for astrophysical modelling.

In Chapter 4, we demonstrated the applicability of our newly developed radiative transfer library, Magritte, by creating synthetic observations for several hydro-chemical models of stellar outflows arising from mass-losing AGB stars.

In Chapter 5, we focused on the issue of spatial, spectral, and directional discretisation in radiative transfer simulations. We presented an adaptive ray-tracing algorithm that can adaptively refine the directional discretisation and interpolate along the ray where necessary. Furthermore, we demonstrated an algorithm that can reduce typical input models for radiative transfer by an order of magnitude without significant loss of accuracy in the results. This strongly suggests the existence of more efficient representations for radiative transfer models.

In Chapter 6, we introduced a probabilistic numerical method to solve linear partial differential equations (PDEs) and, in particular, the radiative transfer equation. Assuming a Gaussian distribution for all input parameters, and a Gaussian prior over all possible functions, it computes the posterior distribution over the resulting functions, conditioned on the PDE and the corresponding boundary conditions. This naturally leads to a local measure of uncertainty on the result, based on the variance of the posterior distribution. Moreover, it can be used to reformulate the heuristic reduction algorithm (Chapter 5) in a more rigorous way. Furthermore, it provides a probabilistic framework to look more generally for more efficient representations of radiative transfer models. This is still work in progress, and we pointed out several connections, e.g. with more established finite element methods and physics-informed neural networks, and suggested possible directions of for future research which can lead to more rigorous optimisation methods for radiative transfer simulations.

7.2 Conclusions

This thesis covers several legs of an, as yet unfinished, journey towards optimally fast and reliable radiative transfer. This quest for optimality is not merely an academic question, but a sheer necessity to properly include all relevant radiative processes in astrophysical simulations, such as, for instance, stellar wind models[1] (see e.g. [21]). An optimal solution is still out of reach, but, nevertheless, several milestones have been reached, and there is a clear map for the road ahead.

The main milestone is our newly developed radiative transfer library Magritte. We established it as a state-of-the-art library that can leverage modern computing hardware in a scalable and portable way, thus ensuring its future place as a useful tool for both small and large-scale simulations. Furthermore, several advances were made along the way that can improve the performance of radiative transfer solvers in general, such as the improved Ng-acceleration method (Section 2.3.3), adaptive ray-tracing (Section 5.3), and the heuristic mesh reduction algorithm (Section 5.4.2). All of these advances result in a speed-up, but the latter two, which have the highest impact on performance, also imply concessions in accuracy. In an attempt to optimise this trade-off between computational speed and accuracy, we introduced a probabilistic numerical method in radiative transfer, which naturally allows for uncertainty quantification (Chapter 6). So far, we only explored the foundations of this framework, but could not yet provide an optimal solution. Therefore, to conclude this thesis, we envision three different routes for the journey ahead.

First, the representation for the radiative transfer model should be optimised. The results of our mesh reduction method (Section 5.5.2) strongly suggest that much sparser representations exist for typical radiative transfer models. The probabilistic numerical method, moreover, provides a framework to search for more generally optimised representations by constructing the optimal set of basis functions, given the partial differential equation and boundary conditions for the problem at hand. This optimal representation can be obtained either in a constructive way, such as, with

[1]Rough estimates suggest that with the current performance of Magritte, including radiative transfer in hydro-chemical stellar wind models would slow them down by more than an order of magnitude, making it in principle feasible, but still highly unpractical to do so.

operator-adapted wavelets tailored to a problem (see e.g. [158]), or in a reductive way, starting from an highly under-constrained representation, similar to the architecture optimisation methods for neural networks. A practical implementation of this will most likely rely on a hybrid scheme that balances between the optimisation of the representation and the solution of the problem, similar to neural architecture search, in which the architecture of the network is optimised during training.

Second, it should be noted that most astrophysical models do not require the full solution of the radiation field, but rather only require derived quantities, such as, radiative pressure, heating, or cooling. Since these typically do not depend on direction or frequency, they are of much smaller dimension than the radiation field, which suggests that they can be well-approximated much faster than the radiation field can be computed. Such an approximation can be achieved, for instance, by emulation, which has already proven to be a powerful approximation technique to replace time-consuming simulations (see also Section 1.5.2.1). However, with great approximation power comes great responsibility, in particular about the accuracy of the result. Therefore, these approximations should take into account the uncertainties in the exact method, as well as the additional uncertainties that are introduced by the approximation. Also here, the probabilistic numerical method can be employed to estimate both these uncertainties. Furthermore, an optimised representation for the radiative transfer model could also greatly improve the approximation power.

Finally, we could reduce the need for large and computationally expensive radiative transfer models by using them more efficiently and making more targeted models, for instance, of particular objects. Usually, in astronomy, we are hardly ever interested in all the details of a particular model, but rather in general trends in the parameter dependencies over large ensembles of models, unless, of course, when modelling a specific object. In the case of parameter studies, we can reduce the need for expensive radiative transfer by employing approximation methods, as discussed above. Still, when more detail is required, for instance, when modelling a particular object or particular observations, we often cannot afford to make many approximations, especially not in the radiative transfer, which determines how the model would appear in observations. We can, however, reduce the number of required models by specifically tailoring them to the observations. This can be achieved by de-projecting the observation into a model, which is in a sense the inverse of a synthetic observation. De-projection is a classical problem in astronomy for which several algorithms have been devised, see e.g. [172–174]. However, their use is limited by their difficulty to cope with the central fact that de-projection is an intrinsically under-constrained problem, i.e. not all information about the 3D distribution of a function can be derived form its 2D projection, see e.g. [175]. Recent advances in 3D-reconstruction with trained algorithms show promising results and can cope with the under-constrained problem in a probabilistic way [176]. Instead of generating a single de-projection for an image, they generate a distribution over all possible de-projections given an image, similar but conversely to the way in which a probabilistic radiative transfer model would result in a distribution over the possible synthetic observations. Therefore, we would propose to study the probabilistic numerical approach to radiative transfer both for the forward and inverse problem, such that trained algorithms, as in [176], could be employed to de-project astronomical observations into probabilistic models, thus allowing us to do more targeted and hence more efficient modelling.

7.3 Epilogue

René Magritte's most famous painting, *The Treachery of Images*, depicts a pipe with the French subscript: *“Ceci n'est pas une pipe"*, translated: *“This is not a pipe"*. For that reason, we named our radiative transfer library MAGRITTE, as a constant reminder that any synthetic observation it produces is merely a theoretical model, not a real observation, and thus the result of our numerous limiting assumptions. In an idealistic attempt to alleviate this somewhat disparaging connotation, one could ask: What would it take to raise the status of these models to the same level of scientific value as real observations? Or phrased more provocatively: What would it take to turn these simulations into science? To conclude this thesis, I will try to formulate an answer from my personal point of view.

The scientific value of a single realisation of a complex simulation is usually fairly limited[2] (see e.g. any single example in Chapter 4). On the other hand, an ensemble of realisations with varying input parameters can be of great scientific value, since it can elucidate, for instance, the macroscopic properties emerging from the microscopic laws implemented in the model (see e.g. [115, 116, 119]). However, as long as a simulation does not somehow correspond to reality, i.e. as long as it cannot be compared with observations, it will always remain, at least up to some level, a "scientist's impression" of some idealised world. Therefore, to elevate the status of simulations in science, they should be tied more closely to reality. This can be done either with hyper-realistic simulations including all relevant physics, making them as good as reality by construction, or by directly simulating specific observations with approximate models, thus explicitly introducing the link with reality. Since we often specifically lack the "relevant physics" required in the former approach, I see more potential in the latter. Directly simulating specific observations is exactly what can be achieved with de-projection (see Section 7.2). The key question is then, given a certain de-projection into an approximate model: In how far do the model and reality agree? This can be answered with a probabilistic approach, in which not a single de-projection is made, but rather a distribution over all possible de-projections is given. I suspect that such a distribution over all possible de-projections into an approximate model can have a similar or perhaps even a higher level of scientific value than the observation it is based upon. After all, in a sense, they provide a form of augmented reality, since we can use the probabilistic model to do experiments and test hypotheses. Ideally, within the probabilistic framework, we can also combine the data from different observations in the model, to arrive at a single virtual representation of the physical object, of course within the assumptions of the underlying model, but nevertheless a representation of a real object that we can experiment with. This would, at least up to some level, overthrow the commonly stated truism that in astronomy we cannot perform experiments on our objects of study, and could indeed further elevate the scientific status of simulations.

[2]Nevertheless, it can be of great artistic value, as a realisation (picture, movie, or game) of an idealised world, and, in the case of the black hole in the movie *Interstellar*, it can even be both [177].

Unfortunately, we are not there yet, but the probabilistic numerical approach to radiative transfer, proposed in this thesis, can hopefully be a step in this direction.

Appendix A

Numerical solution schemes

In this appendix, we briefly discuss the different numerical solution schemes that are implemented in MAGRITTE to solve the radiative transfer equation along a ray. Although these are widely covered in the literature (see e.g. [26, 27, 178]), we present them here to introduce our naming conventions, comment on some specific details of our implementations, and as a reference for the code documentation.

A.1 Second-order Schuster-Feautrier solver

The default solver in MAGRITTE is the improved second-order Schuster-Feautrier solver, proposed by Rybicki and Hummer in Appendix A of [91]. To appreciate the improvement in their method, we first introduce the original Feautrier scheme [33]. The second-order form (1.16) of the radiative transfer equation (1.1) was described in Section 1.3.4. Here, we discuss its numerical implementation.

A.1.1 Discretisation

Equation (1.16) can readily be written in discrete form, yielding,

$$u_n - \frac{\dfrac{u_{n+1}-u_n}{\Delta\tau_{n+1}} - \dfrac{u_n - u_{n-1}}{\Delta\tau_n}}{\dfrac{\Delta\tau_{n+1}+\Delta\tau_n}{2}} = S_n. \tag{A.1}$$

The indices, n, denote the location on the ray and are defined for u and S as, $n \in \{0, \ldots, N-1\}$, and for $\Delta\tau$ as, $n \in \{0, \ldots, N-2\}$, where in the latter there is one index less because $\Delta\tau$ is only defined in-between two consecutive points on the ray.

The discretised equation (A.1) can be written more schematically as,

$$-A_n u_{n-1} + B_n u_n - C_n u_{n+1} = S_n, \tag{A.2}$$

where the coefficients, for $n \in \{1, \ldots, N-2\}$, are defined as,

$$A_n \equiv \frac{2}{(\Delta\tau_{n-1} + \Delta\tau_n)\Delta\tau_{n-1}}, \tag{A.3}$$

$$B_n \equiv 1 + A_n + C_n, \tag{A.4}$$

$$C_n \equiv \frac{2}{(\Delta\tau_{n-1} + \Delta\tau_n)\Delta\tau_n}. \tag{A.5}$$

The other coefficients, for $n \in \{0, N-1\}$, are determined by the boundary conditions and will be discussed in the next section. Equation (A.2) can be interpreted as a tridiagonal matrix equation of the form,

$$\begin{pmatrix} B_0 & -C_0 & 0 & \cdots & 0 \\ -A_1 & B_1 & -C_1 & \cdots & 0 \\ 0 & -A_2 & B_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & B_{N-1} \end{pmatrix} \begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ u_{N-1} \end{pmatrix} = \begin{pmatrix} S_0 \\ S_1 \\ S_2 \\ \vdots \\ S_{N-1} \end{pmatrix}. \tag{A.6}$$

One can solve this tridiagonal matrix equation in a two-step process of elimination and back-substitution. The first element of the solution, u_0, can be related to the second one, u_1, using the first line in equation (A.6), yielding,

$$u_0 = S_0/B_0 + C_0/B_0 u_1. \tag{A.7}$$

Now one can show that it is always possible to write u_n in terms of u_{n+1}, i.e.,

$$u_n = z_n + D_n u_{n+1}, \tag{A.8}$$

where we introduced two new coefficients, z_n and D_n, which will be determined shortly. Substituting equation (A.8) in the first term of equation (A.2), one obtains,

$$u_n = (B_n - A_n D_{n-1})^{-1} (S_n + A_n z_{n-1}) + (B_n - A_n D_{n-1})^{-1} C_n u_{n+1}. \tag{A.9}$$

The two new coefficients can now be expressed in terms of their previous values,

$$z_n \equiv (B_n - A_n D_{n-1})^{-1} (S_n + A_n z_{n-1}), \tag{A.10}$$

$$D_n \equiv (B_n - A_n D_{n-1})^{-1} C_n. \tag{A.11}$$

To find all u_n, we first need to calculate all D_n and z_n (elimination step), and then substitute these back into equation (A.8) to retrieve the u_n (back-substitution).

One should note that the solution scheme presented above is only well-defined for non-vanishing optical depth increments. In the absence of population inversions, this can be guaranteed by using a small lower limit on the opacity. Another possible issue is the loss of significance when $B_n \approx A_n D_{n-1}$, due to numerical cancellation effects in the subtractions in equations (A.9, A.10, and A.11). This was remedied in the improved scheme proposed by Rybicki and Hummer in 1991 [91]. The key idea is to rewrite the recursion scheme (A.9, A.10, and A.11) in terms of a new variable,

$$F_n \equiv D_n^{-1} - 1. \tag{A.12}$$

This allows us to rewrite the elimination step in equations (A.10 and A.11) as,

$$F_n \equiv \left(1 + \frac{A_n F_{n-1}}{1 + F_{n-1}}\right) C_n^{-1}, \tag{A.13}$$

$$z_n \equiv \frac{S_n + A_n z_{n-1}}{C_n (1 + F_n)}. \tag{A.14}$$

Note the absence of any subtractions, thus resolving the possible loss of significance. Since each step refers to the previous one, we still need an equation for the first step.

This can be derived from equation (A.7),

$$F_0 = B_0/C_0 - 1, \tag{A.15}$$

$$z_0 = S_0/B_0. \tag{A.16}$$

Note that also the last step is well-defined. The mean intensity up and down the ray, u_n, can now again be obtained through back-substitution, using,

$$u_n = z_n + (1 + F_n)^{-1} u_{n+1}. \tag{A.17}$$

Now we defined a solution scheme relating the elements in the bulk of the ray, it remains to define appropriate boundary conditions on the two endpoints of the ray.

A.1.2 Boundary conditions

The boundary conditions on the two endpoints of the ray can be defined in terms of the intensities injected into the ray, i.e. $I_0^+ \equiv I_0(\hat{\boldsymbol{n}})$ and $I_N^- \equiv I_N(-\hat{\boldsymbol{n}})$, as described by Auer [108]. Suppose I_0^+ and I_N^- are known. These boundary conditions can be used to relate the mean intensity, u, and flux up and down the ray, v, using their definitions as, $v_0 = I_0^+ - u_0$, and $v_D = u_D - I_D^-$. We can use this to formulate an expression for the first and last row in the matrix equation. One should note that the first and last row describe a relation between the first and second elements of u, and between the last and one to last elements of u, respectively. Such a relation can be obtained by making a Taylor expansion from the boundary inward. For the boundary at index, $n = 0$, the Taylor expansion up to second order reads,

$$u_1 = u_0 + \Delta\tau_0 \left.\frac{\mathrm{d}u}{\mathrm{d}\tau}\right|_0 + \frac{1}{2}\Delta\tau_0^2 \left.\frac{\mathrm{d}^2u}{\mathrm{d}\tau^2}\right|_0. \tag{A.18}$$

From equations (1.11, neglecting scattering) and (1.16), together with the above boundary conditions on their variables, and assuming an isotropic source function at the boundary, we obtain,

$$u_1 = u_0 + \Delta\tau_0 \left(u_0 - I_0^+\right) + \frac{1}{2}\Delta\tau_0^2 \left(u_0 - S_0\right). \tag{A.19}$$

This can be rewritten as,

$$\left(1 + \frac{2}{\Delta\tau_0} + \frac{2}{\Delta\tau_0^2}\right) u_0 - \frac{2}{\Delta\tau_0^2} u_1 = S_0 + \frac{2}{\Delta\tau_0} I_0^+, \tag{A.20}$$

from which we can derive the coefficients for the first row of equation (A.6),

$$B_0 = 1 + \frac{2}{\Delta\tau_0} + \frac{2}{\Delta\tau_0^2}, \tag{A.21}$$

$$C_0 = \frac{2}{\Delta\tau_0^2}, \tag{A.22}$$

and we note that we have to add an extra term to the first element of the source,

$$S_0 \longrightarrow S_0 + \frac{2}{\Delta\tau_0} I_0^+. \tag{A.23}$$

Higher order relations can easily be obtained in a similar way. However, it turns out that second order suffices in most cases [108]. A similar calculation can be done for the other boundary, resulting in the relations,

$$A_{N-1} = \frac{2}{\Delta\tau_{N-2}^2}, \tag{A.24}$$

$$B_{N-1} = 1 + \frac{2}{\Delta\tau_{N-2}} + \frac{2}{\Delta\tau_{N-2}^2}, \tag{A.25}$$

and an extra term in the last element of the source,

$$S_{N-1} \longrightarrow S_{N-1} + \frac{2}{\Delta\tau_{N-2}} I_{N-1}^+. \tag{A.26}$$

Note that the index of $\Delta\tau$ only goes up to $n = N - 2$, since it is only defined in-between two consecutive points on the ray. This completes the default solution method to compute the radiation field in Magritte. However, for line radiative transfer, we often also want to retrieve the approximate Lambda operator from the solver to accelerate the iterative scheme (see also Section 2.3.2).

A.1.3 Approximate Lambda operator

In Section 2.3.2, the Lambda operator is defined as the operator that yields the mean intensity in the line evaluated at a certain point, when acting on the set of all level populations. Here, we will use the method outlined by Rybicki and Hummer in Appendix B of [91], to compute it for the numerical second-order Schuster-Feautrier method outlined above.

As a first step, we want to obtain the matrix, **L**, that yields the mean intensity along a ray, when acting on the set of sources,

$$\boldsymbol{u} \; = \; \mathbf{L}\boldsymbol{S}. \tag{A.27}$$

Comparing equations (A.27) and (A.6), it is clear that the matrix **L** is equal to the inverse of the tridiagonal matrix in equation (A.6). Using the schematic notation of equation (A.2), we can thus write,

$$-A_n \mathrm{L}_{n+1,p} \; + \; B_n \mathrm{L}_{n,p} \; - \; C_n \mathrm{L}_{n+1,p} \; = \; \delta_{n,p}. \tag{A.28}$$

Solving this row by row, top to bottom, one can obtain the recursion relations,

$$\begin{aligned} \mathrm{L}_{n,p} \; &= \; D_n \mathrm{L}_{n+1,p} \; + \; \mathrm{Z}_{n,p}, && \text{(A.29)} \\ \mathrm{Z}_{n,p} \; &\equiv \; \left(B_n \; - \; A_n D_{n-1}\right)^{-1} \left(\delta_{n,p} \; + \; A_n \mathrm{Z}_{n-1,p}\right), && \text{(A.30)} \\ D_n \; &\equiv \; \left(B_n \; - \; A_n D_{n-1}\right)^{-1} C_n. && \text{(A.31)} \end{aligned}$$

Similarly, by solving row by row, bottom to top, on can obtain,

$$\begin{aligned} \mathrm{L}_{n,p} \; &= \; E_n \mathrm{L}_{n-1,p} \; + \; \mathrm{W}_{n,p}, && \text{(A.32)} \\ \mathrm{W}_{n,p} \; &\equiv \; \left(B_n \; - \; C_n E_{n+1}\right)^{-1} \left(\delta_{n,p} \; + \; C_n \mathrm{W}_{n+1,p}\right), && \text{(A.33)} \\ E_n \; &\equiv \; \left(B_n \; - \; C_d E_{n+1}\right)^{-1} A_n. && \text{(A.34)} \end{aligned}$$

Using that $\delta_{n,p} = 0$, for $n \neq p$, we can find that $\mathrm{Z}_{n,p} = 0$, for $n < p$, and that $\mathrm{W}_{n,p} = 0$, for $n > p$. Using this in equations (A.29, A.30, and A.31), we can obtain,

$$\mathrm{L}_{n,n} = D_n \mathrm{L}_{n+1,n} + \mathrm{Z}_{n,n} \tag{A.35}$$

$$\mathrm{L}_{n-1,n} = D_{n-1} \mathrm{L}_{n,n} \tag{A.36}$$

$$\mathrm{Z}_{n,n} = (B_n - A_n D_{n-1})^{-1}. \tag{A.37}$$

Similarly for equations (A.32, A.33, and A.34), this yields,

$$\mathrm{L}_{n,n} = E_n \mathrm{L}_{n-1,n} + \mathrm{W}_{n,n} \tag{A.38}$$

$$\mathrm{L}_{n+1,n} = E_{n+1} \mathrm{L}_{n,n} \tag{A.39}$$

$$\mathrm{W}_{n,n} = (B_n - C_n E_{n+1})^{-1}. \tag{A.40}$$

One can now use both equations (A.35, A.36, and A.37) and (A.38, A.39, and A.40) to obtain the diagonal elements of the matrix, **L**,

$$\mathrm{L}_{n,n} = (1 - D_n E_{n+1})^{-1} (B_n - A_n D_{n-1})^{-1}. \tag{A.41}$$

The off-diagonal elements can furthermore be computed recursively using,

$$\mathrm{L}_{n,p} = D_n \mathrm{L}_{n+1,p} \quad \text{for} \quad n < p, \tag{A.42}$$

$$\mathrm{L}_{n,p} = E_n \mathrm{L}_{n-1,p} \quad \text{for} \quad n > p. \tag{A.43}$$

Hence, using equations (A.41, A.42, and A.43), the entire matrix, **L**, can be reconstructed. However, there are still subtractions which can lead to numerical loss of significance. This can again be remedied by introducing new variables,

$$F_n \equiv D_n^{-1} - 1, \tag{A.44}$$

$$G_n \equiv E_n^{-1} - 1. \tag{A.45}$$

Note that F_n is the same variable that was used before. These can be used to rewrite the recursion relations as,

$$F_n \equiv \left(1 + \frac{A_n F_{n-1}}{1 + F_{n-1}}\right) C_n^{-1}, \tag{A.46}$$

$$G_n \equiv \left(1 + \frac{C_n G_{n+1}}{1 + G_{n+1}}\right) A_n^{-1}. \tag{A.47}$$

Since the recursion relation for F_n refers to F_{n-1}, and the recursion relation for G_n refers to G_{n+1}, we still need to define the first and last elements respectively. These can again be derived from the first and last rows in matrix equation (A.28),

$$F_0 = B_0/C_0 - 1, \tag{A.48}$$

$$G_N = B_N/A_N - 1. \tag{A.49}$$

The diagonal elements of **L** can then be computed in terms of the new variables as,

$$\mathrm{L}_{n,n} = \left(F_n + \frac{G_{n+1}}{1 + G_{n+1}}\right)^{-1} C_n^{-1}. \tag{A.50}$$

Furthermore, the off-diagonal elements can be computed as,

$$\mathrm{L}_{n,p} = (1 + F_n)^{-1}\, \mathrm{L}_{n+1,p} \quad \text{for} \quad n < p, \tag{A.51}$$

$$\mathrm{L}_{n,p} = (1 + G_n)^{-1}\, \mathrm{L}_{n-1,p} \quad \text{for} \quad n > p. \tag{A.52}$$

Finally, the Lambda operator can be constructed from **L**, by averaging over all directions (i.e. rays originating from a point) and integrating over the line profile,

$$J_{ij} = \Lambda_{ij}[\mathbf{N}] \equiv \oint \frac{\mathrm{d}\hat{\mathbf{n}}}{4\pi} \int_0^{\infty} \mathrm{d}\nu\, \phi_\nu^{ij}\, \mathbf{L}\, \boldsymbol{S}. \tag{A.53}$$

Approximations to the Λ-operator can now be constructed using approximations to the matrix **L**. Typically, band-diagonal approximations to **L** are used, which can conveniently be constructed using equations (A.42 and A.43).

A.2 Fourth-order Schuster-Feautrier solver

In 1976, Auer proposed a simple scheme [109] to improve the accuracy of the Schuster-Feautrier scheme that requires minimal modifications and can readily be combined with the improvements made by Rybicki and Hummer [91]. This scheme is known as the fourth-order, or Hermitian, Schuster-Feautrier solver.

A.2.1 Discretisation

The key idea is to replace equation (A.2) by an equation of the form,

$$-\alpha_n u_{n-1} + \beta_n u_n - \gamma_n u_{n+1} = a_n S_{n-1} + b_n S_n + c_n S_{n+1}, \tag{A.54}$$

that also takes into account the source at multiple points along the ray. In order to find the coefficients, we can eliminate the sources using (1.16), which should hold at each point, i.e. $u_n - u_n'' = S_n$, for all n, which yields,

$$A_n u_{n-1} - B_n u_n + C_n u_{n+1} = a_n u_{n-1}'' + b_n u_n'' + c_n u_{n+1}'', \tag{A.55}$$

where the primes denote derivatives with respect to the optical depth, and we rather suggestively defined the three new coefficients,

$$A_n \equiv \alpha_n + a_n, \tag{A.56}$$

$$B_n \equiv \beta_n - b_n, \tag{A.57}$$

$$C_n \equiv \gamma_n + c_n. \tag{A.58}$$

In order to relate the discretized equation (A.55) back to its continuum equivalent, we can Taylor expand u and u'', to relate the neighbouring values,

$$u_{n\pm1} = u_n \pm u_n' \Delta\tau_\pm + \frac{1}{2} u_n'' \Delta\tau_\pm^2 \pm \frac{1}{6} u_n''' \Delta\tau_\pm^3 + \frac{1}{24} u_n'''' \Delta\tau_\pm^4, \tag{A.59}$$

$$u_{n\pm1}'' = u_n'' \pm u_n''' \Delta\tau_\pm + \frac{1}{2} u_n'''' \Delta\tau_\pm^2, \tag{A.60}$$

where, using the indexing conventions, $\Delta\tau_+ \equiv \Delta\tau_n$, and $\Delta\tau_- \equiv \Delta\tau_{n-1}$. We limit ourselves to fourth order, since that will impose sufficient constraints to determine the coefficients. Substituting the expansions in equation (A.55) and collecting the terms for each order of u yields,

$$\begin{aligned} & u_n\Big(A_n - B_n + C_n\Big) \\ & - u_n'\Big(A_n\Delta\tau_- - C_n\Delta\tau_+\Big) \\ & + u_n''\left(\frac{1}{2}\left(A_n\Delta\tau_-^2 + C_n\Delta\tau_+^2\right) - a_n - b_n - c_n\right) \\ & - u_n'''\left(\frac{1}{6}\left(A_n\Delta\tau_-^3 + C_n\Delta\tau_+^3\right) + a_n\Delta\tau_- - c_n\Delta\tau_+\right) \\ & + u_n''''\left(\frac{1}{24}\left(A_n\Delta\tau_-^4 + C_n\Delta\tau_+^4\right) + \frac{1}{2}\left(a_n\Delta\tau_-^2 - c_n\Delta\tau_+^2\right)\right) = 0. \end{aligned} \tag{A.61}$$

Enforcing this equality for every order of u results in a set of five equations,

$$\begin{cases} A_n + C_n & = B_n \\ A_n\Delta\tau_- - C_n\Delta\tau_+ & = 0 \\ \frac{1}{2}\left(A_n\Delta\tau_-^2 + C_n\Delta\tau_+^2\right) & = a_n + b_n + c_n \\ \frac{1}{6}\left(A_n\Delta\tau_-^3 + C_n\Delta\tau_+^3\right) & = c_n\Delta\tau_+ - a_n\Delta\tau_- \\ \frac{1}{12}\left(A_n\Delta\tau_-^4 + C_n\Delta\tau_+^4\right) & = c_n\Delta\tau_+^2 - a_n\Delta\tau_-^2 \end{cases} . \tag{A.62}$$

These five equations are not sufficient to determine the six coefficients. However, this was to be expected, since the coefficients defined by equation (A.54) are only determined up to a constant factor. Hence, we can choose the value of one of the coefficients. Considering the relation between A_n and C_n, given by the second equation in (A.62), and considering the values of the coefficients in the original second-order scheme, a particularly useful choice is to define,

$$A_n \equiv \frac{2}{(\Delta\tau_{n-1}+\Delta\tau_n)\Delta\tau_{n-1}}, \tag{A.63}$$

$$B_n \equiv A_n + C_n, \tag{A.64}$$

$$C_n \equiv \frac{2}{(\Delta\tau_{n-1}+\Delta\tau_n)\Delta\tau_n}. \tag{A.65}$$

Note that these are almost the same definitions as in the original second-order scheme (A.3, A.4, and A.5), but that they appear in the recursion relation (A.54) in a slightly different way. This leaves us with a set of three equations and three unknowns,

$$\begin{cases} 1 & = a_n + b_n + c_n \\ \frac{\Delta\tau_-^2+\Delta\tau_+^2}{3(\Delta\tau_{n-1}+\Delta\tau_n)} & = c_n\Delta\tau_+ - a_n\Delta\tau_- \, , \\ \frac{\Delta\tau_-^3+\Delta\tau_+^3}{6(\Delta\tau_{n-1}+\Delta\tau_n)} & = c_n\Delta\tau_+^2 - a_n\Delta\tau_-^2 \end{cases} \tag{A.66}$$

where we again used the old indexing convention ($\Delta\tau_+ \equiv \Delta\tau_n$, and $\Delta\tau_- \equiv \Delta\tau_{n-1}$). This set of equations can readily be solved, yielding all coefficients,

$$A_n \equiv \frac{2}{(\Delta\tau_{n-1}+\Delta\tau_n)\Delta\tau_{n-1}}, \tag{A.67}$$

$$B_n \equiv A_n + C_n, \tag{A.68}$$

$$C_n \equiv \frac{2}{(\Delta\tau_{n-1}+\Delta\tau_n)\Delta\tau_n}, \tag{A.69}$$

$$a_n \equiv \frac{1}{12}\left(2 - A_n\Delta\tau_n^2\right), \tag{A.70}$$

$$b_n \equiv 1 - a_n - c_n, \tag{A.71}$$

$$c_n \equiv \frac{1}{12}\left(2 - C_n\Delta\tau_{n-1}^2\right). \tag{A.72}$$

Hence, the original second-order solver can be turned into a fourth-order solver by substituting in the original equations,

$$A_n \;\longrightarrow\; \alpha_n = A_n - \frac{1}{12}\left(2 - A_n \Delta\tau_n^2\right), \tag{A.73}$$

$$B_n \;\longrightarrow\; \beta_n = 1 + (A_n - a_n) + (C_n - c_n), \tag{A.74}$$

$$C_n \;\longrightarrow\; \gamma_n = C_n - \frac{1}{12}\left(2 - C_n \Delta\tau_{n-1}^2\right), \tag{A.75}$$

and treating the source function accordingly,

$$S_n \;\longrightarrow\; a_n S_{n-1} + b_n S_n + c_n S_{n+1}. \tag{A.76}$$

Note that the old definition of $B_n \longrightarrow \beta_n$ remains valid when substituting the new definitions of A_n and C_n. As a result, also the solution methods presented for the second-order solver remain valid.

A.2.2 Boundary conditions

To obtain the boundary conditions for the fourth-order scheme, we consider the first and last row in the matrix equation corresponding to (A.54). The first row yields,

$$\beta_0 u_0 - \gamma_0 u_1 = b_0 S_0 + c_0 S_1 + S^+_{\text{bdy}}, \tag{A.77}$$

where we envisioned an additional term in the source, S_{bdy}, similar to the boundary conditions in the original second-order scheme. We can again eliminate the sources by demanding that (1.16) holds for both $n = 0$ and $n = 1$, yielding,

$$-B_0 u_0 + C_0 u_1 = b_0 u_0'' + c_0 u_1'' - S^+_{\text{bdy}}, \tag{A.78}$$

where we again introduced auxiliary variables,

$$B_0 \equiv \beta_0 - b_0, \tag{A.79}$$

$$C_0 \equiv \gamma_0 + c_0. \tag{A.80}$$

To relate the values of u at the two points, we can again perform a Taylor expansion,

$$\begin{aligned} u_1 &= u_0 + u_0' \Delta\tau_0 + \frac{1}{2} u_0'' \Delta\tau_0^2 + \frac{1}{6} u_0''' \Delta\tau_0^3, && \text{(A.81)} \\ u_1'' &= u_0'' + u_0''' \Delta\tau_0. && \text{(A.82)} \end{aligned}$$

We limit ourselves to third order, since that will impose sufficient constraints to determine the coefficients. Substituting the expansions in (A.78) yields,

$$\begin{aligned} & u_0\Big(-B_0 + C_0 \Big) \\ & - u_0'\Big(-C_0 \Delta\tau_0 \Big) \\ & + u_0''\left(\frac{1}{2} C_0 \Delta\tau_0^2 - b_0 - c_0 \right) \\ & - u_0'''\left(\frac{1}{6} C_0 \Delta\tau_0^3 - c_0 \Delta\tau_0 \right) = -S^{+}_{\text{bdy}}. \end{aligned} \tag{A.83}$$

Now, we still have to impose the boundary condition, given by the intensity injected into the ray, i.e. $I_0^+ \equiv I_0(\hat{\boldsymbol{n}})$, this yields,

$$u_0' = -v_0 = u_0 - I_0^+. \tag{A.84}$$

Using this in equation (A.83) finally gives,

$$\begin{aligned} & u_0\Big(-B_0 + C_0 + C_0 \Delta\tau_0 \Big) \\ & + u_0''\left(\frac{1}{2} C_0 \Delta\tau_0^2 - b_0 - c_0 \right) \\ & - u_0'''\left(\frac{1}{6} C_0 \Delta\tau_0^3 - c_0 \Delta\tau_0 \right) = -S^{+}_{\text{bdy}} + C_0 \Delta\tau_0 I_0^+. \end{aligned} \tag{A.85}$$

Enforcing this equality for every order of u results in a set of four equations,

$$\begin{cases} B_0 & = \ (1 + \Delta\tau_0)\, C_0 \\ \frac{1}{2} C_0 \Delta\tau_1^2 & = \ b_0 + c_0 \\ \frac{1}{6} C_0 \Delta\tau_0^2 & = \ c_0 \\ S^+_{\text{bdy}} & = \ C_0 \Delta\tau_0 I_0^+ \end{cases} . \tag{A.86}$$

These four equations are not sufficient to determine the five coefficients, which is again not surprising, since they are only defined up to a constant factor. Hence, we can choose one of the coefficients. Considering the values of the original Feautrier scheme, a particularly useful choice is,

$$C_0 = \frac{2}{\Delta\tau_0^2}, \tag{A.87}$$

resulting in the complete solution of the system, given by,

$$B_0 = \frac{2}{\Delta\tau_0} + \frac{2}{\Delta\tau_0^2}, \tag{A.88}$$

$$C_0 = \frac{2}{\Delta\tau_0^2}, \tag{A.89}$$

$$b_0 = \frac{2}{3}, \tag{A.90}$$

$$c_0 = \frac{1}{3}, \tag{A.91}$$

$$S^+_{\text{bdy}} = \frac{2}{\Delta\tau_0} I_0^+. \tag{A.92}$$

So we can deduce coefficients accounting for the boundary conditions,

$$B_0 \ \longrightarrow \ \beta_0 = \frac{2}{3} + \frac{2}{\Delta\tau_0} + \frac{2}{\Delta\tau_0^2}, \tag{A.93}$$

$$C_0 \ \longrightarrow \ \gamma_0 = \frac{2}{\Delta\tau_0^2} - \frac{1}{3}. \tag{A.94}$$

Similarly, one can find for the other end of the ray,

$$\begin{aligned} A_{N-1} &= \frac{2}{\Delta\tau_{N-2}^2}, && \text{(A.95)} \\ B_{N-1} &= \frac{2}{\Delta\tau_{N-2}^2} + \frac{2}{\Delta\tau_{N-2}}, && \text{(A.96)} \\ a_{N-1} &= \frac{1}{3}, && \text{(A.97)} \\ b_{N-1} &= \frac{2}{3}, && \text{(A.98)} \\ S_{\text{bdy}}^{-} &= \frac{2}{\Delta\tau_{N-2}} I_{N-1}^{-}. && \text{(A.99)} \end{aligned}$$

resulting in the coefficients,

$$\begin{aligned} A_{N-1} \;\rightarrow\; \gamma_{N-1} &= \frac{2}{\Delta\tau_{N-2}^2} - \frac{1}{3}, && \text{(A.100)} \\ B_{N-1} \;\rightarrow\; \beta_{N-1} &= \frac{2}{3} + \frac{2}{\Delta\tau_{N-2}} + \frac{2}{\Delta\tau_{N-2}^2}. && \text{(A.101)} \end{aligned}$$

Note that the index of $\Delta\tau$ only goes up to $n = N - 2$, since it is only defined in-between two consecutive points on the ray. This completes the fourth-order (Hermitian) solution scheme for the radiative transfer equation.

Appendix B

Mathematical background

This appendix provides some further mathematical background to support some claims in the main text of Chapter 6 that were presented without proof.

B.1 Equivalence between primal & dual formulation

To show the equivalence between the primal and dual formulation of the linear regression problem in Section 6.2.1, we have to prove (6.12), or equivalently, that,

$$\sigma^{-2}\mathbf{\Phi}\left(\mathbf{\Phi}^{\mathrm{T}}\sigma^{-2}\mathbf{\Phi}+\lambda^{-2}\right)^{-1}-\left(\mathbf{\Phi}\lambda^{2}\mathbf{\Phi}^{\mathrm{T}}+\sigma^{2}\right)^{-1}\mathbf{\Phi}\lambda^{2}=0. \tag{B.1}$$

Using the Woodburry matrix identity, the second term can be expanded as,

$$\sigma^{-2}\mathbf{\Phi}\lambda^{2}-\sigma^{-2}\mathbf{\Phi}\left(\lambda^{-2}+\mathbf{\Phi}^{\mathrm{T}}\sigma^{-2}\mathbf{\Phi}\right)^{-1}\mathbf{\Phi}^{\mathrm{T}}\sigma^{-2}\mathbf{\Phi}\lambda^{2}. \tag{B.2}$$

Substituting the result (B.2) in equation (B.1), ignoring the overall factor, $\sigma^{-2}\mathbf{\Phi}$, and isolating the terms with the inverse, it remains to show that,

$$\left(\mathbf{\Phi}^{\mathrm{T}}\sigma^{-2}\mathbf{\Phi}+\lambda^{-2}\right)^{-1}\left(1+\mathbf{\Phi}^{\mathrm{T}}\sigma^{-2}\mathbf{\Phi}\lambda^{2}\right)-\lambda^{2}=0. \tag{B.3}$$

Rewriting the second factor by extracting the λ^{2}, yields,

$$\left(\mathbf{\Phi}^{\mathrm{T}}\sigma^{-2}\mathbf{\Phi}+\lambda^{-2}\right)^{-1}\left(\lambda^{-2}+\mathbf{\Phi}^{\mathrm{T}}\sigma^{-2}\mathbf{\Phi}\right)\lambda^{2}-\lambda^{2}=0. \tag{B.4}$$

This clearly holds, and thus the primal and dual solutions are indeed equivalent.

B.2 Marginal & conditional Gaussians

Given a (marginal) Gaussian distribution, $p(\boldsymbol{x})$, and a corresponding conditional Gaussian distribution, $p(\boldsymbol{y}\,|\,\boldsymbol{x})$, which are both defined as,

$$p(\boldsymbol{x}) = \mathcal{N}\big(\boldsymbol{\mu}_{\boldsymbol{x}},\ \boldsymbol{\Sigma}_{\boldsymbol{x}}\big), \tag{B.5}$$

$$p(\boldsymbol{y}\,|\,\boldsymbol{x}) = \mathcal{N}\big(A\boldsymbol{x}+\boldsymbol{b},\ \boldsymbol{\Sigma}_{\boldsymbol{y}|\boldsymbol{x}}\big), \tag{B.6}$$

the other corresponding marginal distribution, $p(\boldsymbol{y})$, and the reverse conditional distribution, $p(\boldsymbol{x}\,|\,\boldsymbol{y})$, are given by,

$$p(\boldsymbol{y}) = \mathcal{N}\left(A\boldsymbol{\mu}_{\boldsymbol{x}}+\boldsymbol{b},\ \boldsymbol{\Sigma}_{\boldsymbol{y}|\boldsymbol{x}} + A\boldsymbol{\Sigma}_{\boldsymbol{x}}A^{\mathrm{T}}\right), \tag{B.7}$$

$$p(\boldsymbol{x}\,|\,\boldsymbol{y}) = \mathcal{N}\left(\boldsymbol{\Sigma}\left(A^{\mathrm{T}}\boldsymbol{\Sigma}_{\boldsymbol{y}|\boldsymbol{x}}^{-1}(\boldsymbol{y}-\boldsymbol{b}) + \boldsymbol{\Sigma}_{\boldsymbol{x}}^{-1}\boldsymbol{\mu}_{\boldsymbol{x}}\right),\ \boldsymbol{\Sigma}\right), \tag{B.8}$$

in which we defined the covariance matrix,

$$\boldsymbol{\Sigma} \equiv \left(\boldsymbol{\Sigma}_{\boldsymbol{x}}^{-1} + A^{\mathrm{T}}\boldsymbol{\Sigma}_{\boldsymbol{y}|\boldsymbol{x}}^{-1}A\right)^{-1}. \tag{B.9}$$

These relations can explicitly be derived by the method of "completing the square" in the distribution function and collecting the relevant terms, as described in detail, for instance, in [146].

B.3 Conditioning a Gaussian

Consider a stochastic vector variable, $\boldsymbol{y}$, defined by two separate stochastic vector variables, $\boldsymbol{a}$ and $\boldsymbol{b}$, and assume that all their components follow a (multivariate) Gaussian distribution, i.e.,

$$\boldsymbol{y} = \begin{bmatrix}\boldsymbol{a}\\ \boldsymbol{b}\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix}\boldsymbol{\mu}_{\boldsymbol{a}}\\ \boldsymbol{\mu}_{\boldsymbol{b}}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\Sigma}_{\boldsymbol{aa}} & \boldsymbol{\Sigma}_{\boldsymbol{ab}}\\ \boldsymbol{\Sigma}_{\boldsymbol{bb}} & \boldsymbol{\Sigma}_{\boldsymbol{bb}}\end{bmatrix}\right), \tag{B.10}$$

in which, $\boldsymbol{\mu}_{\boldsymbol{a}}$ and $\boldsymbol{\mu}_{\boldsymbol{b}}$ are the mean vectors and the matrices $\boldsymbol{\Sigma}_{\boldsymbol{aa}}$, $\boldsymbol{\Sigma}_{\boldsymbol{ab}} = \boldsymbol{\Sigma}_{\boldsymbol{ba}}^{\mathrm{T}}$, and $\boldsymbol{\Sigma}_{\boldsymbol{bb}}$, together form the covariance matrix. Now, we can ask what the resulting distribution of $\boldsymbol{a}$ would be, given prior knowledge about the value for $\boldsymbol{b}$. Fixing the value of $\boldsymbol{b}$

again yields a multivariate Gaussian distribution,

$$p\,(\boldsymbol{a} \mid \boldsymbol{b}) \;=\; \mathcal{N}\big(\boldsymbol{\mu}_{\boldsymbol{a}|\boldsymbol{b}}, \boldsymbol{\Sigma}_{\boldsymbol{a}|\boldsymbol{b}}\big), \tag{B.11}$$

in which the conditioned mean and variance are given by,

$$\boldsymbol{\mu}_{\boldsymbol{a}|\boldsymbol{b}} \;=\; \boldsymbol{\mu}_{\boldsymbol{a}} \,+\, \boldsymbol{\Sigma}_{\boldsymbol{ab}}\boldsymbol{\Sigma}_{\boldsymbol{bb}}^{-1}\left(\boldsymbol{b}-\boldsymbol{\mu}_{\boldsymbol{b}}\right), \tag{B.12}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{a}|\boldsymbol{b}} \;=\; \boldsymbol{\Sigma}_{\boldsymbol{aa}} \,-\, \boldsymbol{\Sigma}_{\boldsymbol{ab}}\boldsymbol{\Sigma}_{\boldsymbol{bb}}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{ba}}. \tag{B.13}$$

These relations can explicitly be derived by the method of "completing the square" in the distribution function and collecting the relevant terms, as described in detail, for instance, in [146]. Note that without any correlation between $\boldsymbol{a}$ and $\boldsymbol{b}$, i.e. $\boldsymbol{\Sigma}_{\boldsymbol{ab}} = \boldsymbol{\Sigma}_{\boldsymbol{ba}}^{\mathrm{T}} = 0$, the prior knowledge about $\boldsymbol{b}$ will not affect the distribution of $\boldsymbol{a}$, which is in line with what one would expect.

B.4 A kernel based on Green's functions

Given a linear PDE, and given the corresponding Green's function, G, for the differential operator, one can construct a kernel of the form,

$$k(z,s) \;=\; \int_{-\infty}^{+\infty} \mathrm{d}s' \int_{-\infty}^{+\infty} \mathrm{d}z' \;\kappa(s',z')\; G(z',z)\,G(s',s), \tag{B.14}$$

based on another kernel, κ. For later convenience, we define a new function, g, that, using the Green's functions, can be expressed as,

$$g(s,z) \;\equiv\; \mathscr{L}_2 k(z,s) \;=\; \int_{-\infty}^{+\infty} \mathrm{d}z' \;\kappa(s,z')\; G(z',z), \tag{B.15}$$

$$g(z,s) \;\equiv\; \mathscr{L}_1 k(z,s) \;=\; \int_{-\infty}^{+\infty} \mathrm{d}s' \;\kappa(s',z)\; G(s',s), \tag{B.16}$$

in which the subscript on the differential operator, $\mathscr{L}$, indicates whether it acts on the first or second argument. Note that both definitions are indeed consistent, since

k is symmetric in its arguments. Using the Green's functions again, one can derive,

$$\mathscr{L}_1\mathscr{L}_2 k(z,s) = \mathscr{L}_1 g(s,z) = \kappa(s,z), \tag{B.17}$$

$$\mathscr{L}_2\mathscr{L}_1 k(z,s) = \mathscr{L}_2 g(z,s) = \kappa(s,z). \tag{B.18}$$

Solving the PDE as a probabilistic regression problem, the corresponding covariance matrix of the joint distribution, is given by,

$$\begin{pmatrix} \mathscr{L}_1\mathscr{L}_2 k(\boldsymbol{a},\boldsymbol{a}) & \mathscr{L}_1\mathscr{B}_2 k(\boldsymbol{a},\boldsymbol{b}) & \mathscr{L}_1 k(\boldsymbol{a},s) \\ \mathscr{B}_1\mathscr{L}_2 k(\boldsymbol{b},\boldsymbol{a}) & \mathscr{B}_1\mathscr{B}_2 k(\boldsymbol{b},\boldsymbol{b}) & \mathscr{B}_1 k(\boldsymbol{b},s) \\ \mathscr{L}_2 k(s,\boldsymbol{a}) & \mathscr{B}_2 k(s,\boldsymbol{b}) & k(s,s) \end{pmatrix} \tag{B.19}$$

and can be simplified using the definitions above yielding,

$$\begin{pmatrix} \kappa(\boldsymbol{a},\boldsymbol{a}) & \mathscr{B}_1 g(\boldsymbol{a},\boldsymbol{b}) & g(\boldsymbol{a},s) \\ \mathscr{B}_1 g(\boldsymbol{a},\boldsymbol{b}) & \mathscr{B}_1\mathscr{B}_2 k(\boldsymbol{b},\boldsymbol{b}) & \mathscr{B}_1 k(\boldsymbol{b},s) \\ g(\boldsymbol{a},s) & \mathscr{B}_2 k(s,\boldsymbol{b}) & k(s,s) \end{pmatrix}. \tag{B.20}$$

The requirement that this matrix is positive semi-definite for all Green's functions, G, can be reduced to the condition that,

$$\kappa(\boldsymbol{a},\, s)^{\mathrm{T}}\,\kappa(\boldsymbol{a},\, \boldsymbol{a})^{-1}\,\kappa(\boldsymbol{a},\, z) \;\leq\; \kappa(s,\, z), \tag{B.21}$$

holds for all $s, z \in D$, which is equivalent to the condition that also the kernel, κ, from (B.14) is positive semi-definite.

B.5 Equivalent kernel: method of characteristics

In the method of characteristics (Section 6.3.1), we considered the special case where the second kernel has the additional property that it cannot correlate the regions $s > s_0$ and $s < s_0$, which can mathematically be expressed as,

$$\begin{aligned} \kappa(z,s) \;\equiv\; & \Theta(s_0 - z)\Theta(s_0 - s)\,\kappa(z,s) \\ & + \Theta(z - s_0)\Theta(s - s_0)\,\kappa(z,s). \end{aligned} \tag{B.22}$$

Using the Green's function from the radiative transfer equation, this implies for the kernel in (6.55), and assuming $z \geq b$ and $s \geq b$, that,

$$\begin{aligned} k(z,s) \;=\; & \int_b^z \mathrm{d}z' \int_b^s \mathrm{d}s' \; \kappa(z', s') \, e^{-\tau(z',z)} \, e^{-\tau(s',s)} \\ & + \; k(b,b) \, e^{-\tau(b,z)} \, e^{-\tau(b,s)}. \end{aligned} \tag{B.23}$$

Similarly, this implies, for $z \geq b$ and $s \geq b$, that,

$$g(s,z) \;=\; \int_b^z \mathrm{d}z' \; \kappa(s,z') \; e^{-\tau(z',z)}, \tag{B.24}$$

and in particular that for $s \geq b$, we have that $g(s,b) = 0$. As a result, the matrix that needs to be inverted (6.18) to obtain the approximation (6.11) simplifies to,

$$\begin{aligned} & \begin{pmatrix} \mathscr{L}_1\mathscr{L}_2 k(\boldsymbol{a},\boldsymbol{a}) + \boldsymbol{\sigma}_{\mathrm{L}}^2 & \mathscr{L}_1\mathscr{B}_2 k(\boldsymbol{a},b) \\ \mathscr{B}_1\mathscr{L}_2 k(b,\boldsymbol{a}) & \mathscr{B}_1\mathscr{B}_2 k(b,b) + \sigma_{\mathrm{B}}^2 \end{pmatrix} \\ & = \begin{pmatrix} \kappa(\boldsymbol{a},\boldsymbol{a}) + \boldsymbol{\sigma}_{\mathrm{L}}^2 & g(\boldsymbol{a},b) \\ g(\boldsymbol{a},b)^{\mathrm{T}} & k(b,b) + \sigma_{\mathrm{B}}^2 \end{pmatrix} \\ & = \begin{pmatrix} \kappa(\boldsymbol{a},\boldsymbol{a}) + \boldsymbol{\sigma}_{\mathrm{L}}^2 & \mathbf{0} \\ \mathbf{0}^{\mathrm{T}} & k(b,b) + \sigma_{\mathrm{B}}^2 \end{pmatrix}. \end{aligned} \tag{B.25}$$

Define the matrix, $\mathsf{K} \equiv \kappa(\boldsymbol{a},\boldsymbol{a}) + \boldsymbol{\sigma}_{\mathrm{L}}^2$, and suppose that there is no uncertainty on the boundary condition, i.e. $\sigma_{\mathrm{B}} = 0$. The function approximation in the dual formulation (6.11) then reads,

$$\begin{aligned} \tilde{f}_{\mathrm{dual}}(s) \;&=\; \begin{pmatrix} \boldsymbol{\eta} \\ I_0 \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} \mathsf{K} & \mathbf{0} \\ \mathbf{0}^{\mathrm{T}} & k(b,b) \end{pmatrix}^{-1} \begin{pmatrix} g(\boldsymbol{a},s) \\ k(b,s) \end{pmatrix} \\ &=\; I_0 \, e^{-\tau(b,s)} \; + \; \boldsymbol{\eta}^{\mathrm{T}} \mathsf{K}^{-1} \int_b^s \mathrm{d}s' \; \kappa(\boldsymbol{a},s') \, e^{-\tau(s',s)}. \end{aligned} \tag{B.26}$$

We clearly recognise the result from the method of characteristics (6.54). Similarly, the corresponding variance (or error measure) reads,

$$\begin{aligned}\tilde{\sigma}^2_{\text{dual}}(s) &= k(s,s) - \begin{pmatrix} g(\boldsymbol{a},s) \\ k(b,s) \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} \mathsf{K} & \mathbf{0} \\ \mathbf{0}^{\mathrm{T}} & k(b,b) \end{pmatrix}^{-1} \begin{pmatrix} g(\boldsymbol{a},s) \\ k(b,s) \end{pmatrix} \\ &= \int_b^s \mathrm{d}s' \int_b^s \mathrm{d}z'\, G(z',s)\, G(s',s) \\ &\qquad \times \Big(\kappa(s',z') - \kappa(\boldsymbol{a},s')^{\mathrm{T}} \mathsf{K}^{-1} \kappa(\boldsymbol{a},z') \Big), \end{aligned} \tag{B.27}$$

where, in the brackets, we can identify the resulting conditioned variance for the interpolation scheme (6.53) that was used for the emissivity.

B.6 Lambda operator formulation

The error measure defined in Section (6.39) can also be derived in another context. Alluding to the radiative transfer nomenclature, define an operator, Λ, that maps the data values, $\boldsymbol{y}$, to the corresponding approximation function, such that,

$$f(x) \equiv \Lambda[\boldsymbol{y}](x). \tag{B.28}$$

Note that the particular form of the Λ-operator will depend on the locations, $\boldsymbol{x}$, of the data points. Following the primal formulation, the Λ-operator can now be approximated by an operator, $\tilde{\Lambda}$, such that,

$$\tilde{f}(x) \equiv \tilde{\Lambda}[\boldsymbol{y}](x) \equiv \boldsymbol{\phi}(x)^{\mathrm{T}} \mathsf{W} \boldsymbol{y}, \tag{B.29}$$

where we expanded the operator in the set of basis functions, $\{\phi_i\}$, by defining the $(N_{\text{b}} \times N_{\text{d}})$-dimensional matrix W. Given the data locations, $\boldsymbol{x}$, we want to find W such that for any set of corresponding data values, $\boldsymbol{y}$, we have that,

$$\boldsymbol{y} = \boldsymbol{\Phi} \mathsf{W} \boldsymbol{y}. \tag{B.30}$$

Since we want this to hold for any data values, $\boldsymbol{y}$, we require that,

$$\mathbf{\Phi}\mathsf{W} = \mathbb{1}, \tag{B.31}$$

in which $\mathbb{1}$ is the $(N_\mathrm{d} \times N_\mathrm{d})$-dimensional identity matrix. Since $\mathbf{\Phi}$ is not necessarily invertible, we approximate W by minimising the mean squared error for any set of data values, $\boldsymbol{y}$, which reads,

$$\mathrm{MSE}(\mathsf{W}) \equiv \left\|\sigma^{-1}\left(\mathbf{\Phi}\mathsf{W} - \mathbb{1}\right)\boldsymbol{y}\right\|^2 + \left\|\lambda^{-1}\mathsf{W}\boldsymbol{y}\right\|^2, \tag{B.32}$$

where we weighted the different columns with the variance of the data, $\boldsymbol{\sigma}$, and added a regularisation term characterised by $\boldsymbol{\lambda}$. If W is the exact right inverse of $\mathbf{\Phi}$, the error term vanishes and only the regularisation remains. By construction the error function (B.32) can thus be interpreted as a measure of how well W approximates the (right) inverse of $\mathbf{\Phi}$, under the given regularisation. By perturbing the matrix, $\mathsf{W} \to \mathsf{W} + \delta\mathsf{W}$, the minimum of the error function can be found to be attained at,

$$\mathsf{W}_\mathrm{min} = \left(\mathbf{\Phi}^\mathrm{T}\sigma^{-2}\mathbf{\Phi} + \lambda^{-2}\right)^{-1}\mathbf{\Phi}^\mathrm{T}\sigma^{-2}, \tag{B.33}$$

which corresponds, when substituted in (B.29), to the primal solution (6.5).

A similar derivation can be made in the dual formulation which we omit here since the duality will again imply the equivalence between both results. We only note that also in the dual formulation constraints are only given for right inverses.

This raises the question whether there is a relation between W and the left inverse of $\mathbf{\Phi}$. To investigate this, we search for the approximate left inverse of $\mathbf{\Phi}$, i.e. the matrix W' that (approximately) satisfies,

$$\mathsf{W}'\mathbf{\Phi} = \mathbb{1}, \tag{B.34}$$

where $\mathbb{1}$ is now the $(N_\mathrm{b} \times N_\mathrm{b})$-dimensional identity matrix. We can again solve this by formulating it as a regularised minimisation problem. One particularly interesting mean squared error function is

$$\mathrm{MSE}'(\mathbf{W}') \equiv \left\|\lambda(\mathbf{\Phi}^{\mathrm{T}}\mathbf{W}'^{\mathrm{T}} - \mathbb{1})\boldsymbol{\phi}(x)\right\|^2 + \left\|\boldsymbol{\sigma}\mathbf{W}'^{\mathrm{T}}\boldsymbol{\phi}(x)\right\|^2, \tag{B.35}$$

in which we weighted the different rows with the regularisation vector, $\boldsymbol{\lambda}$, and added a regularisation term characterised by the variance of the data $\boldsymbol{\sigma}$. By perturbing the matrix, $\mathbf{W}' \to \mathbf{W}' + \delta\mathbf{W}'$, the minimum of the error function can be found to be attained at,

$$\mathbf{W}'_{\min} = \left(\mathbf{\Phi}\lambda^2\mathbf{\Phi}^{\mathrm{T}} + \boldsymbol{\sigma}^2\right)^{-1}\mathbf{\Phi}\lambda^2, \tag{B.36}$$

which corresponds to the dual solution (6.11). Due to the equivalence between the primal and dual solution (6.12), we can conclude that,

$$\mathbf{W}'_{\min} = \mathbf{W}^{\mathrm{T}}_{\min}. \tag{B.37}$$

As a result, in the same sense that $\mathbf{W}$ is the optimal approximate right inverse of $\mathbf{\Phi}$ with respect to the error function (B.32), we find that $\mathbf{W}^{\mathrm{T}}$ is the optimal approximate left inverse of $\mathbf{\Phi}$ with respect to the error function (B.35). Furthermore, the mean squared error function (B.35) has the interesting property that[1],

$$\mathrm{MSE}'(\mathbf{W}'_{\min}) = \tilde{\varepsilon}^2(x), \tag{B.39}$$

so we can interpret the error estimate, $\tilde{\varepsilon}^2(x)$, defined in (6.39), as a measure of how well $\mathbf{W}'$ approximates the left inverse of $\mathbf{\Phi}$ with respect to the error function (B.35).

[1]In contrast, substituting the primal and dual solutions (6.5 and 6.11) in their respective mean squared error functions (6.3 and 6.8), yields the minimal mean squared error,

$$\mathrm{MSE}(\boldsymbol{w}_{\min}) = \mathrm{MSE}(\boldsymbol{v}_{\min}) = \boldsymbol{y}^{\mathrm{T}}\left(\mathbf{\Phi}\lambda^2\mathbf{\Phi}^{\mathrm{T}} + \sigma^2\right)^{-1}\boldsymbol{y}, \tag{B.38}$$

which can less straightforwardly be related to the error estimate, $\tilde{\varepsilon}^2(x)$, than (B.39).

Bibliography

It has been seen.
It did not go unnoticed.

– after Gerard Reve, *De Avonden*

[1] R. Schlickeiser, Cosmic Ray Astrophysics, in, ser. Astronomy and Astrophysics Library, Springer Berlin Heidelberg, 2002.

[2] T. Gaisser and F. Halzen, IceCube, *Annual Review of Nuclear and Particle Science*, vol. 64, no. 1, p. 101, 2014.

[3] B. P. Abbott, R. Abbott, T. D. Abbott, M. R. Abernathy, F. Acernese *et al.*, GW151226: Observation of Gravitational Waves from a 22-Solar-Mass Binary Black Hole Coalescence, *Physical Review Letters*, vol. 116, no. 24, p. 241103, 2016.

[4] M. G. Taylor, N. Altobelli, B. J. Buratti and M. Choukroun, *The Rosetta mission orbiter science overview: The comet phase*, 2017.

[5] J. C. Maxwell, VIII. A dynamical theory of the electromagnetic field, *Philosophical Transactions of the Royal Society of London*, vol. 155, p. 459, 1865.

[6] M. Planck, Ueber irreversible Strahlungsvorgänge, *Annalen der Physik*, vol. 306, no. 1, p. 69, 1900.

[7] M. Planck, Entropie und Temperatur strahlender Wärme, *Annalen der Physik*, vol. 306, no. 4, p. 719, 1900.

[8] M. Planck, Ueber das Gesetz der Energieverteilung im Normalspectrum, *Annalen der Physik*, vol. 309, no. 3, p. 553, 1901.

[9] A. Einstein, Über einen die Erzeugung und Verwandlung des Lichtes betreffenden heuristischen Gesichtspunkt, *Annalen der Physik*, vol. 322, no. 6, p. 132, 1905.

[10] S. Tomonaga, On a Relativistically Invariant Formulation of the Quantum Theory of Wave Fields, *Progress of Theoretical Physics*, vol. 1, no. 2, p. 27, 1946.

[11] J. Schwinger, On Quantum-Electrodynamics and the Magnetic Moment of the Electron, *Physical Review*, vol. 73, no. 4, p. 416, 1948.

[12] R. P. Feynman, Space-time approach to quantum electrodynamics, *Physical Review*, vol. 76, no. 6, p. 769, 1949.

[13] M. Gardner, Mathematical Games, *Scientific American*, vol. 223, no. 4, p. 120, 1970.

[14] A. Schuster, Radiation Through a Foggy Atmosphere, *The Astrophysical Journal*, vol. 21, p. 1, 1905.

[15] K. Schwarzschild, On the equilibrium of the Sun's atmosphere, in *Nachrichten von der Königlichen Gesellschaft der Wissenschaften zu Göttingen. Math.-phys. Klasse*, vol. 195, 1906, p. 41.

[16] A. S. Eddington, On the Radiative Equilibrium of the Stars, *Monthly Notices of the Royal Astronomical Society*, vol. 77, no. 1, p. 16, 1916.

[17] J. H. Jeans, The Equations of Radiative Transfer of Energy, *Monthly Notices of the Royal Astronomical Society*, vol. 78, no. 1, p. 28, 1917.

[18] S. Chandrasekhar, *Radiative Transfer*. Dover Publications, 1960.

[19] G. Kanschat, E. Meinköhn, R. Rannacher, R. Wehrse, W. von Waldenfels *et al.*, *Numerical Methods in Multidimensional Radiative Transfer*, G. Kanschat, E. Meinköhn, R. Rannacher and R. Wehrse, Eds. Springer Berlin Heidelberg, 2009.

[20] S. M. Carroll, *Spacetime and Geometry*. Cambridge University Press, 2019.

[21] L. Decin, Evolution and Mass Loss of Cool Aging Stars: A Daedalean Story, *Annual Review of Astronomy and Astrophysics*, vol. 59, no. 1, 2021.

[22] P. J. Huggins and A. E. Glassgold, The photoproduction of circumstellar OH maser shells, *The Astronomical Journal*, vol. 87, p. 1828, 1982.

[23] K. I. Öberg, *Photochemistry and Astrochemistry: Photochemical Pathways to Interstellar Complex Organic Molecules*, 2016.

[24] M. Van de Sande and T. J. Millar, The Role of Internal Photons on the Chemistry of the Circumstellar Envelopes of AGB Stars, *The Astrophysical Journal*, vol. 873, no. 1, p. 36, 2019.

[25] M. Van de Sande, C. Walsh and T. J. Millar, Chemical modelling of dust–gas chemistry within AGB outflows – III. Photoprocessing of the ice and return to the ISM, *Monthly Notices of the Royal Astronomical Society*, vol. 501, no. 1, p. 491, 2020.

[26] I. Hubeny and D. Mihalas, *Theory of Stellar Atmospheres*. Princeton University Press, 2014.

[27] D. Mihalas and B. Weibel-Mihalas, *Foundations of Radiation Hydrodynamics*. Oxford University Press, 1984.

[28] F. De Ceuster, W. Homan, J. Yates, L. Decin, P. Boyle *et al.*, Magritte, a modern software library for 3D radiative transfer: I. Non-LTE atomic and molecular line modelling, *Monthly Notices of the Royal Astronomical Society*, vol. 492, no. 2, p. 1812, 2019.

[29] L. Boltzmann, Weitere studien über das wärmegleichgewicht unter gasmolekülen, *Sitzungs. Akad. Wiss. Wein*, vol. 66, p. 275, 1872.

[30] C. J. Cannon, A general formulation of the transfer equation. I. Anisotropic scattering, *Australian Journal of Physics, vol. 24, p.341*, vol. 24, p. 341, 1971.

[31] C. J. Cannon, A General Formulation of the Transfer Equation. II. Line Formation with General Redistribution, *Australian Journal of Physics*, vol. 25, no. 2, p. 177, 1972.

[32] S. N. Shore, Blue sky and hot piles: The evolution of radiative transfer theory from atmospheres to nuclear reactors, *Historia Mathematica*, vol. 29, no. 4, p. 463, 2002.

[33] P. Feautrier, Sur la resolution numerique de l'equation de transfert., *Comptes Rendus Academie des Sciences (serie non specifiee)*, vol. 258, p. 3189, 1964.

[34] U. M. Noebauer and S. A. Sim, Monte Carlo radiative transfer, *Living Reviews in Computational Astrophysics*, vol. 5, no. 1, p. 1, 2019.

[35] C. P. Dullemond, A. Juhasz, A. Pohl, F. Sereshti, R. Shetty *et al.*, *RADMC-3D: A multi-purpose radiative transfer tool*, Astrophysics Source Code Library, 2012.

[36] S. Verstocken, D. Van De Putte, P. Camps and M. Baes, SKIRT: Hybrid parallelization of radiative transfer simulations, *Astronomy and Computing*, vol. 20, p. 16, 2017.

[37] B. Vandenbroucke and K. Wood, The Monte Carlo photoionization and moving-mesh radiation hydrodynamics code CMacIonize, *Astronomy and Computing*, vol. 23, p. 40, 2018.

[38] T. J. Harries, T. J. Haworth, D. Acreman, A. Ali and T. Douglas, The TORUS radiation transfer code, *Astronomy and Computing*, vol. 27, p. 63, 2019.

[39] F. Yusef-Zadeh, M. Morris and R. L. White, Bipolar reflection nebulae - Monte Carlo simulations, *The Astrophysical Journal*, vol. 278, p. 186, 1984.

[40] P. Camps and M. Baes, The Failure of Monte Carlo Radiative Transfer at Medium to High Optical Depths, *The Astrophysical Journal*, vol. 861, no. 2, p. 80, 2018.

[41] G. Altay, R. A. Croft and I. Pelupessy, Sphray: A smoothed particle hydrodynamics ray tracer for radiative transfer, *Monthly Notices of the Royal Astronomical Society*, vol. 386, no. 4, p. 1931, 2008.

[42] T. G. Bisbas, T. A. Bell, S. Viti, J. Yates and M. J. Barlow, 3d-pdr : a new three-dimensional astrochemistry code for treating photodissociation regions, *Monthly Notices of the Royal Astronomical Society*, vol. 427, no. 3, p. 2100, 2012.

[43] T. Frostholm, T. Haugbølle and T. Grassi, Lampray: Multi-group long characteristics ray tracing for adaptive mesh radiation hydrodynamics, *arXiv 1809.05541*, 2018.

[44] P. G. Dykema, R. I. Klein and J. I. Castor, A New Scheme for Multidimensional Line Transfer. III. A Two-dimensional Lagrangian Variable Tensor Method with Discontinuous Finite-Element SN Transport, *The Astrophysical Journal*, vol. 457, p. 892, 1996.

[45] S. Richling, E. Meinköhn, N. Kryzhevoi and G. Kanschat, Radiative transfer with finite elements: I. Basic method and tests, *Astronomy and Astrophysics*, vol. 380, no. 2, p. 776, 2001.

[46] M. Badri, P. Jolivet, B. Rousseau and Y. Favennec, High performance computation of radiative transfer equation using the finite element method, *Journal of Computational Physics*, vol. 360, p. 74, 2018.

[47] M. R. Hogerheijde and F. F. S. van der Tak, An accelerated Monte Carlo method to solve two-dimensional radiative transfer and molecular excitation, *Astronomy & Astrophysics*, vol. 64, no. 1, p. 137, 2000.

[48] C. Brinch and M. R. Hogerheijde, LIME: a flexible, non-LTE line excitation and radiation transfer method for millimeter and far-infrared wavelengths, *Astronomy & Astrophysics*, vol. 523, no. 15333, p. A25, 2010.

[49] A. S. Booth, C. Walsh and J. D. Ilee, First detections of H13CO+ and HC15N in the disk around HD 97048, *Astronomy & Astrophysics*, vol. 629, p. A75, 2019.

[50] M. Montargès, W. Homan, D. Keller, N. Clementel, S. Shetye *et al.*, NOEMA maps the CO J = 2 - 1 environment of the red supergiant mu Cep, *Monthly Notices of the Royal Astronomical Society*, vol. 485, no. 2, p. 2417, 2019.

[51] W. Homan, T. Danilovich, L. Decin, A. De Koter, J. Nuth *et al.*, ALMA detection of a tentative nearly edge-on rotating disk around the nearby AGB star R Doradus, *Astronomy and Astrophysics*, vol. 614, p. A113, 2018.

[52] M. G. Evans, T. W. Hartquist, P. Caselli, A. C. Boley, J. D. Ilee *et al.*, Gravitational instabilities in a protosolar-like disc III: molecular line detection and sensitivities, *Monthly Notices of the Royal Astronomical Society*, 2018.

[53] C. Walsh, R. A. Loomis, K. I. Öberg, M. Kama, M. L. R. van 't Hoff *et al.*, First detection of gas-phase methanol in a protoplanetary disk, *The Astrophysical Journal*, vol. 823, no. 1, p. L10, 2016.

[54] E. A. Bergin, L. I. Cleeves, U. Gorti, K. Zhang, G. A. Blake *et al.*, An old disk still capable of forming a planetary system, *Nature*, vol. 493, no. 7434, p. 644, 2013.

[55] M. Maercker, S. Mohamed, W. H. Vlemmings, S. Ramstedt, M. A. Groenewegen *et al.*, Unexpectedly large mass loss during the thermal pulse cycle of the red giant star R Sculptoris, *Nature*, vol. 490, no. 7419, p. 232, 2012.

[56] S. M. Andrews, D. J. Wilner, A. M. Hughes, C. Qi, K. A. Rosenfeld *et al.*, The TW Hya disk a 870 μm: Comparison of CO and dust radial structures, *The Astrophysical Journal*, vol. 744, no. 2, p. 162, 2012.

[57] J. L. Hennessy and D. A. Patterson, *Computer Architecture: A Quantitive Approach*, Sixth Edit. Morgan KaufMann Publishers, 2017.

[58] G. M. Moore, Cramming more components onto integrated circuits With unit cost, *Electronics*, vol. 38, no. 8, p. 114, 1965.

[59] R. Dennard, F. Gaensslen, H.-N. Yu, V. Rideout, E. Bassous *et al.*, Design of ion-implanted MOSFET's with very small physical dimensions, *IEEE Journal of Solid-State Circuits*, vol. 9, no. 5, p. 256, 1974.

[60] G. E. Moore, Progress in Digital Integrated Electronics, in *International Electron Devices Meeting, IEEE*, 1975, p. 11.

[61] L. J. Flynn, Intel Halts Development Of 2 New Microprocessors, *The New York Times*, 2004.

[62] H. Carter Edwards, C. R. Trott and D. Sunderland, Kokkos: Enabling manycore performance portability through polymorphic memory access patterns, *Journal of Parallel and Distributed Computing*, vol. 74, no. 12, p. 3202, 2014.

[63] D. A. Beckingsale, T. R. Scogland, J. Burmark, R. Hornung, H. Jones *et al.*, RAJA: Portable Performance for Large-Scale Scientific Applications, in *2019 IEEE/ACM International Workshop on Performance, Portability and Productivity in HPC (P3HPC)*, IEEE, 2019, p. 71.

[64] J. Reinders, B. Ashbaugh, J. Brodman, M. Kinsner, J. Pennycook *et al.*, *Data Parallel C++*. Berkeley, CA: Apress, 2021.

[65] T. Deakin, *Which performance portable programming model should I use? (http://uob-hpc.github.io/2020/05/05/choosing-models.html)*, 2020.

[66] MPI: A message passing interface, in *Proceedings of the Supercomputing Conference*, 1993, p. 878.

[67] G. Guennebaud, B. Jacob *et al.*, *Eigen*, 2010.

[68] P. A. Boyle, G. Cossu, A. Yamaguchi and A. Portelli, Grid: A next generation data parallel C++ QCD library, in *Proceedings of The 33rd International Symposium on Lattice Field Theory — PoS(LATTICE 2015)*, vol. 14-18-July, Trieste, Italy: Sissa Medialab, 2016, p. 023.

[69] NVIDIA, P. Vingelmann and F. H. P. Fitzek, *CUDA*, 2020.

[70] A. Karpathy, *Software 2.0 (https://karpathy.medium.com/software-2-0-a64152b37c35)*, 2017.

[71] J. Sacks, W. J. Welch, T. J. Mitchell and H. P. Wynn, Design and Analysis of Computer Experiments, *Statistical Science*, vol. 4, no. 4, p. 409, 1989.

[72] D. De Mijolla, S. Viti, J. Holdship, I. Manolopoulou and J. Yates, Incorporating astrochemistry into molecular line modelling via emulation, *Astronomy and Astrophysics*, vol. 630, p. A117, 2019.

[73] M. F. Kasim, D. Watson-Parris, L. Deaconu, S. Oliver, P. Hatfield *et al.*, Building high accuracy emulators for scientific simulations with deep neural architecture search, *arXiv 2001.08055*, 2020.

[74] J. Holdship, S. Viti, T. J. Haworth and J. D. Ilee, Chemulator: Fast, accurate thermochemistry for dynamical models through emulation, 2021.

[75] I. E. Lagaris, A. Likas and D. I. Fotiadis, Artificial neural networks for solving ordinary and partial differential equations, *IEEE Transactions on Neural Networks*, vol. 9, no. 5, p. 987, 1998.

[76] I. E. Lagaris, A. C. Likas and D. G. Papageorgiou, Neural-network methods for boundary value problems with irregular boundaries, *IEEE Transactions on Neural Networks*, vol. 11, no. 5, p. 1041, 2000.

[77] M. Raissi, P. Perdikaris and G. E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, *Journal of Computational Physics*, vol. 378, p. 686, 2019.

[78] S. Mishra and R. Molinaro, Physics informed neural networks for simulating radiative transfer, *Journal of Quantitative Spectroscopy and Radiative Transfer*, vol. 270, p. 107705, 2021.

[79] F. De Ceuster, J. Yates, P. Boyle, L. Decin and J. Hetherington, Magritte: a new multidimensional accelerated general-purpose radiative transfer code, *Proceedings of the International Astronomical Union*, vol. 14, no. S343, p. 381, 2018.

[80] F. De Ceuster, J. Bolte, W. Homan, S. Maes, J. Malfait *et al.*, Magritte, a modern software library for 3D radiative transfer: II. Adaptive ray-tracing, mesh construction, and reduction, *Monthly Notices of the Royal Astronomical Society*, vol. 499, no. 4, p. 5194, 2020.

[81] N. Bohr, On the constitution of atoms and molecules, *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 26, no. 151, p. 1, 1913.

[82] C. H. Payne, Stellar Atmospheres; a Contribution to the Observational Study of High Temperature in the Reversing Layers of Stars., PhD thesis, Radcliffe College, Harvard, 1925.

[83] A. E. Douglas and G. Herzberg, Note on CH+ in Interstellar Space and in the Laboratory., *The Astrophysical Journal*, vol. 94, p. 381, 1941.

[84] S. Weinreb, A. H. Barrett, M. L. Meeks and J. C. Henry, Radio observations of OH in the interstellar medium, *Nature*, vol. 200, no. 4909, p. 829, 1963.

[85] B. T. Draine, *Physics of the Insterstellar and Intergalactic Medium*. Princeton University Press, 2010.

[86] D. Mihalas and R. G. Athay, The Effects of Departures from LTE in Stellar Spectra, *Annual Review of Astronomy and Astrophysics*, vol. 11, no. 1, p. 187, 1973.

[87] F. L. Schöier, F. F. S. van der Tak, E. F. van Dishoeck and J. H. Black, An atomic and molecular database for analysis of submillimetre line observations, *Astronomy & Astrophysics*, vol. 432, no. 1, p. 369, 2005.

[88] I. Hubeny, Accelerated Lambda Iteration, in *Stellar Atmosphere Modeling, ASP Conference Proceedings*, I. Hubeny, D. Mihalas and K. Werner, Eds., Tuebingen: Springer Berlin Heidelberg, 2003, p. 375.

[89] C. Cannon, Angular quadrature perturbations in radiative transfer theory, *Journal of Quantitative Spectroscopy and Radiative Transfer*, vol. 13, no. 7, p. 627, 1973.

[90] C. J. Cannon, Frequency-Quadrature Perturbations in Radiative-Transfer Theory, *The Astrophysical Journal*, vol. 185, no. 7, p. 621, 1973.

[91] G. B. Rybicki and D. G. Hummer, An accelerated lambda iteration method for multilevel radiative transfer. I - Non-overlapping lines with background continuum, *Astronomy & Astrophysics*, vol. 245, p. 171, 1991.

[92] G. L. Olson, L. H. Auer and J. R. Buchler, A rapidly convergent iterative solution of the non-LTE line radiation transfer problem, *Journal of Quantitative Spectroscopy and Radiative Transfer*, vol. 35, no. 6, p. 431, 1986.

[93] G.-J. van Zadelhoff, C. P. Dullemond, F. F. S. van der Tak, J. A. Yates, S. D. Doty *et al.*, Numerical methods for non-LTE line radiative transfer: Performance and convergence characteristics, *Astronomy & Astrophysics*, vol. 395, no. 1, p. 373, 2002.

[94] P. H. Hauschildt, H. Störzer and E. Baron, Convergence properties of the accelerated Λ-iteration method for the solution of radiative transfer problems, *Journal of Quantitative Spectroscopy and Radiative Transfer*, vol. 51, no. 6, p. 875, 1994.

[95] K. C. Ng, Hypernetted chain solutions for the classical one-component plasma up to $\Gamma = 7000$, *The Journal of Chemical Physics*, vol. 61, no. 7, p. 2680, 1974.

[96] J. R. Buchler and L. H. Auer, Iterative Solution to the Non-LTE Line Transport Problem, in *Radiative Properties of Hot Dense Matter*, J. Davis, C. Hooper, R. Lee, A. Merts and B. Rozsnyai, Eds., Sarasota, Florida USA: World Scientific Publishing Company, 1985, p. 58.

[97] D. G. Anderson, Iterative Procedures for Nonlinear Integral Equations, *Journal of the ACM (JACM)*, vol. 12, no. 4, p. 547, 1965.

[98] D. Scieur, A. D'Aspremont and F. Bach, Regularized nonlinear acceleration, *Mathematical Programming*, vol. 179, no. 1-2, p. 47, 2020.

[99] W. Karush, *Minima of Functions of Several Variables with Inequalities as Side Constraints*, 1939.

[100] H. W. Kuhn and A. W. Tucker, Nonlinear programming, in *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, California, USA: University of California Press, 1951, p. 481.

[101] C. M. J. Osborne and I. Milić, The Lightweaver Framework for NLTE Radiative Transfer in Python, 2021.

[102] J. Wenzel, J. Rhinelander and D. Moldovan, *pybind11: Seamless operability between C++11 and Python*, 2017.

[103] K. Beck, *Test-driven development: by example*. Addison-Wesley Professional, 2002, p. 220.

[104] A. Nanthaamornphong and J. C. Carver, Test-Driven Development in HPC Science: A Case Study, *Computing in Science & Engineering*, vol. 20, no. 5, p. 98, 2018.

[105] T.-R. Teschner, A practical guide towards agile test-driven development for scientific software projects, 2020.

[106] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier *et al.*, Jupyter Notebooks—a publishing format for reproducible computational workflows, in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, F. Loizides and B. Schmidt, Eds., vol. 16, IOS Press, 2016, p. 87.

[107] P. Duvall, S. Matyas, A. Glover and S. M. Matyas, *Continuous integration: improving software quality and reducing risk*. Addison-Wesley, 2007.

[108] L. Auer, Improved Boundary Conditions for the Feautrier Method, *The Astrophysical Journal*, vol. 150, no. October, p. L53, 1967.

[109] L. Auer, An Hermitian method for the solution of radiative transfer problems, *Journal of Quantitative Spectroscopy and Radiative Transfer*, vol. 16, no. 11, p. 931, 1976.

[110] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy *et al.*, SciPy 1.0: fundamental algorithms for scientific computing in Python, *Nature Methods*, vol. 17, no. 3, p. 261, 2020.

[111] F. H. Shu, Self-similar collapse of isothermal spheres and star formation, *The Astrophysical Journal*, vol. 214, p. 488, 1977.

[112] D. Rundle, T. J. Harries, D. M. Acreman and M. R. Bate, Three-dimensional molecular line transfer: A simulated star-forming region, *Monthly Notices of the Royal Astronomical Society*, vol. 407, no. 2, p. 986, 2010.

[113] L. Decin, M. Montargès, A. M. S. Richards, C. A. Gottlieb, W. Homan *et al.*, (Sub)stellar companions shape the winds of evolved stars, *Science*, vol. 369, no. 6510, p. 1497, 2020.

[114] I. El Mellah, J. Bolte, L. Decin, W. Homan and R. Keppens, Wind morphology around cool evolved stars in binaries: The case of slowly accelerating oxygen-rich outflows, *Astronomy and Astrophysics*, vol. 637, p. A91, 2020.

[115] S. Maes, W. Homan, J. Malfait, L. Siess, J. Bolte *et al.*, SPH modelling of companion-perturbed AGB outflows including a new morphology classification scheme, *Astronomy & Astrophysics*, vol. 653, p. A25, 2021.

[116] J. Malfait, W. Homan, S. Maes, J. Bolte, L. Siess *et al.*, SPH modelling of wind-companion interactions in eccentric AGB binary systems, *Astronomy & Astrophysics*, vol. 652, p. A51, 2021.

[117] T. J. Haworth, S. C. Glover, C. M. Koepferl, T. G. Bisbas and J. E. Dale, Synthetic observations of star formation and the interstellar medium, *New Astronomy Reviews*, vol. 82, p. 1, 2018.

[118] C. Xia, J. Teunissen, I. E. Mellah, E. Chané and R. Keppens, MPI-AMRVAC 2.0 for Solar and Astrophysical Applications, *The Astrophysical Journal Supplement Series*, vol. 234, no. 2, p. 30, 2018.

[119] J. Bolte, in preparation,

[120] D. J. Price, J. Wurster, T. S. Tricco, C. Nixon, S. Toupin *et al.*, Phantom: A Smoothed Particle Hydrodynamics and Magnetohydrodynamics Code for Astrophysics, *Publications of the Astronomical Society of Australia*, vol. 35, p. e031, 2018.

[121] W. Homan, L. Decin, A. de Koter, A. J. van Marle, R. Lombaert *et al.*, Simplified models of stellar wind anatomy for interpreting high-resolution data, *Astronomy & Astrophysics*, vol. 579, p. A118, 2015.

[122] W. Homan, J. Boulangier, L. Decin and A. de Koter, Simplified models of circumstellar morphologies for interpreting high-resolution data, *Astronomy & Astrophysics*, vol. 596, p. A91, 2016.

[123] N. Harada, T. A. Thompson and E. Herbst, Modeling the molecular composition in an active galactic nucleus disk, *The Astrophysical Journal*, vol. 765, no. 2, p. 108, 2013.

[124] M. R. Hogerheijde, E. A. Bergin, C. Brinch, L. I. Cleeves, J. K. Fogel *et al.*, Detection of the water reservoir in a forming planetary system, *Science*, vol. 334, no. 6054, p. 338, 2011.

[125] K. I. Öberg, V. V. Guzmán, K. Furuya, C. Qi, Y. Aikawa *et al.*, The comet-like composition of a protoplanetary disk as revealed by complex cyanides, *Nature*, vol. 520, no. 7546, p. 198, 2015.

[126] P. K. Romano, N. E. Horelik, B. R. Herman, A. G. Nelson, B. Forget *et al.*, OpenMC: A state-of-the-art Monte Carlo code for research and development, *Annals of Nuclear Energy*, vol. 82, p. 90, 2015.

[127] E. Brun, F. Damian, C. M. Diop, E. Dumonteil, F. X. Hugot *et al.*, Tripoli-4®, CEA, EDF and AREVA reference Monte Carlo code, *Annals of Nuclear Energy*, vol. 82, p. 151, 2015.

[128] C. J. Werner, J. S. Bull, C. J. J. Solomon, F. B. Brown, G. W. Mckinney *et al.*, 'MCNP Version 6.2 Release Notes', Tech. Rep., 2018, p. 1.

[129] S. Glassner Andrew, E. Haines, P. Hanrahan, R. L. Cook, J. Avro *et al.*, *An introduction to ray tracing*, A. S. Glassner, Ed. London, UK: Academic Press Ltd., 1989, p. 327.

[130] M. J. Berger and P. Colella, Local adaptive mesh refinement for shock hydrodynamics, *Journal of Computational Physics*, vol. 82, no. 1, p. 64, 1989.

[131] V. Springel, E pur si muove: Galilean-invariant cosmological hydrodynamical simulations on a moving mesh, *Monthly Notices of the Royal Astronomical Society*, vol. 401, no. 2, p. 791, 2010.

[132] L. B. Lucy, A numerical approach to the testing of the fission hypothesis, *The Astronomical Journal*, vol. 82, p. 1013, 1977.

[133] R. A. Gingold and J. J. Monaghan, Smoothed particle hydrodynamics: theory and application to non-spherical stars, *Monthly Notices of the Royal Astronomical Society*, vol. 181, no. 3, p. 375, 1977.

[134] J. Ritzerveld and V. Icke, Transport on adaptive random lattices, *Physical Review E*, vol. 74, no. 2, p. 026704, 2006.

[135] P. Camps, M. Baes and W. Saftly, Using 3D Voronoi grids in radiative transfer simulations, *Astronomy & Astrophysics*, vol. 560, p. A35, 2013.

[136] W. Saftly, P. Camps, M. Baes, K. D. Gordon, S. Vandewoude *et al.*, Using hierarchical octrees in Monte Carlo radiative transfer simulations, *Astronomy and Astrophysics*, vol. 554, p. A10, 2013.

[137] W. Saftly, M. Baes and P. Camps, Hierarchical octree and k -d tree grids for 3D radiative transfer simulations, *Astronomy & Astrophysics*, vol. 561, p. A77, 2014.

[138] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions*. New York: Dover Publications, 1972.

[139] K. M. Gorski, E. Hivon, A. J. Banday, B. D. Wandelt, F. K. Hansen *et al.*, HEALPix: A Framework for High-Resolution Discretization and Fast Analysis of Data Distributed on the Sphere, *The Astrophysical Journal*, vol. 622, no. 2, p. 759, 2005.

[140] T. Abel and B. D. Wandelt, Adaptive ray tracing for radiative transfer around point sources, *Monthly Notices of the Royal Astronomical Society*, vol. 330, no. 3, p. L53, 2002.

[141] P. Fernique, M. G. Allen, T. Boch, A. Oberto, F. X. Pineau *et al.*, Hierarchical progressive surveys: Multi-resolution HEALPix data structures for astronomical images, catalogues, and 3-dimensional data cubes, *Astronomy and Astrophysics*, vol. 578, p. A114, 2015.

[142] R. W. Youngren and M. D. Petty, A multi-resolution HEALPix data structure for spherically mapped point data, *Heliyon*, vol. 3, no. 6, p. e00332, 2017.

[143] F. J. Thompson, K. B. Soni and P. N. Weatherill, *Handbook of Grid Generation*, N. Weatherill, B. Soni and J. Thompson, Eds., 1997. CRC Press, 1998, p. 1096.

[144] P.-L. George and P. J. Frey, *Mesh Generation*. ISTE, Wiley, 2008.

[145] C. Geuzaine and J. F. Remacle, Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities, *International Journal for Numerical Methods in Engineering*, vol. 79, no. 11, p. 1309, 2009.

[146] C. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[147] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[148] K. J. Bathe, *Finite Element Procedures*, Second Edi. 2014.

[149] A. Shah, A. G. Wilson and Z. Ghahramani, Student-t processes as alternatives to Gaussian processes, in *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics (AISTATS)*, vol. 33, Reykjavik, 2014, p. 877.

[150] W. L. Oberkampf and C. J. Roy, *Verification and Validation in Scientific Computing*. Cambridge University Press, 2010.

[151] P. Hennig, M. A. Osborne and M. Girolami, Probabilistic numerics and uncertainty in computations, *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 471, no. 2179, p. 20150142, 2015.

[152] J. Cockayne, C. J. Oates, T. J. Sullivan and M. Girolami, Bayesian Probabilistic Numerical Methods, *SIAM Review*, vol. 61, no. 3, p. 756, 2019.

[153] C. J. Oates and T. J. Sullivan, A modern retrospective on probabilistic numerics, *Statistics and Computing*, vol. 29, no. 6, p. 1335, 2019.

[154] H. Poincaré, *Calcul des probabilités*. Georges Carré, 1896.

[155] P. Diaconis, Bayesian Numerical Analysis, in *Statistical Decision Theory and Related Topics IV*, S. S. Gupta and J. O. Berger, Eds., Springer New York, 1988, p. 163.

[156] P. R. Conrad, M. Girolami, S. Särkkä, A. Stuart and K. Zygalakis, Statistical analysis of differential equations: introducing probability measures on numerical solutions, *Statistics and Computing*, vol. 27, no. 4, p. 1065, 2017.

[157] J. Cockayne, C. Oates, T. Sullivan and M. Girolami, Probabilistic numerical methods for PDE-constrained Bayesian inverse problems, in *AIP Conference Proceedings*, vol. 1853, American Institute of Physics Inc., 2017, p. 060001.

[158] H. Owhadi and C. Scovel, *Operator-Adapted Wavelets, Fast Solvers, and Numerical Homogenization*. Cambridge University Press, 2019.

[159] B. Fornberg and N. Flyer, Solving PDEs with radial basis functions, *Acta Numerica*, vol. 24, p. 215, 2015.

[160] R. Schaback, Error estimates and condition numbers for radial basis function interpolation, *Advances in Computational Mathematics*, vol. 3, no. 3, p. 251, 1995.

[161] R. Schaback, On the efficiency of interpolation by radial basis functions, *Proceedings of Chamonix*, A. Le Mehaute, C. Rabut and L. L. Schumaker, Eds., p. 309, 1997.

[162] Z. Zhang, Y. Xu, J. Yang, X. Li and D. Zhang, A Survey of Sparse Representation: Algorithms and Applications, *IEEE Access*, vol. 3, p. 490, 2015.

[163] F. Schäfer, T. J. Sullivan and H. Owhadi, Compression, Inversion, and Approximate PCA of Dense Kernel Matrices at Near-Linear Computational Complexity, *Multiscale Modeling & Simulation*, vol. 19, no. 2, p. 688, 2021.

[164] T. Grätsch and K.-J. Bathe, A posteriori error estimation techniques in practical finite element analysis, *Computers & Structures*, vol. 83, no. 4-5, p. 235, 2005.

[165] R. Verfürth, *A Posteriori Error Estimation Techniques for Finite Element Methods*. Oxford University Press, 2013.

[166] K. Hornik, M. Stinchcombe and H. White, Multilayer feedforward networks are universal approximators, *Neural Networks*, vol. 2, no. 5, p. 359, 1989.

[167] J. Park and I. W. Sandberg, Universal Approximation Using Radial-Basis-Function Networks, *Neural Computation*, vol. 3, no. 2, p. 246, 1991.

[168] D. X. Zhou, Universality of deep convolutional neural networks, *Applied and Computational Harmonic Analysis*, vol. 48, no. 2, p. 787, 2020.

[169] S. Mishra and R. Molinaro, Estimates on the generalization error of physics-informed neural networks for approximating a class of inverse problems for PDEs, *IMA Journal of Numerical Analysis*, 2021.

[170] T. Elsken, J. H. Metzen and F. Hutter, Neural architecture search: A survey, *Journal of Machine Learning Research*, vol. 20, no. 55, p. 1, 2019.

[171] Y. Li and L. Zikatanov, A posteriori error estimates of finite element methods by preconditioning, *Computers & Mathematics with Applications*, vol. 91, p. 192, 2021.

[172] P. L. Palmer, The deprojection of axisymmetric galaxies, *Monthly Notices of the Royal Astronomical Society*, vol. 266, no. 3, p. 697, 1994.

[173] M. Bremer, A 3D iterative deprojection technique. I. Development of the algorithm and tests., *Astronomy and Astrophysics Supplement Series*, vol. 112, p. 551, 1995.

[174] O. Gerhard and J. Binney, On the deprojection of axisymmetric bodies, *Monthly Notices of the Royal Astronomical Society*, vol. 279, no. 3, p. 993, 1996.

[175] G. B. Rybicki, Deprojection of Galaxies: How much can be Learned?, in *Structure and Dynamics of Elliptical Galaxies*, vol. 127, Springer Netherlands, 1987, p. 397.

[176] G. Balakrishnan, A. Dalca, A. Zhao, J. Guttag, F. Durand *et al.*, Visual Deprojection: Probabilistic Recovery of Collapsed Dimensions, in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, vol. 2019-Octob, IEEE, 2019, p. 171.

[177] O. James, E. von Tunzelmann, P. Franklin and K. S. Thorne, Gravitational lensing by spinning black holes in astrophysics, and in the movie Interstellar, *Classical and Quantum Gravity*, vol. 32, no. 6, p. 065001, 2015.

[178] A. Peraiah, *An Introduction to Radiative Transfer*. Cambridge University Press, 2001.