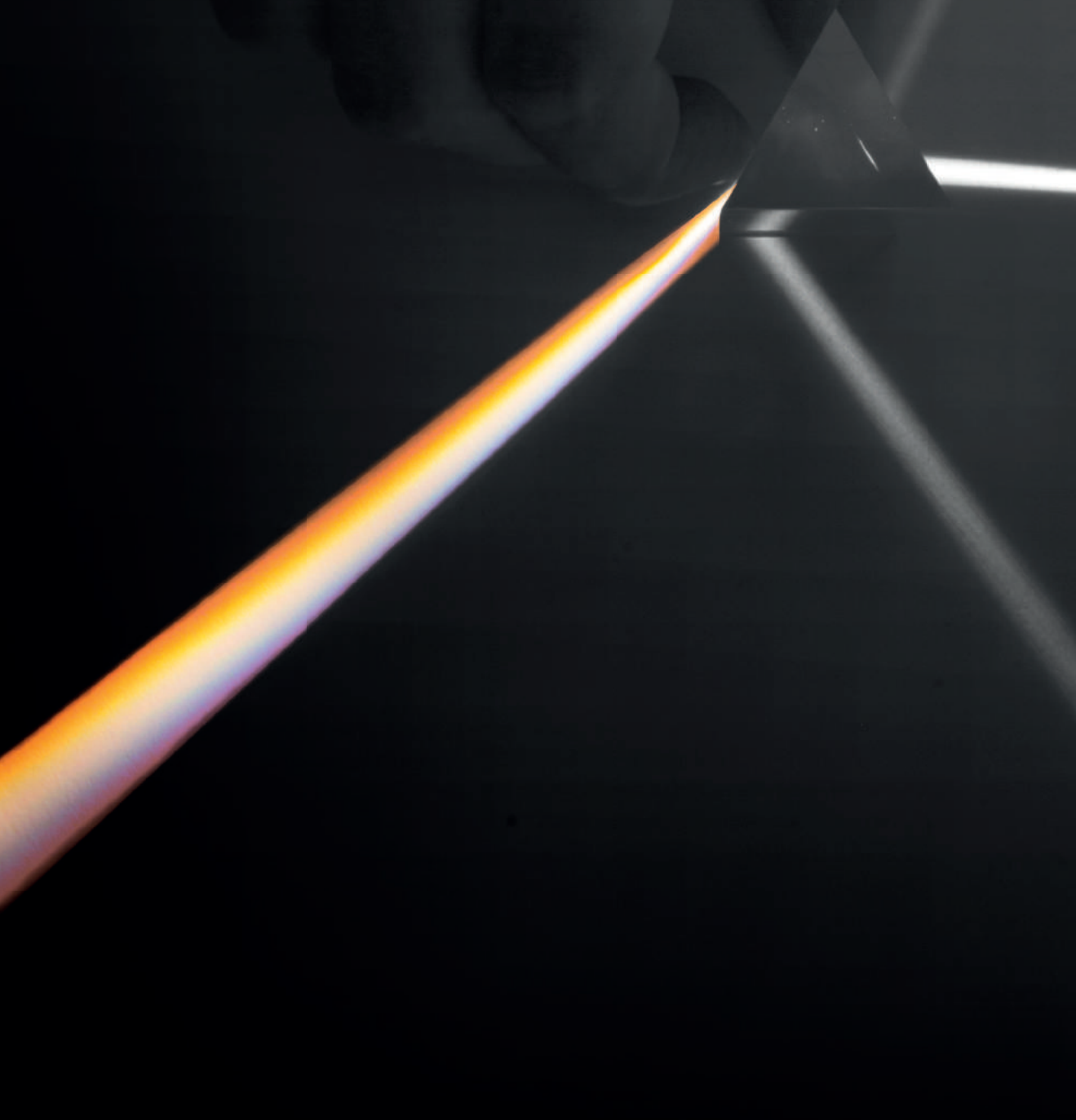
Principles of computational illumination optics



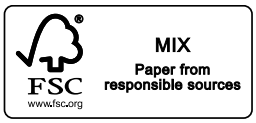
PRINCIPLES OF COMPUTATIONAL ILLUMINATION OPTICS

B.S. van Lith

Principles of computational illumination optics

Cover design by Naomi Jansen.

Printed by Gildeprint, Enschede, The Netherlands.



This research was performed within the framework of the strategic joint research program on Intelligent Lighting between TU/e and Philips Lighting B.V.

A catalogue record is available from the Eindhoven University of Technology Library.

ISBN: 978-94-6233-812-8

Principles of computational illumination optics

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Technische Universiteit Eindhoven, op gezag van de rector magnificus prof. dr. ir. F.P.T. Baaijens, voor een commissie aangewezen door het College voor Promoties, in het openbaar te verdedigen op woensdag 13 december 2017 om 14:00 uur

door

Bart Serapion van Lith

geboren te Berlicum

Dit proefschrift is goedgekeurd door de promotoren en de samenstelling van de promotiecommissie is als volgt:

voorzitter:	prof. dr. ir. O.J. Boxma
1e promotor:	prof. dr. ir. W.L. IJzerman
copromotor:	dr. ir. J.H.M. ten Thije Boonkkamp
leden:	prof. dr. H.P. Urbach (TUD)
	prof. dr. ir. J.J.W. van der Vegt (UTwente)
	dr. ir. S. M. B. Bäumer (TNO)
	prof. dr. ir. B. Koren
	prof. dr. M.A. Peletier

Het onderzoek of ontwerp dat in dit proefschrift wordt beschreven is uitgevoerd in overeenstemming met de TU/e Gedragscode Wetenschapsbeoefening.

Foreword

Little did he know...

Litarary cliché

Welcome, dear reader, to my thesis. The title hasn't been subject to a lot of change, even though I suggested three: the one on the cover, "computational illumination optics manifesto", and "journey of the sorcerer" possibly with some suitable subtitle. Needless to say my supervisors preferred the more traditional option. "Manifesto" seemed a good choice to me as I consider this thesis a first exploration and delineation of what is arguably a new field. The word "sorcerer" is an oblique reference to Clarke's third law. "Journey" was inspired by the fact that I see a PhD as a deeply personal experience. One embarks on an epic quest of discovery, setting out to push the frontier of knowledge. In my opinion, this personal aspect is perhaps slightly underlit, especially in the thesis. To give some glimpses into my experiences, opinions and thought processes, I've therefore added a few footnotes throughout this work. They're meant to show that behind the cold logic and austere mathematics, there's someone who actually had to dream up all this stuff. And I'd like you to get to know him a little better as well through reading this dissertation.

Another thing I'd like to get out of the way is the utilisation of contractions like "we're", "it's", "doesn't", etc. Almost everyone who's read a draft version of my thesis has advised me against using them. However, I insisted and as such I think I owe an explanation. First, I believe contractions read and sound more natural, which is of course the reason they were invented in the first place. But there's another, slightly finer point: they do carry information. Like rhythm and metre indicated by punctuation, there's a subtle difference between using a contraction and not doing so. In particular, not using a contraction puts more emphasis on the word that's otherwise contracted. For instance: "contrary to popular belief it's true" versus "contrary to popular belief it is true". In the second sentence, because "it is" is written out in full, the word "is" is slightly emphasised. If even more emphasis is needed, I'd consider italics, but sometimes you only need a little bit of emphasis. Conversely, using a contraction reduces the risk that a reader inadvertently interprets the emphasis of a sentence different than what was intended. Using the same example, in the first case there's absolutely no way someone might read it and think "it's" should be stressed. This allows the writer to focus the attention on the reader on the fact that will inevitably follow. In the second sentence, on the other hand, the

contradiction itself is emphasised. Therefore, I consider the deliberate usage of contractions simply a writing technique that can be employed to produce more effective writing. As a writer, why would anyone restrict their range of expression?

Then for a more technical note. Part of this work is based on some research articles I've produced over the years, while another part of it has yet to be converted into a few scientific papers. Specifically, the published articles I've authored are:

- "A novel scheme for Liouville's equation with a discontinuous Hamiltonian and applications to geometrical optics," *Journal of Scientific Computing*, vol. 68, no. 2, pp. 739 – 771. Chapters 4 and 8 are an elaboration of the contents found in the article.

- "Embedded WENO: a design strategy to improve existing WENO schemes," *Journal of Computational Physics*, vol. 330, pp. 529 – 549. A large part of the article is exhibited verbatim in Chapter 5. For the complete story, I recommend reading the research article. I've chosen not to put it in entirely since it's fairly long and deals predominantly with the Euler equations of fluid dynamics.

- "High-order embedded WENO schemes," in *Spectral and High Order Methods for Partial Differential Equations ICOSAHOM 2016*, Springer, 2016. This proceedings contribution is an extension of the work of the previous article. It isn't covered here, please read the conference proceedings if you're interested.

- "Full linear multistep methods as root-finders," *Applied Mathematics and Computation*, vol. 320, pp. 190 – 201. This article is the basis for Chapter 3.

There are a further three articles in preparation. Their prospective content can be found inside this thesis. The titles aren't fixed yet, but to give an indication:

- A scientific article titled "The Magic Bullet: a ray tracing method that is almost guaranteed to find the correct intersection point" is in preparation. Chapter 2 is an elaboration of the intended paper.

- Chapters 6 and 9 are based on another article that's in preparation titled "An active flux scheme for Liouville's equation of geometric optics."

- Chapters 10, 13 and 14 will be compiled into an article titled "Optimal design in geometric optics through optimal control of Liouville's equation."

With this list of my scientific articles, be they published or intended, there's only one thing left for me to do now. I'd like to wish you, dear reader, good luck and happy readings.

Cheers,
Bart van Lith.

Contents

Part I

Fundamentals of illumination optics

Chapter 1

Introduction

> "Begin at the beginning," the King said, very gravely, "and go on till you come to the end: then stop."
>
> Lewis Caroll
> *Alice in Wonderland*

Illumination optics deals with designing optics for lighting systems. A whole system, consisting of a light source and an optic, is called a luminaire [1, 2]. Suppose you wish to illuminate a room and you want the lighting to be as uniform as possible. Assuming a given source, the optic of each luminaire has to be designed accordingly. Conversely, if you want a nicely focussed spotlight for use in the theatre, again, the optic has to suit the task.

There are two *a priori* reasons why we'd like to investigate computational methods for illumination optics. The first one is pragmatic: the current state-of-the-art is ray tracing methods, which take a relatively long time for a computation. Reducing the return time allows optical engineers to carry out more iterations of their design, leading to better optics. The second reason is more abstract: a deeper theoretical understanding of computational optics may lead to new ways of thinking and designing.

The ultimate goal in both cases is to provide tools that improve illumination optics design. One example is all that is needed to motivate this desire. We live in an age where we can send people and equipment to space relatively easily. For instance, the International Space Station is continuously occupied [3]. If

you were to look out of the window from the ISS while in orbit over Europe, you'd see something like Figure 1.1.

Figure 1.1: Europe from space, courtesy of NASA Earth Observatory [4].

The figure shows, roughly, what a human would see from space with the naked eye. It's easy to identify major cities and population centres. The logic of improving optics is simple: all the light that you can see from space is wasted energy, radiated out to places we don't care for lighting in the first place. We don't want to light the universe, we only want to light our houses, streets and cities. Improving optics would allow us to direct the light to where we want it to go. The upshot is that we'd need less energy for lighting, something that should resonate with everyone in this time of ecological awareness.

1.1 Foundations of optics

Illumination optics is a branch of optics, which is itself again a branch of electrodynamics [5, 6]. The inclusions are as follows

$$\text{illumination optics} \subset \text{optics} \subset \text{electrodynamics}.$$

We restrict ourselves to the geometrical-optics part of illumination optics. Hence, we'll now derive the basic governing equations of geometric optics from the fun-

damental equations of electrodynamics to provide a complete overview. Electrodynamic phenomena are governed by a set of differential equations known as Maxwell's equations. Although the basic laws were nearly complete at the time, around 1862 Maxwell fixed a crucial inconsistency in the theory by introducing the displacement current [7]. He then used the completed theory to show that, as Maxwell put it: "we can scarcely avoid the inference that light consists in the transverse undulations of the same medium which is the cause of electric and magnetic phenomena" [8].

Behold *Maxwell's equations*:[1]

$$\nabla \cdot \mathcal{D} = \rho_q \qquad \text{(Gauß's Law)}, \tag{1.1a}$$

$$\nabla \cdot \mathcal{B} = 0 \qquad \text{(no name)}, \tag{1.1b}$$

$$\nabla \times \mathcal{E} = -\frac{\partial \mathcal{B}}{\partial t} \qquad \text{(Faraday's Law)}, \tag{1.1c}$$

$$\nabla \times \mathcal{H} = \mathcal{J}_q + \frac{\partial \mathcal{D}}{\partial t} \qquad \text{(Ampère's Law)}, \tag{1.1d}$$

James Clerk Maxwell.

where $\mathcal{E}$ is the electric field, $\mathcal{B}$ is the magnetic field, $\mathcal{D}$ is the displacement field, $\mathcal{H}$ the magnetizing field[2], $\mathcal{J}_q$ is the free current density, all in $\mathbb{R}^3$ [6]. Finally, ρ_q is the free charge density. The term $\frac{\partial \mathcal{D}}{\partial t}$ is due to Maxwell himself and is called the displacement current. We assume linear constitutive relations $\mathcal{D} = \epsilon\mathcal{E}$ and $\mu\mathcal{H} = \mathcal{B}$. This form of Maxwell's equations (1.1) is valid in a bulk material where ϵ is the permittivity and μ the permeability. The vacuum parameters are usually indicated with a subscript zero. These equations describe all classical electric and magnetic phenomena, including all electronics.

From (1.1), it's possible to derive wave solutions that propagate without end: electromagnetic waves. Assuming for simplicity that the material parameters ϵ and μ are constant and there are no free charges or currents, taking the curl of Faraday's and Ampère's laws (1.1c) - (1.1d) leads us to the wave equation, i.e.,

$$\nabla \times \nabla \times \mathcal{E} = -\nabla \times \frac{\partial \mathcal{B}}{\partial t} = -\mu \frac{\partial}{\partial t} \nabla \times \mathcal{H} = -\mu\epsilon \frac{\partial^2 \mathcal{E}}{\partial t^2}, \tag{1.2a}$$

$$\nabla \times \nabla \times \mathcal{B} = \mu\epsilon \nabla \times \frac{\partial \mathcal{E}}{\partial t} = \mu\epsilon \frac{\partial}{\partial t} \nabla \times \mathcal{E} = -\mu\epsilon \frac{\partial^2 \mathcal{B}}{\partial t^2}. \tag{1.2b}$$

[1] Teus Tukker, who was one of my supervisors before he moved to ASML, once told me: "all good optics textbooks start with Maxwell's equations." Naturally therefore, I wanted to start this work with Maxwell's equations as well.

[2] There's some debate about the names of $\mathcal{B}$ and $\mathcal{H}$. Griffiths, for instance calls $\mathcal{B}$ the magnetic field and simply refuses to give a name to $\mathcal{H}$ [6].

The vector identity $\nabla \times \nabla \times \vec{v} = \nabla(\nabla \cdot \vec{v}) - \nabla^2 \vec{v}$, together with the zero-divergence conditions (1.1a) - (1.1b) produces the wave equation for both $\mathcal{E}$ and $\mathcal{B}$, i.e.,

$$\frac{\partial^2 \mathcal{E}}{\partial t^2} = \frac{1}{\mu\epsilon}\nabla^2 \mathcal{E}, \tag{1.3a}$$

$$\frac{\partial^2 \mathcal{B}}{\partial t^2} = \frac{1}{\mu\epsilon}\nabla^2 \mathcal{B}. \tag{1.3b}$$

The coefficient on the right-hand sides form the square of the speed of light in a material, $v = \frac{1}{\sqrt{\mu\epsilon}}$. In vacuum the speed of light is $c = \frac{1}{\sqrt{\epsilon_0 \mu_0}} \approx 3.00 \cdot 10^8$ m/s. The *refractive index* of a material is defined as the ratio of the speed of light in vacuum and the speed of light in the material v, i.e.,

$$n = \frac{c}{v}. \tag{1.4}$$

As it turns out, the speed of light in vacuum is the fastest speed there is, hence $n \geq 1$ for any material.

The wave equations (1.3) support radiation of any frequency, thereby forming an entire spectrum. Visible radiation, or just light for short, is only the tiniest sliver of all these possible frequencies. Lighting is therefore, at its core, a special case of electrodynamics as described by Maxwell's equations. In daily life, however, the wave character of light is never directly apparent. Our familiar surroundings tend to be much larger than any length scale associated with light. Typical wavelengths of light are in the 300 to 800 nm range, which is to a handful of orders of magnitude smaller than everyday objects.

For these reasons, illumination is described in terms of geometric optics, disregarding the microscopic wave character of light and distilling out the macroscopic behaviour. As we'll see, this leads to a description in terms of purely geometric objects such as light rays. Nonetheless, the basic laws of geometric optics follow inexorably from Maxwell's equations.

1.2 The geometric optics limit

Geometric optics can be formally defined as the high-frequency limit of Maxwell's equations, implying infinitely short wavelengths [9]. For illumination this makes a lot of sense, as our to-be-lit objects are humongous compared to the wavelength of light. The previous discussion argued that light is a wave phenomenon,

so we should use propagating waves as a trial solution. The finite speed of light results in identifiable wave fronts, so that the spatial dependence of the phase is determined by a single function [9]. This function, denoted S, measures the optical path length. The wave fronts are given by level sets of S. Hence, our ansatz should read

$$\mathcal{E}(\vec{x},t) = \vec{E}(\vec{x})e^{i(\kappa S(\vec{x})-\omega t)}, \tag{1.5a}$$

$$\mathcal{B}(\vec{x},t) = \vec{B}(\vec{x})e^{i(\kappa S(\vec{x})-\omega t)}, \tag{1.5b}$$

with $\kappa = \frac{\omega}{c}$ the vacuum wave number and ω the angular frequency. The vacuum wave number is also related to the wavelength by $\kappa = \frac{2\pi}{n\lambda}$. The spatial field vectors $\vec{E}$ and $\vec{B}$ are yet to be determined. Furthermore, we again assume that there are no free charges or currents, so that $\rho_q = 0$ and $\mathcal{J}_q = \vec{0}$. We allow ε and μ to vary in space, but we do assume that they're independent of time. Inserting (1.5) into Maxwell's equations (1.1), together with the application of the product rules for curl and divergence, yields

$$\nabla \times \vec{E} + i\kappa \nabla S \times \vec{E} = i\omega \vec{B}, \tag{1.6a}$$

$$\nabla \times \vec{B} + i\kappa \nabla S \times \vec{B} - \tfrac{1}{\mu}\nabla\mu \times \vec{B} = -i\omega\mu\epsilon\vec{E}, \tag{1.6b}$$

$$\nabla \cdot \vec{E} + i\kappa \vec{E} \cdot \nabla S = 0, \tag{1.6c}$$

$$\nabla \cdot \vec{B} + i\kappa \vec{B} \cdot \nabla S = 0. \tag{1.6d}$$

Next, we use (1.4) to rewrite $\mu\epsilon$ in terms of n and c. After dividing by $i\kappa$, we are in a position to let the frequency tend to infinity, so that $\kappa \to \infty$. The resulting set of equations is given by[3]

$$\nabla S \times \vec{E} = c\vec{B}, \tag{1.7a}$$

$$\nabla S \times \vec{B} = -\frac{n^2}{c}\vec{E}, \tag{1.7b}$$

$$\nabla S \cdot \vec{E} = 0, \tag{1.7c}$$

$$\nabla S \cdot \vec{B} = 0. \tag{1.7d}$$

Conditions (1.7c) and (1.7d) tell us essentially that light is a transverse electromagnetic wave with $\vec{E}, \vec{B} \perp \nabla S$. Conditions (1.7a) and (1.7b) furthermore

[3] I find these equations rather appealing. Compare them with Maxwell's equations, which specify the curl and divergence of the electric and magnetic fields. The nabla operators are shifted onto an auxiliary function S, while the time derivative is replaced with a simple multiplication by a scalar.

imply that $\vec{E} \perp \vec{B}$. An equation involving only S and n can be found by dividing (1.7a) by c and substituting the result into (1.7b), after which we obtain

$$\frac{1}{c}\left((\nabla S \cdot \vec{E})\nabla S - \|\nabla S\|^2 \vec{E}\right) = -\frac{n^2}{c}\vec{E}, \tag{1.8}$$

where we've again used the identity $\vec{u} \times (\vec{v} \times \vec{w}) = (\vec{u} \cdot \vec{w})\vec{v} - (\vec{u} \cdot \vec{v})\vec{w}$. Next, applying the fact that the waves are transverse, (1.7c), and realising that the resulting relation must hold for all field directions $\vec{E}$ leads to the *eikonal equation*, i.e.,

$$\|\nabla S\|^2 = n^2. \tag{1.9}$$

To see what this equation tells us, we consider a contour, or isosurface, where S is constant, see Figure 1.2. In this case, the eikonal equation tells us that the gradient of S has length n. Thus, for any isosurface, say $S = S_0$, the gradient of S is simply the local normal multiplied by n. Fix some point $\vec{q}$ on the contour and move a small distance $\frac{\delta}{n}$ out along the normal. A Taylor expansion of S reveals that

$$S(\vec{q} + \delta\,\vec{\nu}) = S + \frac{\delta}{n}\vec{\nu} \cdot \nabla S + \mathcal{O}(\delta^2) = S \pm \delta + \mathcal{O}(\delta^2). \tag{1.10}$$

The sign ambiguity comes from the fact that we only know the norm of ∇S, but not whether it's parallel or antiparallel to the normal of the isosurface. Hence, doing this for all points on the isosurface leads to a new isosurface with value $S_0 \pm \delta$. Let's denote the actual distance between the two contours $\Delta = \frac{\delta}{n}$, so that we may alternatively write the value at the second contour as $S \pm n\Delta$. If the sign is positive, we know we're making a shortest possible step since the gradient is parallel to the normal. If the sign is negative, this indicates an extremum as well, since for any unit vector $\vec{v}$ we have $-n \leq \vec{v} \cdot \nabla S \leq n$. Therefore, the small step from one contour to the other is extremal.

Between two arbitrary isosurfaces with values S_0 and S_1, we can perform a large number of such small steps. Incidentally, some numerical methods that numerically solve the eikonal equation (1.9) are based on this idea [10]. In the limit of vanishing step size, the result is that $S_1 - S_0$ measures an extremal path, weighted by the refractive index n, between the two contours. This is related to the definition of the optical path length.

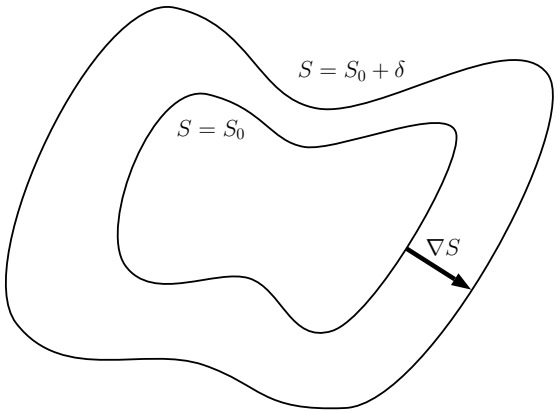


Figure 1.2: Sketch of two isosurfaces of a solution to the eikonal equation for $n = 1$, where C is a constant and δ is a small step.

Let $\mathcal{C}$ be a curve with endpoints $\vec{q}_0$ and $\vec{q}_1$, then its **optical path length** is defined as

$$\int_{\mathcal{C}} n(\vec{q}(s))\,\mathrm{d}s, \tag{1.11}$$

where s is the arc length.

Our previous discussion shows that for solutions S to the eikonal equation (1.9), $S(\vec{q}_1) - S(\vec{q}_0)$ is given by a extremum of the optical path length, where we have to extremise over all curves $\mathcal{C}$ with fixed endpoints $\vec{q}_0$ and $\vec{q}_1$.

1.2.1 Energy transport and rays

Going back to Maxwell's equations (1.1), the energy stored in an electromagnetic field is defined as

$$u = \tfrac{1}{2}\mathcal{E}\cdot\mathcal{D} + \tfrac{1}{2}\mathcal{B}\cdot\mathcal{H} = \tfrac{\epsilon}{2}\|\mathcal{E}\|^2 + \tfrac{1}{2\mu}\|\mathcal{B}\|^2. \tag{1.12}$$

Since the material properties do not depend on time, the time derivative of this quantity in the absence of free currents is determined easily enough, i.e.,

$$\begin{aligned}\frac{\partial u}{\partial t} &= \epsilon\mathcal{E}\cdot\frac{\partial\mathcal{E}}{\partial t} + \frac{1}{\mu}\mathcal{B}\cdot\frac{\partial\mathcal{B}}{\partial t} \\ &= \mathcal{E}\cdot\frac{\partial\mathcal{D}}{\partial t} + \mathcal{H}\cdot\frac{\partial\mathcal{B}}{\partial t}.\end{aligned} \tag{1.13}$$

Using Faraday's and Ampère's laws (1.1c)—(1.1d), we find that

$$\begin{aligned}\frac{\partial u}{\partial t} &= \mathcal{E}\cdot(\nabla\times\mathcal{H}) - \mathcal{H}\cdot(\nabla\times\mathcal{E}) \\ &= -\nabla\cdot(\mathcal{E}\times\mathcal{H}),\end{aligned} \tag{1.14}$$

where in the second step we've used the identity $\nabla\cdot(\vec{u}\times\vec{v}) = \vec{u}\cdot(\nabla\times\vec{v}) - \vec{v}\cdot(\nabla\times\vec{u})$. This is a special case of Poynting's theorem and the vector $\mathcal{P} = \mathcal{E}\times\mathcal{H}$ is called the *Poynting vector*. Just as for $\vec{E}$ and $\vec{B}$, we remove the phase dependence of the Poynting vector by setting $\mathcal{P} = \vec{P}\exp\big(2i(\kappa S - \omega t)\big)$. It can be shown that the Poynting vector is directed[4] along ∇S by applying (1.7a) - (1.7b), i.e.,

$$\begin{aligned}\vec{P} &= \frac{1}{\mu}\vec{E}\times\vec{B} \\ &= -\frac{1}{n^2\mu}(\nabla S\times\vec{B})\times(\nabla S\times\vec{E}) \\ &= -\frac{1}{n^2\mu}\left[\Big(\nabla S\cdot(\vec{B}\times\vec{E})\Big)\nabla S - \Big(\nabla S\cdot(\vec{B}\times\nabla S)\Big)\vec{E}\right],\end{aligned} \tag{1.15}$$

where we've applied the vector algebra identity $\vec{u}\times(\vec{v}\times\vec{w}) = (\vec{u}\cdot\vec{w})\vec{v} - (\vec{u}\cdot\vec{v})\vec{w}$, and the fact that the scalar triple product admits cyclic permutations. The second term in the square brackets is identically zero due to the properties of cross and inner products. In particular, the cross product results in a vector that's orthogonal to both input vectors, while the inner product vanishes for orthogonal vectors. Moreover, we can again identify $\vec{P}$ itself, so that we find

$$\vec{P} = \frac{1}{n^2}\Big(\nabla S\cdot\vec{P}\Big)\nabla S. \tag{1.16}$$

Bearing in mind the eikonal equation (1.9), we see that the right-hand side of (1.16) is the projection of $\vec{P}$ onto ∇S, we see that $\vec{P}$ is parallel to ∇S. Hence, we conclude that energy transport occurs along the normals of the wave fronts.

This motivates the definition of a *light ray*.

[4] I prefer to write that it poynts.

A **light ray** is the integral curve of a wave front normal. As such, rays are the pathways along which energy is transported.

The eikonal equation allows us to find a differential equation for a ray, parametrised by its arc length s, i.e.,

$$\frac{\mathrm{d}\vec{q}}{\mathrm{d}s} = \frac{\nabla S}{\|\nabla S\|} = \frac{1}{n}\nabla S. \tag{1.17}$$

Note that s is the arc length so that $\left\|\frac{\mathrm{d}\vec{q}}{\mathrm{d}s}\right\| = 1$. The ray is the entire curve described by $\vec{q} = \vec{q}(s)$.

As it turns out, the ray can be determined independently from S, which can be shown by multiplying (1.17) by n and differentiating once more with respect to s, i.e.,

$$\frac{\mathrm{d}}{\mathrm{d}s}\left(n\frac{\mathrm{d}\vec{q}}{\mathrm{d}s}\right) = \frac{\mathrm{d}}{\mathrm{d}s}\nabla S = \mathcal{H}_S\frac{\mathrm{d}\vec{q}}{\mathrm{d}s}, \tag{1.18}$$

where $\mathcal{H}_S$ is the Hessian matrix of S, the matrix of second derivatives of S defined as the Jacobi matrix of ∇S. Next, we apply the vector calculus identity $\frac{1}{2}\nabla\|\vec{v}\|^2 = J(\vec{v})\vec{v}$, where $J(\vec{v})$ is the Jacobi matrix of $\vec{v}$. We set $\vec{v} = \nabla S$, so that we obtain $\mathcal{H}_S\nabla S = \frac{1}{2}\nabla(\|\nabla S\|^2) = \frac{1}{2}\nabla n^2$ by the eikonal equation (1.9). This yields the *ray equation*[5], i.e.,

$$\frac{\mathrm{d}}{\mathrm{d}s}\left(n\frac{\mathrm{d}\vec{q}}{\mathrm{d}s}\right) = \frac{1}{n}\mathcal{H}_S\nabla S = \frac{1}{2n}\nabla n^2 = \nabla n. \tag{1.19}$$

The ray equation allows us to find the energy pathways through an optical system in a relatively easy fashion. A single ray can be found without solving the eikonal equation (1.9), a nonlinear PDE, first. Moreover, from the ray equation it's straightforward to see that rays travel in straight lines whenever the refractive index is constant, since the right-hand side of the ray equation then equals zero.

[5]Compare the ray equation to Newton's second law of motion. Interestingly, the refractive index acts as both mass and potential.

1.2.2 Hamiltonian optics

The ray equation, a second-order equation, can be cast into a *Hamiltonian system*, two first-order equations [11]. This is possible by defining the momentum as

$$\vec{p} = n\frac{\mathrm{d}\vec{q}}{\mathrm{d}s}. \tag{1.20}$$

William Rowan Hamilton.

Clearly, using this definition (1.19), we can rewrite the ray equation into a first-order system of ODEs, given by

$$\frac{\mathrm{d}\vec{p}}{\mathrm{d}s} = \nabla n, \tag{1.21a}$$

$$\frac{\mathrm{d}\vec{q}}{\mathrm{d}s} = \frac{\vec{p}}{n}. \tag{1.21b}$$

Rewriting the second-order ODE in terms of two first-order ODEs is a standard trick in ODE theory. However, the system can also be formulated as a Hamiltonian system with Hamiltonian $H(\vec{q},\vec{p}) = \|\vec{p}\| - n(\vec{q})$. Hamilton's equations are defined as

$$\frac{\mathrm{d}\vec{q}}{\mathrm{d}s} = \nabla_{\vec{p}} H = \frac{\vec{p}}{\|\vec{p}\|}, \tag{1.22a}$$

$$\frac{\mathrm{d}\vec{p}}{\mathrm{d}s} = -\nabla_{\vec{q}} H = \nabla n. \tag{1.22b}$$

Note that the Hamiltonian system is equivalent to (1.21) and consequently the ray equation (1.19) only if $H = 0$. We could compute solutions to the Hamiltonian system for any value of the Hamiltonian, but only for the case that $H = 0$ will the pair $(\vec{q},\vec{p})$ represent a physical ray. From here on out, we'll only look at physical rays that have $H = 0$.

> The momentum of a light ray must lie on **Descartes' sphere**, i.e., it must satisfy $\|\vec{p}\| = n$.

Fortunately, one particular property of Hamiltonian systems is that the Hamiltonian is constant if it does not explicitly depend on s, which is easily demonstrated,

$$\frac{\mathrm{d}H}{\mathrm{d}s} = \nabla_{\vec{q}} H \cdot \frac{\mathrm{d}\vec{q}}{\mathrm{d}s} + \nabla_{\vec{p}} H \cdot \frac{\mathrm{d}\vec{p}}{\mathrm{d}s} = 0. \tag{1.23}$$

Hence, if the initial momentum lies on Descartes' sphere, the condition $H = 0$ will be satisfied for all $s > 0$, so that Hamilton's equations (1.22) are indeed equivalent to the ray equation (1.19).

To complete our review of geometric optics, we'll now derive *Fermat's principle*, which follows directly from the fact that rays satisfy Hamilton's equations.

> **Fermat's principle** states that a light ray travelling between two points in space will follow a path such the that optical path length is stationary. Given the collection of all curves $\mathcal{C}$ with fixed endpoints $\vec{q}_0$ and $\vec{q}_1$, a light ray will make (1.11) stationary, i.e., a maximum, minimum or saddle point.

This is the modern version of Fermat's principle. Fermat's original principle stated shortest travel time, which is actually equivalent to shortest optical path length. Fermat, however, didn't know that there are also physical solutions that maximise travel time. A Hamiltonian system is equivalent to an extremal of the action [12], given by

$$S = \int_{\mathcal{C}} \vec{p} \cdot \frac{\mathrm{d}\vec{q}}{\mathrm{d}s} - H \,\mathrm{d}s, \tag{1.24}$$

where we denote the action S since we'll later show that it's equal to the optical path length. The action S attaches a real number to each curve $\mathcal{C}$ that has fixed endpoints $\vec{q}_0$ and $\vec{q}_1$. The integrand is called the Lagrangian $L = \frac{\mathrm{d}\vec{q}}{\mathrm{d}s} \cdot \vec{p} - H$. Contrary to the Hamiltonian, which is interpreted as a function of $\vec{q}$ and $\vec{p}$, the Lagrangian is to be interpreted as a function of $\vec{q}$ and $\vec{q}' = \frac{\mathrm{d}\vec{q}}{\mathrm{d}s}$, where $\vec{q}'$ is considered the independent variable using the relation (1.20). Applying small variations $\delta\vec{q}$ and $\delta\vec{p}$, we see that

$$\begin{aligned} &\int_{\mathcal{C}} (\vec{p} + \delta\vec{p}) \cdot \frac{\mathrm{d}}{\mathrm{d}s} (\vec{q} + \delta\vec{q}) - H(\vec{q} + \delta\vec{q}, \vec{p} + \delta\vec{p}) \,\mathrm{d}s = \\ &S + \int_{\mathcal{C}} \frac{\mathrm{d}\delta\vec{q}}{\mathrm{d}s} \cdot \vec{p} + \frac{\mathrm{d}\vec{q}}{\mathrm{d}s} \cdot \delta\vec{p} - \frac{\partial H}{\partial \vec{q}} \cdot \delta\vec{q} - \frac{\partial H}{\partial \vec{p}} \cdot \delta\vec{p} \,\mathrm{d}s + o\left(\|\delta\vec{q}\|\right) + o\left(\|\delta\vec{p}\|\right), \end{aligned} \tag{1.25}$$

where $\frac{\partial}{\partial \vec{q}} = \nabla_{\vec{q}}$, etc. Integration by parts yields of the term involving the derivative of $\delta\vec{q}$ yields

$$\delta S = \int_{\mathcal{C}} \left(\frac{\mathrm{d}\vec{q}}{\mathrm{d}s} - \frac{\partial H}{\partial \vec{p}} \right) \cdot \delta\vec{p} - \left(\frac{\mathrm{d}\vec{p}}{\mathrm{d}s} + \frac{\partial H}{\partial \vec{q}} \right) \cdot \delta\vec{q} \,\mathrm{d}s + \vec{p} \cdot \delta\vec{q}_1 - \vec{p} \cdot \delta\vec{q}_0, \tag{1.26}$$

where the boundary term vanishes since the endpoints of the ray are fixed, meaning $\delta\vec{q}_0 = \vec{0}$ and $\delta\vec{q}_1 = \vec{0}$. This produces Hamilton's equations (1.22) if we demand that δS vanishes for all small variations in position and momentum by the Fundamental Lemma of the Calculus of Variations [13, 14]. Hence, the trajectory of a ray results in an extremum of the action. All that's left to do is to show that the action is equal to the optical path length to recover Fermat's principle. For this, we need to take a closer look at the Lagrangian.

For physical rays we saw that $H = 0$, while moreover using $\vec{p} = n(\vec{q})\vec{q}\,'$ yields

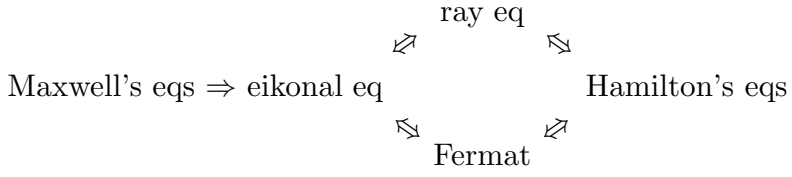
$$S = \int_{\mathcal{C}} n\big(\vec{q}(s)\big)\,\mathrm{d}s, \tag{1.27}$$

which is nothing but the optical path length (1.11). As we've already seen, Fermat's principle provides the formal solution to the eikonal equation (1.9). As advertised light rays, physical solutions to (1.22), therefore extremise the optical path length[6].

Note that S is a path integral, so that we should actually use $L(\vec{q}, \vec{q}\,') = n(\vec{q})\|\vec{q}\,'\|$. This is important to note, since we may then interpret the ray equation as the Euler-Lagrange equation for the extremum of the optical path length (1.11), i.e.,

$$\vec{0} = \frac{\mathrm{d}}{\mathrm{d}s}\frac{\partial L}{\partial \vec{q}\,'} - \frac{\partial L}{\partial \vec{q}} = \frac{\mathrm{d}}{\mathrm{d}s}\left(n\frac{\vec{q}\,'}{\|\vec{q}\,'\|}\right) - \nabla n, \tag{1.28}$$

which yields the ray equation (1.19) if we use again the fact that $\|\vec{q}\,'\| = 1$. Thus, we've found four equivalent formulations of geometric optics: the eikonal equation (1.9), the ray equation (1.19), the Hamiltonian system (1.22) and finally Fermat's principle that the optical path length (1.27) is stationary. The logical structure is as follows:



All are equivalent since we've shown a cycle of implications: the eikonal equations implies the ray equation, while the ray equation implies Hamilton's equa-

[6] Usually, the relation between Fermat's principle and Hamilton's equations is presented the other way around. Fermat's principle is proved from Lagrange's integral invariant, which in turn follows from the fact that the ray direction field has zero curl.

tions, which in turn implies Fermat's principle. Fermat's principle, finally, implies the eikonal equation since it provides its formal solution. All have their use and application, but depending on the problem there can be one that is most easy to work with. We'll focus on the Hamiltonian formulation.

1.2.3 Conservation of étendue

Hamiltonian systems satisfy another interesting property: phase space volume is conserved. Any volume element will have constant volume under transformations generated by Hamilton's equations. As this statement requires a bit more work to prove, we've formulated it in the following theorem. First, however, we present a basic lemma from fluid dynamics, from which the result will follow immediately.

Lemma 1.1. *([15], pp. 15) Given the motion of a fluid particle that is defined by a velocity field $\vec{v}$, i.e.*

$$\frac{\mathrm{d}\vec{x}}{\mathrm{d}t} = \vec{v}(t, \vec{x}), \tag{1.29}$$

the Jacobian determinant $\mathcal{J} = \det\left(\frac{\partial \vec{x}}{\partial \vec{x}_0}\right)$ of the transformation satisfies the following ODE along the motion of the fluid particle,

$$\frac{\mathrm{d}\mathcal{J}}{\mathrm{d}t} = \mathcal{J}\nabla \cdot \vec{v}. \tag{1.30}$$

Thus, the following theorem is evident.

Theorem 1.1. *([12], Chapter 3, Section 16, pp. 69, Theorem 1) For a sufficiently smooth Hamiltonian, volume is conserved under the action of Hamilton's equation* (1.22)*, i.e., let $\mathcal{J}$ be the determinant of the Jacobi matrix of the transformation $(\vec{q}_0, \vec{p}_0) \mapsto (\vec{q}(s), \vec{p}(s))$, then $\mathcal{J} = 1$.*

Proof. It's sufficient to prove that the velocity field defined by Hamilton's equations (1.33) is divergence-free. In particular, we have $\vec{v} = (\nabla_{\vec{p}} H, -\nabla_{\vec{q}} H)$, whereby the divergence of this velocity field is given by

$$\nabla_{\vec{q}} \cdot (\nabla_{\vec{p}} H) - \nabla_{\vec{p}} \cdot (\nabla_{\vec{q}} H) = 0.$$

The order of the differential operators can be interchanged since H is assumed sufficiently smooth. Moreover, the initial condition of the Jacobian matrix is the identity matrix, so that $\mathcal{J}(0) = 1$. □

In optics, the concept of volume in phase space is known as *étendue*[7].

> **Étendue**, i.e., volume in phase space, is conserved under the action of Hamilton's equations. In particular, any volume element has a constant size.

1.2.4 Illumination setting

In illumination optics, it's not very useful to describe the system in terms of the arc length along each ray. It's often more desirable to know the light distribution as a function of the spatial coordinates. For instance, it's often more useful to know the light distribution on a wall rather than on some deformed sphere of fixed optical path length.

To facilitate matters, we recast the Hamiltonian system (1.22) in terms of the third component of $\vec{q}$, which we call z. As z will be used as the evolution coordinate, the remaining positional degrees of freedom are the first two components of the position, which are collected into the now two-dimensional vector $\mathbf{q}$. Similarly, we'll call p_z the third component of $\vec{p}$ and $\mathbf{p}$ the remaining components, leading to

$$\vec{q} = \begin{pmatrix} \mathbf{q} \\ z \end{pmatrix} \quad \text{and} \quad \vec{p} = \begin{pmatrix} \mathbf{p} \\ p_z \end{pmatrix}. \tag{1.31}$$

Since the momentum $\vec{p}$ is confined to Descartes' sphere, we only need to specify the first two components of the momentum and the sign of p_z to recover the three-dimensional vector $\vec{p}$, i.e., $p_z = \pm\sqrt{n^2 - |\mathbf{p}|^2}$. The manifold of all positions and momenta $\mathbf{q}$ and $\mathbf{p}$ is called phase space $\mathcal{P}$. Hence, a point in phase space together with the sign of p_z is sufficient to completely specify a ray. Note that phase space is only the product of position space Q and momentum space P if the refractive index is constant.
It's often useful to take z as the optical axis, as most optics have some axis of symmetry. A plane given by some fixed z is called a screen, which is a plane orthogonal to the optical axis in the case of Cartesian coordinates. The relations between $\mathbf{q}$ and $\mathbf{p}$ and their three-dimensional cousins are sketched in Figure 1.3.

The coordinate z doesn't necessarily need to be some rectilinear coordinate, it may equally well be the radial distance if the problem is easier to treat in

[7] Gilles Vissenberg, Function Owner Illumination Optics at Philips Lighting, was an instructor for the Philips HTO course at the time I did it. He instilled in me the understanding of just how important conservation of étendue is.

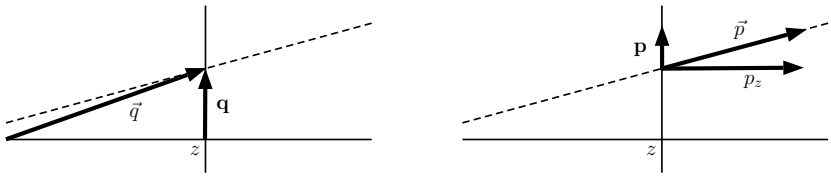


Figure 1.3: Sketch of the screen at z and the relation between the two forms of position and momentum.

spherical coordinates. We can describe geometric optics in terms of z due to the stationary nature of lighting, lamps tend to be fixed. Alternatively, we may use an argument similar to the geometric optics limit: the time scales involved in everyday life are much longer than the time it takes light to travel everyday distances, even if the lighting itself changes rapidly in terms of human perception. The speed of light might as well be infinite as far as we're concerned.

The quantities $\mathbf{q}$ and $\mathbf{p}$ also satisfy a Hamiltonian system, which can be derived from (1.22). Moving along z a little bit changes the position and momentum of each ray on the screen. The relation between s, z and the displacement on the screen can be related to the momentum and its z-component. These relations are sketched in Figure 1.4.

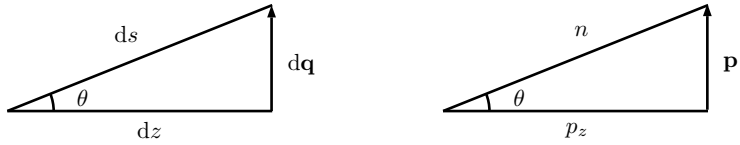


Figure 1.4: Sketch of the relation between ds,dz and the displacement d$\mathbf{q}$ on the screen as related to the momentum and its components.

To find how $\mathbf{q}$ and $\mathbf{p}$ evolve as functions of z, we apply the chain rule and use (1.33), yielding

$$\frac{\mathrm{d}\mathbf{q}}{\mathrm{d}z} = \frac{\mathrm{d}\mathbf{q}}{\mathrm{d}s}\frac{\mathrm{d}s}{\mathrm{d}z} = \frac{\mathbf{p}}{n}\frac{\mathrm{d}s}{\mathrm{d}z}, \tag{1.32a}$$

$$\frac{\mathrm{d}\mathbf{p}}{\mathrm{d}z} = \frac{\mathrm{d}\mathbf{p}}{\mathrm{d}s}\frac{\mathrm{d}s}{\mathrm{d}z} = \frac{\partial n}{\partial \mathbf{q}}\frac{\mathrm{d}s}{\mathrm{d}z}, \tag{1.32b}$$

where $\frac{\partial n}{\partial \mathbf{q}}$ is composed of the first two components of ∇n. Since z is the third component of $\vec{q}$, we see that $n\frac{\mathrm{d}z}{\mathrm{d}s} = p_z$, so that $\frac{\mathrm{d}s}{\mathrm{d}z} = \frac{n}{p_z}$. Clearly, since $\|\vec{p}\| = n$,

we have that $\|\mathbf{p}\| \leq n$, which can be expressed by stating that the momentum on the screen has to lie within Descartes' disc.

The evolution of $\mathbf{q}$ and $\mathbf{p}$ can be formulated as a **Hamiltonian system**, i.e.,

$$\frac{\mathrm{d}\mathbf{q}}{\mathrm{d}z} = \frac{\partial h}{\partial \mathbf{p}} = \sigma \frac{\mathbf{p}}{\sqrt{n(z,\mathbf{q})^2 - \|\mathbf{p}\|^2}}, \tag{1.33a}$$

$$\frac{\mathrm{d}\mathbf{p}}{\mathrm{d}z} = -\frac{\partial h}{\partial \mathbf{q}} = \sigma \frac{n(z,\mathbf{q})}{\sqrt{n(z,\mathbf{q})^2 - \|\mathbf{p}\|^2}} \frac{\partial n}{\partial \mathbf{q}}. \tag{1.33b}$$

The Hamiltonian is now given by

$$h(z,\mathbf{q},\mathbf{p}) = -\sigma\sqrt{n(z,\mathbf{q})^2 - \|\mathbf{p}\|^2} = -p_z, \tag{1.33c}$$

with $\sigma = \operatorname{sgn}\left(\frac{\mathrm{d}s}{\mathrm{d}z}\right)$ representing backward ($\sigma = -1$), marginal ($\sigma = 0$, propagating perpendicular to z) or forward travelling rays ($\sigma = 1$), respectively. Throughout this work, we use $\sigma = 1$ unless mentioned otherwise.

Note that now the Hamiltonian may depend explicitly on z, since n depends on $\vec{q}$. The z-evolution of the Hamiltonian is now given by

$$\frac{\mathrm{d}h}{\mathrm{d}z} = \frac{\partial h}{\partial \mathbf{q}} \cdot \frac{\mathrm{d}\mathbf{q}}{\mathrm{d}z} + \frac{\partial h}{\partial \mathbf{p}} \cdot \frac{\mathrm{d}\mathbf{p}}{\mathrm{d}z} + \frac{\partial h}{\partial z} = \frac{\partial h}{\partial z}. \tag{1.34}$$

Hence, the Hamiltonian will only be preserved in the special case that n doesn't depend on z. Such optics are called guides [16].

1.3 Snell's law and Snell's function

In the absence of smoothly varying refractive index fields, Snell's law and the law of specular reflection are the basic laws of geometric optics. Snell's law was discovered over a millennium ago and has been lost and rediscovered several times over the course of history. Supposedly, Ptolemy of Alexandria had an approximate law roughly 100 AD, while Ibn Sahl had an accurate representation in 984 AD [17, 18]. Independently, Alhazen also formulated an approximate law around 1021 [19]. These works were unknown in western science, where Thomas Harriot discovered the law in 1602, but did not publish his findings [20]. The law was eventually named after Willebrord Snellius, a Dutch scientist

who described it in 1621 [21]. Snellius didn't publish his findings either, but Christiaan Huygens popularised Snellius as the discoverer. It was, in the end, Descartes who first published the sine law in his Dioptrique of 1637 [22].

Willebrord Snellius.

With all our previous machinery in place, Snell's law is straightforward to derive. Suppose a ray hits an interface at some point, where the refractive index jumps from n_1 to n_2, while the interface has local unit normal $\vec{\nu}$. We align the z-axis with the normal, while we choose the origin of the position such that the ray hits the surface at $\vec{q} = \vec{0}$. Using the Hamiltonian system (1.33), we see that $\frac{\partial n}{\partial \mathbf{q}}$ vanishes for this particular choice of coordinate system. Hence, we see that $\mathbf{p}$ is conserved across the interface, which is the tangential momentum. However, due to the orientation of the z-axis, $\mathbf{p}$ represents the momentum parallel to the interface, so that the change in momentum can only occur perpendicular to the interface. From this reasoning, we immediately find that the transmitted momentum must lie in the plane spanned by $\vec{p}_i$ and $\vec{\nu}$. This plane is called the *plane of incidence*. Furthermore, since the change in momentum is in the same direction as the normal, we have

$$\vec{\nu} = \pm \frac{\vec{p}_\mathrm{i} - \vec{p}_\mathrm{t}}{\|\vec{p}_\mathrm{i} - \vec{p}_\mathrm{t}\|}, \tag{1.35}$$

where $\vec{p}_\mathrm{i}$ is the incident momentum and $\vec{p}_\mathrm{t}$ is the transmitted momentum. The sign of $\vec{\nu}$ is such that $\vec{p}_\mathrm{i} \cdot \vec{\nu} \leq 0$. This statement (1.35) provides a full description of Snell's law, but we'll now manipulate it somewhat to lead to a more familiar form. The cross product of any vector with itself is zero, hence taking the cross product of (1.35) with $\vec{\nu}$ yields

$$\vec{p}_\mathrm{i} \times \vec{\nu} = \vec{p}_\mathrm{t} \times \vec{\nu}. \tag{1.36}$$

This is again the same statement that tangential momentum is preserved, expressed differently. Using another property of cross products, we find

$$\|\vec{p}_\mathrm{i}\| \sin\theta_\mathrm{i} = \|\vec{p}_\mathrm{t}\| \sin\theta_\mathrm{t} \tag{1.37}$$

where θ_i is the angle of incidence and θ_t is the angle of transmittance. Due to the sine, there is an ambiguity in the definition of the angles. The convention is that both θ_i and θ_t are always measured positive, see Figure 1.5. Furthermore, the momenta have to lie on Descartes' sphere, so that $\|\vec{p}_\mathrm{i}\| = n_1$ and $\|\vec{p}_\mathrm{t}\| = n_2$, where n_1 and n_2 are the refractive indices.

Consider a light ray propagating in a medium with refractive index n_1 that's incident to an interface where the refractive index changes to n_2. **Snell's law** states that the direction of a light ray is changed discontinuously according to

$$n_1 \sin\theta_\mathrm{i} = n_2 \sin\theta_\mathrm{t}. \tag{1.38}$$

The transmitted ray lies in the plane of incidence, i.e. the plane spanned by the incident ray and the local surface normal.

Snell's law should look familiar to anyone who went to high school. Note that it uses only local information, which is to say only the surface normal at the point of impact of the incident ray is needed. Hence, for a given ray, Snell's law does not depend on the curvature of the surface.

The careful reader will now realise that Snell's law doesn't necessarily always admit a solution θ_t. In particular, when $n_1 > n_2$, there's a range of angles where (1.38) doesn't admit a real-valued solution θ_t. Incident rays with angles in this range suffer *total internal reflection* (TIR). For the situation that $n_1 > n_2$, we can define a critical angle θ_c, where rays incident at the critical angle will cause a transmitted ray with angle $\frac{\pi}{2}$. Hence, by definition, rays with $\theta_\mathrm{i} > \theta_\mathrm{c}$ are reflected and rays with $\theta_\mathrm{i} \leq \theta_\mathrm{c}$ are transmitted. The critical angle is can be found by setting $\theta_\mathrm{t} = \frac{\pi}{2}$ in Snell's law, yielding

$$\theta_\mathrm{c} = \arcsin\left(\frac{n_2}{n_1}\right). \tag{1.39}$$

As angles are represented by momenta in phase space, we can associated with the critical angle a set of critical momenta.

By convention, the incident angle is always positive, while a reflected ray angle is always negative. Therefore, the *law of specular reflection* can be expressed as

$$\theta_\mathrm{r} = -\theta_\mathrm{i}, \tag{1.40}$$

again together with the restriction that the reflected ray lies in the plane of incidence. Note that using $n_2 = -n_1$ in (1.38) results in $\sin\theta_\mathrm{r} = -\sin\theta_\mathrm{i}$, implying (1.40). Hence, a mirror can be thought of as an optical transition to the negative refractive index.

The conclusion of all the above considerations is presented in the following theorem. The result of the theorem is, of course, well known. However, we believe the derivation is new. Alternative derivations can be found in for instance Chaves, see [2], Chapter 12, pp. 403–407.

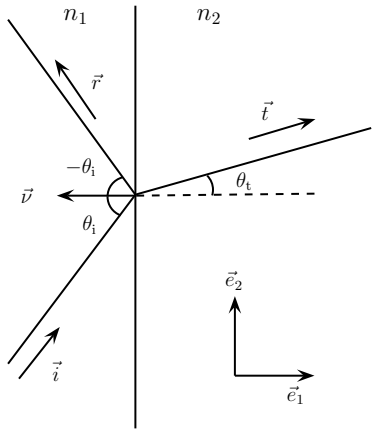


Figure 1.5: Incoming ray with unit vector $\vec{i}$ is refracted to the transmitted ray with unit vector $\vec{t}$. The reflected ray has unit vector $\vec{r}$.

Theorem 1.2. Snell's function
Consider a ray with momentum $\mathbf{p}$ *incident on an interface with surface normal* $\vec{\nu}$ *at the point of incidence. At the interface the refractive index changes discontinuously from* n_1 *to* n_2*. The ray's momentum after encountering the interface is given by* $\mathbf{p}' = \mathcal{S}(\mathbf{p}; n_1, n_2, \boldsymbol{\nu})$*, with* $\mathcal{S}$ *defined as,*

$$\mathcal{S}\left(\mathbf{p}; n_1, n_2, \boldsymbol{\nu}\right) := \begin{cases} \mathbf{p} - \left(\psi + \operatorname{sgn}\left(n_2\right)\sqrt{\delta}\right)\boldsymbol{\nu} & \text{if } \delta \geq 0, \\ \mathbf{p} - 2\psi\boldsymbol{\nu} & \text{if } \delta < 0, \end{cases} \tag{1.41a}$$

where

$$\delta := n_2^2 - n_1^2 + \psi^2 \tag{1.41b}$$

and $\psi := \vec{p} \cdot \vec{\nu}$*. Here,* $\vec{p}$ *and* $\vec{\nu}$ *are the* $\mathbb{R}^3$*-vectors, while* $\mathbf{p}$ *and* $\boldsymbol{\nu}$ *are their first two components, respectively.*

Remark. *Note:* $\boldsymbol{\nu}$ *can be used as an input parameter for* $\mathcal{S}$ *instead of* $\vec{\nu}$*. The first two components of* $\vec{\nu}$ *provide us with enough information, since* $\|\vec{\nu}\| = 1$*. In particular, we have*

$$\psi = \mathbf{p} \cdot \boldsymbol{\nu} \pm \sqrt{n_1^2 - |\mathbf{p}|^2}\sqrt{1 - |\boldsymbol{\nu}|^2}, \tag{1.41c}$$

where we have to choose the sign such that $\psi \leq 0$*, which follows from the angle convention discussed earlier.*

Proof. 1. To simplify our calculations, we'll first construct an orthonormal basis $\{\vec{e}_1, \vec{e}_2\}$ that spans the plane of incidence. Due to the angle convention, an obvious choice for the first basis vector is

$$\vec{e}_1 = -\vec{\nu}.$$

The second basis vector can be found by applying the Gram-Schmidt procedure to the incident unit vector $\vec{i} = \frac{\vec{p}_\mathrm{i}}{n_1}$, yielding

$$\vec{e}_2 = \frac{\vec{i} - \varphi\vec{\nu}}{\sqrt{1-\varphi^2}},$$

where $\varphi := \vec{i} \cdot \vec{\nu} \leq 0$. Since the vectors occurring in the expression of φ are unit vectors, we have $\varphi = -\cos\theta_\mathrm{i}$, so that we can write the sine of θ_i as

$$\sin\theta_\mathrm{i} = \sqrt{1-\varphi^2}.$$

Applying (1.38), we find

$$\sin\theta_\mathrm{t} = \eta_{12}\sqrt{1-\varphi^2}, \quad \cos\theta_\mathrm{t} = \sqrt{1-\eta_{12}^2(1-\varphi^2)},$$

where $\eta_{12} := \frac{n_1}{n_2}$. Due to our choice of basis vectors, the direction vector of the transmitted ray can be expressed as

$$\vec{t} = \cos\theta_\mathrm{t}\vec{e}_1 + \sin\theta_\mathrm{t}\vec{e}_2.$$

Hence, we find $\vec{t}$ as

$$\vec{t} = \eta_{12}\vec{i} - \left(\eta_{12}\varphi + \sqrt{1-\eta_{12}^2(1-\varphi^2)}\right)\vec{\nu}.$$

Recall that the three-dimensional momentum vector lies on Descartes' sphere, meaning $\vec{p}_\mathrm{t} = n_2\vec{t}$ and $\vec{p}_\mathrm{i} = n_1\vec{i}$, whence

$$\vec{p}_\mathrm{t} = \vec{p}_\mathrm{i} - \left(\psi + \mathrm{sgn}\,(n_2)\sqrt{\delta}\right)\vec{\nu}, \tag{$*$}$$

with $\psi := n_1\varphi = \vec{p}_\mathrm{i} \cdot \vec{\nu}$ and $\delta := n_2^2 - n_1^2 + \psi^2$.

2. Whenever $n_2 < n_1$, there are angles at which $\delta < 0$, giving imaginary momenta. These are not physical solutions to Snell's law and must therefore be discarded. In these instances, light suffers total internal reflection and we

must apply the law of specular reflection. As one can see in Figure 1.5, the component of $\vec{i}$ in the direction of $\vec{\nu}$ is reversed, so that we have

$$\vec{r} = \vec{i} - 2(\vec{i} \cdot \vec{\nu})\vec{\nu}.$$

Multiplying with n_1, we obtain the law of specular reflection in momentum form,

$$\vec{p}_{\mathrm{r}} = n_1\vec{r} = \vec{p}_{\mathrm{i}} - 2\psi\vec{\nu},$$

where ψ is as defined earlier.

3. The phase space quantity $\mathbf{p}$ is composed of the first two components of $\vec{p}$. Likewise, $\boldsymbol{\nu}$ is defined as the first two components of $\vec{\nu}$. Compiling all previous information into a single function, we obtain (1.41). □

Remark. *We'll often write* $\mathbf{p}' = \mathcal{S}(\mathbf{p})$ *and take the parameters as understood.*

Remark. *We've left the sign of* n_2 *in the expression of* (1.41a) *so that Snell's function can accommodate mirrors embedded in a medium of refractive index* n_1 *by choosing* $n_2 = -n_1$*. This results in* $\sqrt{\delta} = |\psi|$*, while* $\psi \leq 0$ *so that* $\psi + \operatorname{sgn}(n_2)\sqrt{\delta} = 2\psi$*. In this case, the refraction part of Snell's function is therefore equal to the reflection part.*

We'll refer to $\mathcal{S}$ defined in (1.41) as Snell's function, which also allows us to tackle the reverse problem: given a ray with momentum $\mathbf{p}'$ in a medium with index n_2, find a ray with momentum $\mathbf{p}$ in a medium with index n_1 such that, when refracted, $\mathcal{S}(\mathbf{p}; n_1, n_2, \boldsymbol{\nu}) = \mathbf{p}'$.

Corollary 1.1. Backward ray problem
Given a ray with momentum $\mathbf{p}'$*, the ray with momentum* $\mathbf{p}$ *such that, when refracted, will end up with momentum* $\mathbf{p}'$ *is given by*

$$\mathbf{p} = -\mathcal{S}\left(-\mathbf{p}'; n_2, n_1, -\boldsymbol{\nu}\right). \tag{1.42}$$

Proof. We apply the Helmholtz reciprocity principle [9], to find that we can reverse ray directions with impunity. Recalling the angle convention and applying Snell's function results in (1.42). □

1.4 Overview of this work

With this quick overview of geometric optics, we conclude the first chapter of this thesis. The rest is organised in the following way.

- Part one, which includes this chapter, discusses the fundamentals of illumination optics. This includes a discussion on ray tracing techniques and a more global approach based on phase space distributions and Liouville's equation, a hyperbolic PDE.
- Part two discusses computational methods for hyperbolic PDEs in anticipation of the third part. We visit three numerical methods in their original setting.
- Part three presents methods for the numerical solution of Liouville's equation. The numerical schemes discussed in the previous part are applied to Liouville's equation. All of them require some adaptation to the unique challenges posed by optical interfaces.
- Part four discusses optimal design, the application of optimal control theory to Liouville's equation. We discuss both the optimal design of smooth refractive index fields and freeform interfaces.
- Part five presents conclusions and future research suggestions.

Chapter 2

A magic bullet

> Do it right or don't do it at all.
>
> Ray Charles

In the previous chapter, we showed that rays are the pathways along which electromagnetic energy is transported. Light, as seen by humans, is nothing more than a sensation of that energy. More energy in a single spot means the spot looks brighter. We're therefore interested in how the energy gets directed given a light source and an optic. This will at once show what the arrangement of light will look like. Ray tracing is a straightforward approach that allows us to find out where the energy ends up [23, 24].

It's also possible to do what's known as backward ray tracing, starting at the target and tracing the rays back to the source. This method is in general preferable as it resolves many of the issues that plague forward ray tracing, such as the lack of energy conservation. However, in many cases, it's unknown beforehand where the light is going to end up. Forward ray tracing is then first needed to determine the boundaries of the light distribution on the target, see e.g. Filosa et al. [25]. For this reason, whenever we're talking about ray tracing, we'll mean forward ray tracing.

We can identify three important ingredients of any ray tracing method to compute a light distribution. First, a suitable set of initial conditions, meaning ray starting positions and angles, or a good method of generating them. Second, the actual ray tracer that computes the energy pathways through a given optic. Taking a hint from Ray Charles, this part should work well and give reliable

results. Third and finally, a reconstruction technique to determine the energy distribution from a set of point values.

In this chapter, we'll focus mainly on the second part of such a method, the actual ray tracer. We won't concern ourselves with the generation of initial conditions or the reconstruction of an energy distribution. We'll simply take those as given.

2.1 Elementary ray tracing strategies

There are roughly two types of optics that occur in practice: optics with smoothly varying refractive index fields, and optics with piecewise constant refractive index fields. The first are somewhat rare to encounter, but with the advent of 3d printing technologies, perhaps they'll be a lot more common in the future [26–30]. The second type of optics are dominant by far, mainly due to the relative ease of production. Examples include mirrors, lenses and free-form surfaces.

2.1.1 Smooth refractive index fields

A smoothly varying refractive index field is not so common in practice, although there are such things as gradient index (GRIN) lenses and certain fibre optics that use smooth index fields to contain light. Whenever the refractive index field n is sufficiently smooth, Hamilton's equations (1.33) are well-posed and admit classical solutions, i.e. rays themselves are smooth curves.

In this case, it's prudent[1] to use special structure-preserving integrators [31]. We'll not go into it here, but it's well known that Hamilton's equations generate symplectic transformations. The essential property of such transformations is that they preserve the symplectic two-form[2] [12]. We'll not go into the specifics here, but one can think of it as a more general version of the conservation of étendue. In fact, preservation of the symplectic two-form implies conservation of volume in phase space. A nonexhaustive selection of numerical integrators that respect this property are the leap-frog method, their generalisations in the form of Yoshida integrators [32] and specialised Runge-Kutta integrators [33,34]. Noteworthy among the last one are the methods that bear Gauß's name [35].

[1] I've always failed to see the necessity of symplectic integrators for most optics, see the appendix for a short essay.

[2] This is a concept from differential geometry, a branch of mathematics I find rather beautiful. Some authors, like Vladimir Arnold, claim that no study of classical mechanics is complete

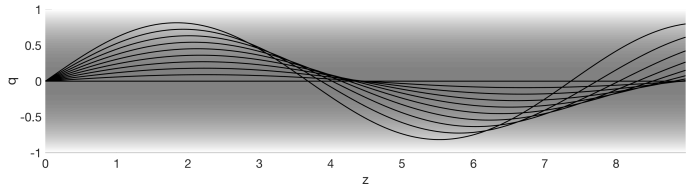


Figure 2.1: Elliptic guide with $n_0 = 1.4$ and κ chosen such that $n(1) = 1$. Several rays, meaning solutions to Hamilton's equations (1.33), starting at $q = 0$ have been plotted for various momenta.

Figure 2.1 shows several rays in an elliptic guide, given by

$$n(q) = \begin{cases} \sqrt{n_0^2 - \kappa^2 q^2} & \text{if } |q| \leq \frac{\sqrt{n_0^2-1}}{\kappa}, \\ 1 & \text{otherwise,} \end{cases} \tag{2.1}$$

which is a smooth guide meant to contain light much like an optical fibre. The refractive index must be great than 1 everywhere, so that $n_0 > 1$. The sign of κ doesn't matter for the shape of the profile, but for convenience we'll assume $\kappa > 0$. We'll look for forward travelling ($\sigma = 1$) solutions of Hamilton's equations (1.33) that remain inside the region $|q| \leq \frac{\sqrt{n_0^2-1}}{\kappa}$. Note that n doesn't depend on z, which is the definition of a guide, so that h is a constant, see (1.34). Therefore, we have $h = -\sqrt{n_0^2 - \kappa^2 q^2 - p^2}$ is a constant from which it follows that

$$\kappa^2 q^2 + p^2 = n_0^2 - h^2. \tag{2.2}$$

Whereas the right-hand side is constant, q and p on the left-hand side change dynamically as we traverse along z. Hence, each ray orbits in a closed elliptical path in phase space, which is why this guide is called elliptic.

The elliptical GRIN profile provides an interesting special case that can be solved exactly. We first manipulate Hamilton's equations (1.33) a little by rewriting them as

$$\begin{aligned} \frac{\mathrm{d}q}{\mathrm{d}z} &= -\frac{p}{h}, \\ \frac{\mathrm{d}p}{\mathrm{d}z} &= \frac{\partial h}{\partial q} = \frac{n}{h}\frac{\mathrm{d}n}{\mathrm{d}q} = \frac{1}{2h}\frac{\mathrm{d}n^2}{\mathrm{d}q}. \end{aligned} \tag{2.3}$$

without studying the differential geometric side of it.

Plugging in the elliptic guide profile (2.1) results in

$$\begin{aligned} \frac{\mathrm{d}q}{\mathrm{d}z} &= -\frac{p}{h}, \\ \frac{\mathrm{d}p}{\mathrm{d}z} &= \frac{\kappa^2}{h} q. \end{aligned} \tag{2.4}$$

Once again, κ and h are both constants, so that differentiating the position equation with respect to z and using the momentum equation results in

$$\frac{\mathrm{d}^2 q}{\mathrm{d}z^2} = -\frac{\kappa^2}{h^2} q. \tag{2.5}$$

Hence, rays are sinusoidal in the elliptic wave guide, though it has to be noted that the spatial frequency $\frac{\kappa}{|h|}$ depends on the initial conditions through the Hamiltonian[3]. Using the ansatz $q(z) = A \sin\left(\frac{\kappa}{|h|} z\right) + B \cos\left(\frac{\kappa}{|h|} z\right)$ quickly results in the general solution

$$\begin{aligned} q(z) &= q_0 \cos\left(\frac{\kappa}{|h|} z\right) + \frac{p_0}{\kappa} \sin\left(\frac{\kappa}{|h|} z\right), \\ p(z) &= p_0 \cos\left(\frac{\kappa}{|h|} z\right) - \kappa q_0 \sin\left(\frac{\kappa}{|h|} z\right), \end{aligned} \tag{2.6}$$

where the initial conditions are given by $(q(0), p(0)) = (q_0, p_0)$. In the paraxial approximation, meaning $|q_0| \ll 1$ and $|p_0| \ll 1$, we have $h = -n_0 + \mathcal{O}(p_0^2) + \mathcal{O}(q_0^2)$. Therefore, rays that stay close to the origin have nearly the same spatial frequency and thus will be approximately in sync. As we progress down the optical axis, the rays will increasingly de-sync with an accumulating phase difference, which is clearly visible in Figure 2.1.

2.1.2 Piecewise constant refractive index fields

Conventional optics, represented by piecewise constant refractive index fields, are by far the most common type of setting for optical and illumination problems. Such fields are identified with shaped optical interfaces like mirrors and lenses. Snell's law and the law of specular reflection in conjunction with specifically designed interfaces are what guides light toward its eventual target. The

[3]Compare (2.5) to the motion of a massive particle in a quadratic potential, $\ddot{x} = -\frac{k}{m} x$, with k the spring constant and m the mass. To me, this is the clearest example of how the refractive index plays the double role of both mass and potential.

basic strategy is to compute the intersection point of a ray with an interface and to change the ray direction there accordingly. This process is repeated until the target is reached. Such methods are also symplectic, albeit implicitly, since Snell's law and the law of specular reflection are in fact canonical transformations [16, 36, 37].

Ray tracing, put in the simplest possible terms, is a four-step process described by the following.

Algorithm 1 Basic ray tracing procedure

Choose a suitable initial condition for the ray.
repeat
  Find the intersection point with the first surface the ray encounters.
  Apply Snell's law or the law of specular reflection accordingly.
until the target surface is reached.

Clearly, these steps should be repeated for each ray independently, which gives some potential for a parallel implementation. Any ray tracing method uses an infinite straight-line representation of the ray, hence the reliability of this process comes down to the reliability of finding intersection points of the line with optically active surfaces. In the simplest situation, therefore, this means selecting the interface that is intersected first along the line. In more complicated cases, the line can intersect a curved surface multiple times and the proper intersection point has to be found. As it turns out, there are no ray tracing methods that come with a guarantee to find the proper intersection point for general optics, at least not that we're aware of. We've developed a method that gives full guarantees in two-dimensional optics and some guarantees in the case of rotationally symmetric three-dimensional optics.

2.2 Magic Bullet

In two-dimensional optics, constructing robust ray tracing methods is relatively easy, which is due to the fact that rays are straight lines in the plane. As such, we'll start with a ray tracing algorithm that is guaranteed to succeed for two-dimensional optics. We'll then later extend this method to three-dimensional optics that exhibit rotational symmetry. Both algorithms are referred to as the Magic Bullet[4].

[4] One of the things I like about composing algorithms is that you get to name them.

For either version of the problem, the following theorem is of the utmost importance.

Theorem 2.1. (Bolzano) *Let $f : [a, b] \to \mathbb{R}$ be a continuous function such that $f(a) \cdot f(b) < 0$. Then there exists at least one root of f in the open interval (a, b). Furthermore, if f is monotone on $[a, b]$, the root is unique.*

Bolzano's Theorem is a special case of the Intermediate Value Theorem, which was proved by Bolzano in his original text [38, 39]. We'll not provide a proof here, though we mention that the proof involves the completeness of the real numbers.

One of the simplest possible methods to subsequently find such a root is the *bisection method*, see e.g. Gautschi [40]. The method starts with the interval $[a, b]$ and iteratively halves the interval while maintaining the sign change. As an algorithm, bisection requires a stopping criterion, which is usually taken to be that the relative size of the interval falls below a specified tolerance[5]. Thus, taking as input the function f, the values a, b and tolerance ϵ, the bisection method can be implemented as Algorithm 2.

Algorithm 2 Bisection

```
while |b − a| ≤ ε|b| do
    m ← a + (b−a)/2.
    if f(m) = 0 return m
    if f(a)f(m) > 0 then
        a ← m
    else
        b ← m
    end if
end while
return b
```

This simple little algorithm is the basis of many robust root-finding methods and the reason is explained by the following lemma.

Lemma 2.1. *Under the assumptions of Bolzano's Theorem, the bisection algorithm is guaranteed to converge. That is, it returns either an approximation to the root with relative error ϵ, or it returns the exact root.*

[5]There's a subtle difference between methods and algorithms: mathematical methods can run forever, while a sound algorithm must terminate after a finite number of steps. Often, an exit condition must be specified to turn a method into an algorithm. The line separating the two is often blurry, but I'll try to be precise.

Proof. Suppose that the exit conditions are not met, then the interval size is halved at every step. Thus, as long as the exit conditions are not met the interval size can only get smaller. Furthermore, it's easy to check that $\text{int}[a, b]$ always brackets a sign change. By Bolzano's Theorem, the root is therefore always contained in the interval. Note that b has at most an absolute error of $|b - a|$. Therefore, if the exit conditions are met, b has at most a relative error of ϵ. Finally, we remark that if m happens to hit upon an exact root, the algorithm exits with the exact root. □

The combination of these two insights allows us to identify the goal of a robust ray tracing algorithm, namely to identify a "good bracket". That is to say an interval, also called a bracket, enclosing the correct root with a sign change. Once we've found one, Bolzano's Theorem and the bisection algorithm guarantee that we can compute the correct root. Hence, if we specify in any of our algorithms to find a root using a bracket, one can imagine that the bisection algorithm is performed. In practice, we'd use a root-finder that is faster and more efficient, but more on this in the next chapter.

2.2.1 Two-dimensional optics

Let the ray be parametrised by its arc length s, so that

$$\mathbf{x}(s) = \mathbf{x}_0 + s\mathbf{v}, \quad s \geq 0, \tag{2.7}$$

where $\mathbf{x}_0 = (x_0, z_0)^T$ and $\mathbf{v} = (L, N)^T$ a unit vector. The surface is described by a piecewise smooth function ζ, called the surface sag, i.e.,

$$z = \zeta(x), \tag{2.8}$$

for $a \leq x \leq b$. In the following, the surface sag ζ is assumed to be at least piecewise twice differentiable. Finding the impact point of a ray with the surface amounts to finding the first intersection point between the smooth function ζ and a straight line. With some abuse of terminology, we'll refer to this straight line as the ray as well. Hence, we're looking for the smallest $s > 0$ such that

$$\Delta z(s) = \zeta(x_0 + sL) - (z_0 + sN) = 0, \tag{2.9}$$

where Δz is called the vertical signed distance function. We assume without loss of generality that $\Delta z(0) > 0$, i.e., the ray starts below the surface and moves towards it so that $N > 0$. For rays starting above the surface we simply flip the

whole problem upside down. Rays starting below the surface with $N < 0$ will never hit it.

The main idea behind the Magic Bullet Algorithm is to first find out whether or not the ray intersects the surface. If the answer is positive, the algorithm should also provide a suitable search bracket containing a unique root. The approach is to divide the surface into pieces that are either convex or concave. This for two reasons: first, a convex or concave function intersects a straight line at most twice; second, the second derivative of Δz has the same sign as ζ''. The first property allows us to identify the correct intersection point, whereas the second property provides smooth convergence for the eventual root-finder.

The division of the surface is done by performing the following algorithm.

Algorithm 3 Bounding triangles

1: Find all points where $\zeta''(x) = 0$ or $\zeta'(x) = 0$ and the endpoints a and b. If the surface is given piecewise, include the breakpoints. Collect these x-values into the ordered set $\mathcal{R}$.
2: Construct the bounding triangles of each segment by the secant line and the two tangent lines.

An example of a result returned from this preprocessing step is shown in Figure 2.2.

The bounding triangles are used to test for an intersection with the smooth surface. We construct two piecewise linear surfaces from the bounding triangles, one which is entirely above the surface and the other entirely below. Clearly, any ray that intersects both surfaces must also intersect the smooth surface in between. Since the surfaces are piecewise linear, the intersection with the ray can be calculated analytically. In particular, if the surface is locally concave, there is no way in which the ray can intersect the bottom surface without intersecting the top surface as well, see Figure 2.3.

In two dimensions, *regular rays* may be classified by the fact that they intersect both the top and bottom surface.

Lemma 2.2. *A regular ray intersects the surface exactly once while* $\Delta z(s_B) > 0$ *and* $\Delta z(s_T) < 0$.

Proof. 1. By construction of the top and bottom surfaces, we have $\Delta z(s_B) > 0$ and $\Delta z(s_T) < 0$. Likewise, we must have both $\Delta z'(s_B) < 0$ and $\Delta z'(s_T) < 0$, as the ray is moving towards the surface at s_B and away from the surface at s_T.

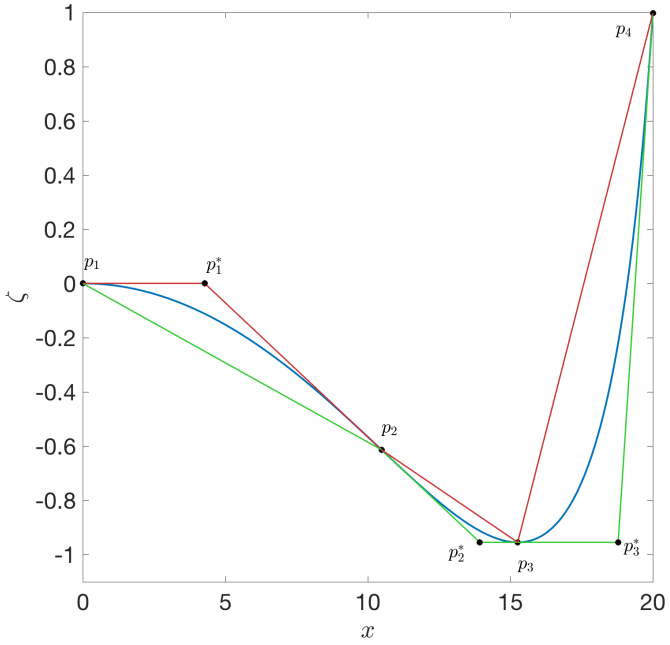


Figure 2.2: Smooth surface (blue) with critical points, p_1 and p_3 are extrema, p_2 is an inflection, and p_4 marks edge of the optic. Points p_1^*, p_2^* and p_3^* are the intersections of the tangent lines on a single piece. The top surface is drawn in red, while the bottom surface is drawn in green.

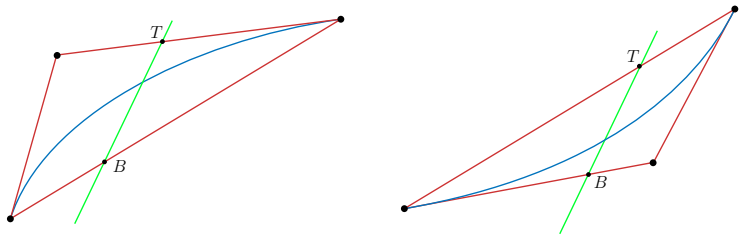


Figure 2.3: Two-dimensional situations for a ray that intersects both bottom and top surfaces. The intersection with the bottom surface is indicated with B, while the one of the top surface is indicated with T.

2. The second derivative of Δz is given by

$$\Delta z''(s) = L^2 \zeta''(x_0 + sL). \tag{2.10}$$

We note that ζ'' has a single sign by construction of the bounding triangles. Thus, $\Delta z'$ is monotone, whereby we find that $\Delta z'(s) < 0$. Thus, we can conclude that Δz is monotone and by Bolzano's Theorem there is a unique root in the bracket $[s_B, s_T]$. □

Thus, regular rays only intersect the surface once, while the $[s_B, s_T]$ provides a suitable search bracket. There is an edge case where the ray is the tangent line at one of the endpoints. This edge case is captured by any decent root-finding algorithm that checks first if $\Delta z(s_B) = 0$ or $\Delta z(s_T) = 0$. Naturally, the smaller of the two should be used, which is by assumption s_B. In the following, we'll assume these edge cases are covered by the root-finder.

On a convex piece, there is a possibility that a ray intersects only the bottom surface without hitting the top surface. Such rays are classified as *skimming rays* and extra care must be taken with them. We mark the points where they intersect the bottom surface B_1 and B_2 and the associated arc lengths $s_{B_1} < s_{B_2}$. They may or may not intersect the surface, and when they do, there are at most two intersection points. The solution is to search first for the minimal vertical signed distance, see Figure 2.4.

The minimal distance is found by solving the nonlinear equation

$$\Delta z'(s) = L\zeta'(x_0 + sL) - N = 0, \tag{2.11}$$

the solution of which is s_D.

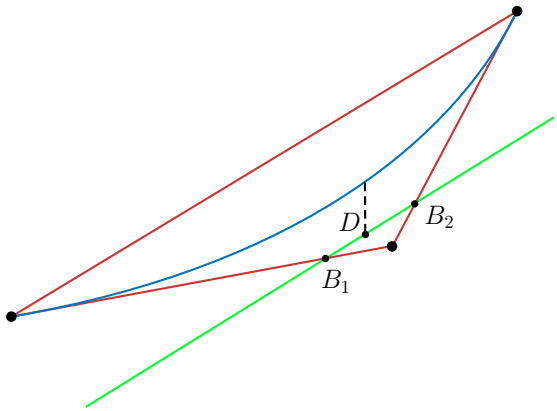


Figure 2.4: Two-dimensional situations for a skimming ray. The first intersection with the bottom surface is indicated with B_1, while the second intersection point is indicated with B_2. The minimal vertical signed distance to the surface is indicated with D.

Lemma 2.3. *Let $\zeta : [a, b] \to \mathbb{R}$ be a smooth convex function, i.e., $\zeta''(x) > 0$. Given a skimming ray, there is a unique root of (2.11), given by $s_D \in \operatorname{int}[s_a, s_b]$. Here, $x(s_a) = a$ and $x(s_b) = b$.*

Proof. 1. The function $\Delta z'$ is monotonous on the interval $\operatorname{int}[s_a, s_b]$, since

$$\Delta z''(s) = L^2 \zeta''(x_0 + sL),$$

so that $\Delta z''(s) > 0$ by the fact that $\zeta''(s) > 0$.
2. Consider the two edges as a piecewise linear function on $[a, b]$ and note that this function is convex. Since the ray is a skimming ray, this means it's a secant line of the piecewise linear function. The derivative $\frac{\mathrm{d}z}{\mathrm{d}x} = \frac{N}{L}$ of the ray is therefore bounded as

$$\zeta'(a) < \frac{N}{L} < \zeta'(b). \qquad (*)$$

Two edge cases can be identified where the secant line of the piecewise linear function overlaps with one of the edges of the bounding triangle. In these cases, the ray has an intersection at one of the end-points.
3. From $(*)$, we observe that $L\zeta'(a) < N < L\zeta'(b)$ if $L > 0$ and $L\zeta'(a) > N > L\zeta'(b)$ if $L < 0$. Therefore, if $L > 0$ we find $\Delta z'(s_a) < 0$ and $\Delta z'(s_b) > 0$. On the other hand, if $L < 0$, we have $\Delta z'(s_a) > 0$ and $\Delta z'(s_b) < 0$. In both

cases $\Delta z'$ exhibits a sign change between s_a and s_b. Furthermore, $\Delta z'$ is a continuous monotone function, so that Bolzano's Theorem tells us there is a unique root. □

Evaluating Δz at s_D reveals whether or not the ray intersects the surface. In particular, if $\Delta z(s_D) > 0$, the ray doesn't intersect the interface. On the other hand, if $\Delta z(s_D) < 0$, we know there must be a unique root in the interval $[s_{B_1}, s_D]$, since $\Delta z'$ has a single sign on that interval. The upshot is that $[s_{B_1}, s_D]$ now provides a suitable bracket to find the correct intersection point of the ray with the surface. The total procedure for one ray may be summarised by Algorithm 4.

Algorithm 4 Ray intersection

```
for each bounding triangle do
    if the ray is a regular ray then
        Set [s_B, s_T] as the search bracket and find the root s of (2.9).
        if s > 0 return s, end if
    else if the ray is a skimming ray then
        Compute s_a and s_b and set [s_a, s_b] as the search bracket
        to find the root s_D of (2.11).
        if s_D < 0 then
            Set [s_B1, s_D] as the search bracket and find the root s of (2.9).
            if s > 0 return s, end if
        end if
    end if
end for
return no intersection
```

The order of inspection of the triangles should be starting at the smallest s and towards increasing s. This way, the first intersection point of the surface and the ray will be found. Note that the special cases where the ray origin is extremely close to the source, for instance when $s_B < 0$, are caught by checking if the root is positive. The Magic Bullet Algorithm consists of using Algorithms 3 and 4 as subroutines. Of course, if many rays are to be traced, the bounding triangle subroutine can be considered a preprocessing step and only has to be performed once per surface, whereas the ray trace intersection subroutine has to be performed at least once for each ray. Due to the previous lemmas, we can conclude the following.

Theorem 2.2. ***(Magic Bullet in 2d)***
Given a piecewise smooth surface and a ray, the Magic Bullet Algorithm is guaranteed to find the correct intersection, or else return that the ray does not intersect the surface.

2.2.2 Rotationally symmetric optics

For rotationally symmetric optics, the problem still reduces to the problem of finding the intersection between two curves in the (r, z)-plane, since the rays are now hyperbolae. This complicates the goal of finding the intersection with the smallest s-value somewhat, though the task is still manageable. As for a guarantee as strong as Theorem 2.2, all bets are off. First, however, we explore some basic facts about ray tracing in this setting. A ray is now given by a line in three dimensions, i.e.,

$$\mathbf{x}(s) = \mathbf{x}_0 + s\mathbf{v}, \quad s \geq 0, \tag{2.12}$$

where $\mathbf{x}_0 = (x_0, y_0, z_0)^T$ and $\mathbf{v} = (L, M, N)^T$.

Lemma 2.4. *Rays are hyperbolae in the (s, r)-plane, i.e., a ray satisfies the equation*

$$\frac{r^2}{r_w^2} - \frac{(s - s_w)^2}{\beta^2} = 1, \tag{2.13a}$$

where

$$s_w = -\frac{x_0 L + y_0 M}{L^2 + M^2}, \tag{2.13b}$$

$$r_w^2 = x_0^2 + y_0^2 - \frac{(x_0 L + y_0 M)^2}{L^2 + M^2}. \tag{2.13c}$$

$$\beta = \frac{r_w}{\sqrt{L^2 + M^2}}. \tag{2.13d}$$

Furthermore, rays are also hyperbolae in the (r, z)-plane, i.e., a ray satisfies the equation

$$\frac{r^2}{r_w^2} - \frac{(z - z_w)^2}{N^2 \beta^2} = 1, \tag{2.14a}$$

where

$$z_w = z_0 + N s_w. \tag{2.14b}$$

Proof. We compute the radial coordinate of a straight line given by (2.12), i.e.,

$$\begin{aligned} r(s) &= \sqrt{(x_0 + sL)^2 + (y_0 + sM)^2} \\ &= \sqrt{x_0^2 + y_0^2 + 2(x_0L + y_0M)s + (L^2 + M^2)s^2}. \end{aligned}$$

Without loss of generality, we can assume that the origin point $(x_0, y_0)^T$ is orthogonal to $(L, M)^T$, as we can shift s by the appropriate amount s_w. Thus, we find

$$r(s) = \sqrt{r_w^2 + (L^2 + M^2)(s - s_w)^2},$$

where r_w ans s_w are defined by (2.13b) - (2.13c). Hence, rearranging terms somewhat gives (2.13a). Furthermore, z is an affine function of s, in fact we have $s = \frac{z - z_0}{N}$, which yields (2.14a) when plugged into (2.13a). □

Remark. *The point where a hyperbola approaches the origin closest is sometimes called the waist, hence the subscript w on all the waist coordinates.*

Remark. *The radial coordinate r is a convex function of s, in particular, we have*

$$r''(s) = \frac{(L^2 + M^2)r_w^2}{(r_w^2 + (L^2 + M^2)(s - s_w)^2)^{\frac{3}{2}}} > 0. \tag{2.15}$$

This can only break down if $r_w = 0$, in which case the ray actually consists of two straight lines in the (r, z)-plane.

We can redefine the bounding triangle subroutine of the previous section on a rotationally symmetrical surface by applying it in the (r, z)-plane. Each edge of a bounding triangle is now rotationally symmetric as well, so that it's in fact a nappe. However, for simplicity we'll still refer to the complex shape that the three nappes make as the bounding triangle. Like in the two-dimensional case, the intersections of a straight line and the edges of the bounding triangle can be found analytically, which we'll now demonstrate. Given a nappe described by the equation $z = a + br$, we look for intersection with a ray given by (2.12), thus

$$a + br(s) = z(s). \tag{2.16}$$

Subtracting a and squaring gives

$$b^2 \left(r_0^2 + 2(x_0L + y_0M)s + (L^2 + M^2)s^2\right) = (z_0 - a)^2 + 2(z_0 - a)Ns + N^2s^2, \tag{2.17}$$

which is a quadratic equation in s, i.e., $Ds^2 + Es + F = 0$, where

$$D = b^2(L^2 + M^2) - N^2, \tag{2.18a}$$
$$E = 2\big(b^2(x_0L + y_0M) + N(a - z_0)\big) \tag{2.18b}$$
$$F = b^2r_0^2 - (z_0 - a)^2. \tag{2.18c}$$

We can characterise the number of intersections by the discriminant, i.e., $E^2 - 4DF$, which as usual allows the distinction of two roots when positive, one when zero and none when negative.

As with the two-dimensional case, the collection of bounding triangles form a top and bottom surface, which are both nappes. If on a single piece the ray doesn't intersect any of the three edges of the bounding triangle, we can confidently state that the ray doesn't intersect the smooth surface. However, unlike in the two-dimensional case, whenever the ray intersects the bottom and top surfaces in a single triangle, we can only state with certainty that there is at least one intersection. Indeed, there may be more than one.

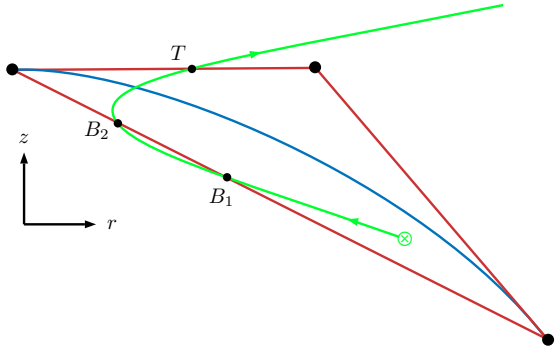


Figure 2.5: Three-dimensional rotationally symmetric optical surface (blue) with a ray (green) that intersects the bottom surface twice and the top surface once. The source of the ray is indicated with a crossed circle.

Another complication will come from the fact that a ray may intersect the bottom surface twice and afterwards intersect the top surface, see Figure 2.5. To avoid any confusion, when we talk about a ray being classified in one way or another, we'll mean that it's classified on a certain interval. The interval is spanned by a pair of consecutive intersections with the nappes.

A zoo of rays

Consider the vertical signed distance function $\Delta z(s) = \zeta(r(s)) - (z_0 + sN)$, where ζ is the surface sag. The surface is given by $z = \zeta(r)$ and ζ is assumed to be piecewise twice differentiable. Like in the two-dimensional case, we assume without loss of generality that the ray starts underneath the surface, so that $\Delta z(0) > 0$ and $N > 0$.

We wish to find conditions under which we are guaranteed to find the correct intersection point, which is given by the smallest $s > 0$ such that $\Delta z(s) = 0$. In the best case, we find an interval where Δz is monotone and exhibits a sign change. Bolzano's theorem and the bisection algorithm provide us with a guarantee that we can close in on the root. Furthermore, if Δz is monotone, the root is unique, this happens when the derivative has a single sign. The derivative of Δz is straightforwardly found, i.e.,

$$\Delta z'(s) = \zeta'\big(r(s)\big)r'(s) - N. \tag{2.19}$$

Unlike in the two-dimensional case, the intersections with the edges of the bounding triangle provide only limited information on the derivative $\frac{\mathrm{d}z}{\mathrm{d}r} = \frac{N}{r'(s)}$ of the ray. Hence, we must look to the second derivative of Δz to glean any information on the monotonicity of Δz, which is given by

$$\Delta z''(s) = \zeta''(r(s))\left[r'(s)\right]^2 + \zeta'(r(s))r''(s). \tag{2.20}$$

Note that both ζ' and ζ'' determine the sign of $\Delta z''(s)$, since $r''(s) > 0$.

In this particular case of rotationally symmetric optics, we can also describe the ray's z-coordinate as a function of r. This leads to a slightly different formulation of the signed vertical distance, which we'll write

$$\delta z(r) = \zeta(r) - z(r). \tag{2.21}$$

Clearly, we have the relation $\Delta z(s) = \delta z\big(r(s)\big)$. The ray is now given by the curves

$$z(r) = z_w \pm N\beta\sqrt{\frac{r^2}{r_w^2} - 1}, \tag{2.22}$$

from Lemma 2.4. Note that the ray is now described by two curves, one for $z < z_w$ and one for $z \geq z_w$. As a function of r, the curve $z \geq z_w$ is increasing concave, while the curve $z < z_w$ is decreasing convex. The derivatives of δz are simply given by $\delta z'(r) = \zeta'(r) - z'(r)$ and $\delta z''(r) = \zeta''(r) - z''(r)$. The sign of $\delta z''$ is now constant whenever ζ'' and z'' have opposite sign. This discussion leads us to the following definition.

Regular rays intersect the bottom and top surfaces in a bounding triangle where one of the following conditions is met:

- $\zeta'' > 0$ and $s_B > s_w$,
- $\zeta'' < 0$ and $s_T < s_w$,
- $\zeta'' \cdot \zeta' > 0$.

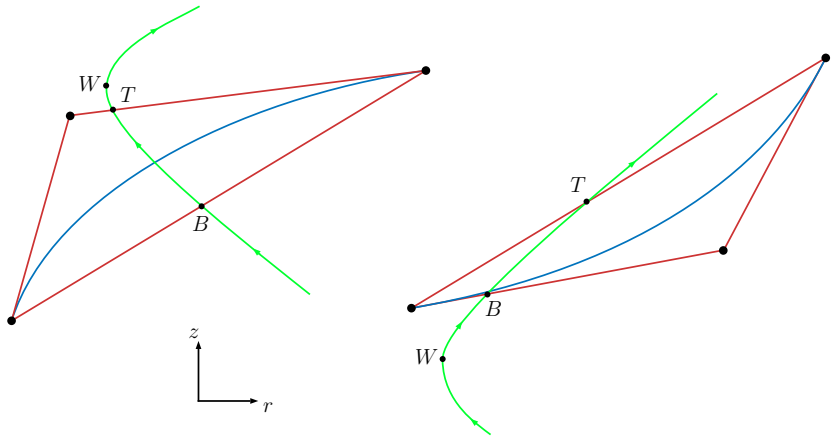


Figure 2.6: Sketch of some regular rays with intersection points B and T indicated, together with the waist W.

Lemma 2.5. *Regular rays intersect the surface exactly once while* $\Delta z(s_B) > 0$ *and* $\Delta z(s_T) < 0$.

Proof. 1. We start with the third case of the regular rays, namely $\zeta''(r) \cdot \zeta'(r) > 0$. Immediately, we see that $\Delta z''(s)$ has a single sign for all $s \in (s_B, s_T)$, according to (2.20) and (2.15). Due to the fact that the ray intersects the top and bottom surfaces, we furthermore have that $\Delta z'(s_B) < 0$ and $\Delta z'(s_T) < 0$. Thus, $\Delta z' < 0$ over the entire interval (s_B, s_T) and hence Δz is monotone. Again, due to the fact that the top and bottom surfaces are both intersected, we have $\Delta z(s_B) > 0$ and $\Delta z(s_T) < 0$. By Bolzano's Theorem, there is a unique root in the interval $[s_B, s_T]$.
2. Next, consider the first case of the regular rays. We note that the condition

$s > s_w$ is equivalent to $z > z_w$ and correspondingly $s < s_w$ is equivalent to $z < z_w$. This is due to the fact that the z-coordinate of the ray can be written as $z(s) = z_w + (s - s_w)N$, while $N > 0$ by assumption. Thus, $s_B > s_w$ implies only the positive branch of (2.22) is used, so that $z''(r) < 0$. We therefore find that $\delta z''(r) > 0$, so that $\delta z'$ is monotone. Furthermore, due to the fact that the ray intersects both top and bottom surfaces, we also have $\delta z'(r) < 0$ at both $r(s_B)$ and $r(s_T)$. Thus, $\delta z'(r) < 0$ over the whole interval. Again, we note that $\delta z(r(s_B)) > 0$ while $\delta z(r(s_T)) < 0$. By Bolzano's Theorem, there is a unique root in the interval $[r(s_B), r(s_T)]$.
3. The second case of the regular rays is completely similar to 2, with the only difference being $\delta z''(r) < 0$. However, we still have $\delta z'(r) < 0$ over the whole interval and a sign change in δz over the interval $[r(s_T), r(s_B)]$. Applying Bolzano's Theorem once more proves the claim. □

Some examples of regular rays are sketched in Figure 2.6, while the ray sketched in Figure 2.5 would be classified as a regular ray on the interval $[s_{B_1}, s_T]$. An algorithm to find the proper intersection of a regular ray is easy to construct, see Algorithm 5.

Algorithm 5 Regular ray subroutine

Set $[s_B, s_T]$ as the search bracket and find the root s of Δz.
if $s > 0$ **return** s, **end if**
return no intersection

The second class of rays that we discuss are rays that only intersect the bottom surface, akin to the skimming rays in the two-dimensional case.

Skimming rays intersect the bottom surface twice in a piece where $\zeta'' \cdot \zeta' > 0$.

Lemma 2.6. *For skimming rays with $\zeta''(r) > 0$, there is at most one root of δz in the interval $[r_w, r(s_{B_1})]$ with $z < z_w$ and there is a unique minimum in δz over the interval $[r_w, r(s_{B_2})]$ for $z > z_w$.*
For skimming rays with $\zeta''(r) < 0$, there are no intersections.

Proof. 1. Consider the case $\zeta''(r) > 0$, so that $\zeta'(r) > 0$.
1a. Suppose $z < z_w$ or equivalently $s < s_w$. Here, we know that z is monotonically decreasing in r while ζ is monotonically increasing. Thus, δz is monotone and any intersection point must be unique. However, as $\delta z\big(r(s_{B_1})\big) > 0$, this

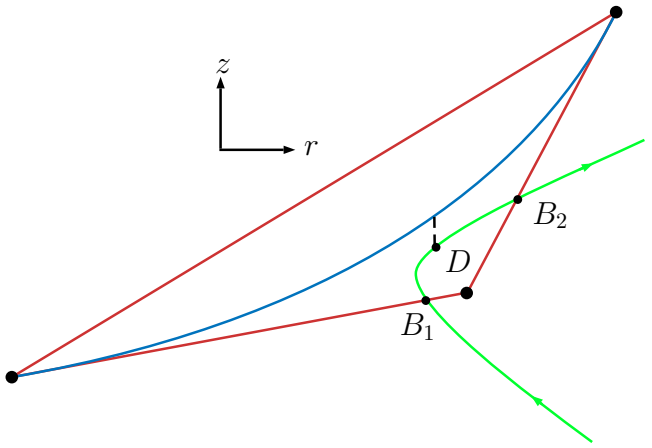


Figure 2.7: Sketch of a skimming ray with intersection points A and B indicated, together with D, the point of minimum Δz.

can only happen if $\delta z(r_w) \leq 0$. Therefore, we either have a suitable interval with sign change, namely $[r_w, r(s_{B_1})]$, or we immediately find the correct root in the case that $\delta z(r_w) = 0$.
1b. Suppose now that $z > z_w$ or equivalently $s > s_w$. Here, z is concave so that $z''(r) < 0$ whereby $\delta z''(r) > 0$. Smooth convex functions have unique minima, provided the derivative $\delta z'(r)$ undergoes a sign change. This must occur since the ray is moving towards the surface at r_w and away at $r(s_{B_2})$ by the fact that it's a skimming ray. Thus, there is a unique minimum of δz over the interval $[r_w, r(s_{B_2})]$.
2. Consider the case $\zeta''(r) < 0$, so that $\zeta'(r) < 0$.
2a. In this case, a skimming ray can only occur for $z < z_w$. Suppose we have a skimming ray for $z > z_w$, which means $z'(r) > 0$, implying that $\delta z'(r) < 0$. However, to be a skimming ray it must also exit through the bottom surface, so that $\int_{r_w}^{r(s_{B_1})} \delta z'(r)\, \mathrm{d}r > 0$. Since $r_w < r(s_{B_1})$, the Mean Value Theorem then implies that there exists some $r^\star$ where $\delta z'(r^\star) > 0$, causing a contradiction. Hence, skimming rays cannot occur for $z < z_w$ in this case.
2b. Since $z < z_w$, we have $z''(r) > 0$, so that $\delta z''(r) < 0$ and thus δz is concave on $\mathrm{int}[r(s_{B_1}), r(s_{B_2})]$. Any concave function must lie entirely above the line segment connecting δz on the endpoints of the interval, hence there is no intersection. □

An example of a skimming ray is sketched in Figure 2.7, while the ray sketched in Figure 2.5 would be classified as a skimming ray on the interval $[s_{B_1}, s_{B_2}]$. The lemma guarantees that we can deal with skimming rays by visiting each of the cases. An algorithm that finds the proper intersection, or determines that there isn't any, is given by Algorithm 6.

Algorithm 6 Skimming ray subroutine

if ζ is convex on the bounding triangle **then**
  Compute r_w.
  if $\delta z(r_w) < 0$ **then**
    Set $[r_w, r(s_{B_1})]$ as the search bracket
    and find the root r of δz with $z < z_w$.
    $s \leftarrow \frac{z(r)-z_0}{N}$.
    if $s > 0$ **return** s, **end if**
  else
    Set $[r_w, r(s_{B_2})]$ as the search bracket
    and find the root r_D of $\delta z'$ with $z > z_w$.
    if $\delta z(r_D) < 0$ **then**
      Set $[r_w, r_D]$ as the search bracket
      and find the root r of δz with $z > z_w$.
      $s \leftarrow \frac{z(r)-z_0}{N}$.
      if $s > 0$ **return** s, **end if**
    end if
  end if
end if
return no intersection

Unlike two-dimensional optics, it's also possible for a ray to intersect the bottom surface twice in concave pieces, i.e., pieces where only the secant edge of the bounding triangle is below the surface. This follows from the fact that there are two solutions of (2.17).

Dipping rays intersect the bottom surface twice in a piece where $\zeta'' \cdot \zeta' < 0$.

Lemma 2.7. *For dipping rays with $\zeta''(r) < 0$, there is at most one root of δz in the interval $[r_w, r(s_{B_1})]$ with $z < z_w$, while there is at least one minimum of δz on $[r_w, r(s_{B_2})]$ with $z > z_w$.*

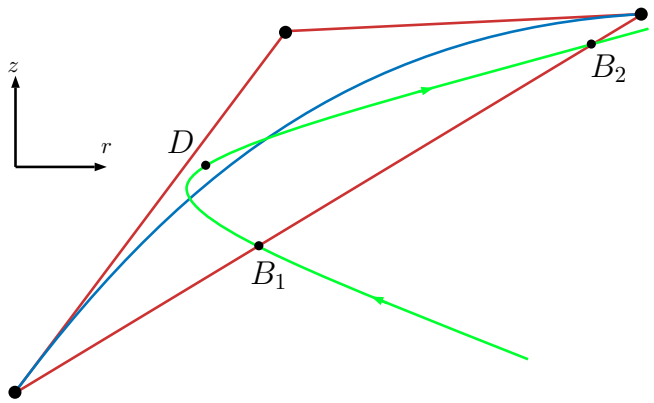


Figure 2.8: Sketch of a dipping ray with intersection points B_1 and B_2 indicated, together with D, the point of minimum vertical distance.

Dipping rays where $\zeta''(r) > 0$ can only occur for $z < z_w$ and there is at least one minimum of δz on $[r_w, r(s_{B_1})]$.

Proof. 1. Consider first the case that $\zeta''(r) < 0$, so that $\zeta'(r) > 0$.
1a. Like with the proof of Lemma 2.6, in the case that $z < z_w$, there can only be at most one intersection point, since δz is monotone. Hence, if δz exhibits a sign change on $[r_w, r(s_{B_1})]$, there is a unique root.
1b. In the case that $z > z_w$, we know there is a sign change in $\delta z'$, with $\delta z'(r_w) < 0$ and $\delta z'(r(s_{B_2})) > 0$, as the ray only intersects the bottom surface, again analogous to the proof of Lemma 2.6. Thus, by Bolzano's Theorem there is at least one minimum in δz on the interval $[r_w, r(s_{B_2})]$. 2. Consider now the case that $\zeta'' > 0$, so that $\zeta' < 0$.
2a. Consider the case that $z > z_w$ and hence $z'(r) > 0$, implying $\delta z'(r) < 0$. However, the ray must eventually leave through the bottom surface, so that $\int_{r_w}^{r(s_{B_2})} \delta z'(r)\,\mathrm{d}r > 0$. Since $r(s_{B_2}) > r_w$, by the Mean Value Theorem there then exists some $r^\star$ where $\delta z'(r^\star) > 0$, which is a contradiction. Hence, dipping rays for $z > z_w$ and $\zeta'' > 0$ cannot occur.
2b. If $z < z_w$, we have that $\delta z'(r(s_{B_2}))) < 0$ while $\delta z'(r(s_{B_1})) > 0$, whence by Bolzano's Theorem there is at least one root of $\delta z'$ on the interval. □

An example of a dipping ray[6] is sketched in Figure 2.8. It must be remarked

[6]The name is new and motivated by the fact that the ray in the (r, z)-plane seems to dip

that it's possible for a dipping ray to have multiple intersections with the surface. We'll show a construction that allows any number of intersections with the surface and a ray. Hence, this limits somewhat our ability to correctly process dipping rays. However, we must note that finding a minimum in δz allows us to find the correct root provided the ray has at most two intersections with the surface on $[r_w, r(s_{B_2})]$ with $z > z_w$ or on $[r_w, r(s_{B_1})]$ with $z < z_w$. An algorithm that finds some intersection point of a dipping ray, or otherwise concludes that there is no intersection, is given by Algorithm 7. The dipping ray subroutine calls a function FIND_MINIMUM, given by Algorithm 8, which determines a minimum with the smallest possible $s > 0$.

Algorithm 7 Dipping ray subroutine

if ζ is concave on the bounding triangle **then**
  Compute r_w.
  if $\delta z(r_w) < 0$ **then**
    Set $[r_w, r(s_{B_1})]$ as the search bracket
    and find the root r of δz with $z < z_w$.
    $s \leftarrow \frac{z(r)-z_0}{N}$.
    if $s > 0$ **return** s, **end if**
  else
    $r_D \leftarrow$ FIND_MINIMUM($[r_w, r(s_{B_2})]$,**true**)
    if r_D is defined **and** $\delta z(r_D) < 0$ **then**
      Set $[r_w, r_D]$ as the search bracket
      and find the root r of δz with $z > z_w$.
      $s \leftarrow \frac{z(r)-z_0}{N}$.
      if $s > 0$ **return** s, **end if**
    end if
  end if
else
  $r_D \leftarrow$ FIND_MINIMUM($[r_w, r(s_{B_1})]$,**false**)
  if r_D is defined **and** $\delta z(r_D) < 0$ **then**
    Set $[r_w, r_D]$ as the search bracket
    and find the root r of δz with $z < z_w$.
  end if
end if
return no intersection

into the bounding triangle through one edge.

Algorithm 8 Function that finds a minimum of δz. The inputs are such that $a, b \in \mathbb{R}$ and topbranch is a logical.

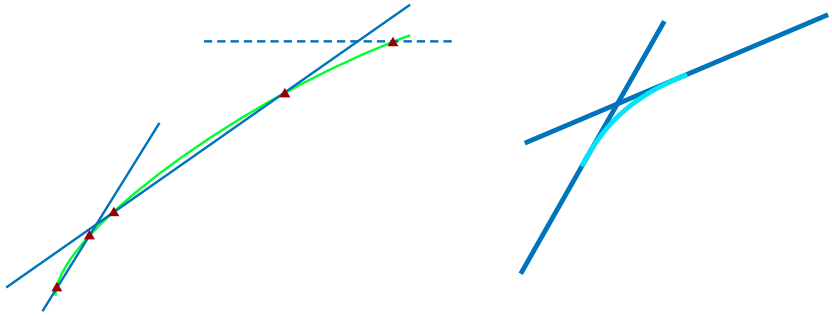
```
procedure FIND_MINIMUM([a, b],TOPBRANCH)
    Check that δz'(a) < 0 and Δz'(b) > 0.
    s ← 0
    repeat
        Find a root r̃ of δz' on the bracket.
        s̃ ← (z(r̃) − z_0)/N
        if δz''(r̃) ≥ 0 and s̃ > 0 then r ← r̃, s ← s̃
        if TOPBRANCH then
            Set the bracket to [a, r̃ − ε].
        else
            Set the bracket to [r̃ + ε, b].
        end if
    until δz''(r̃) ≥ 0
    if s > 0 then
        return r
    else
        return no minimum
    end if
end procedure
```

For dipping rays, we do know there must be an even number of intersections, if we count with the multiplicity. Let's order the roots of δz and label them, taking into account the multiplicity. The first root, the one we wish to find, has $\delta z'(r) < 0$, since the ray is going into the surface there. Sadly, there's no way to distinguish the first root from any odd-numbered root, since all those will have $\delta z'(r) < 0$. The best we can do, therefore, is to ensure that we find an odd-numbered root. Immediately to the left of any even-numbered root, say a step the size of the tolerance, δz will be negative. Thus, any even-numbered root provides us with a smaller bracket that exhibits a sign change in Δz.

The keen reader will notice there's one final class of rays missing, those that intersect the top and bottom surfaces and we have $\zeta'' < 0$ and $\zeta' > 0$, i.e., rays on increasing concave pieces. Unfortunately, it's impossible to say anything useful about these rays other than the fact that there's at least one intersection with the surface, which follows from Bolzano's Theorem. However, monotonicity of Δz is not guaranteed for arbitrary hyperbolae. In other words, for a given ray,

it's always possible to construct a surface that intersects the ray an arbitrary number of times while being concave and increasing. This can be demonstrated with the following construction.

Recall that a chord is a line segment connecting two points on a curve. The secant line, on the other hand, is the whole infinitely extended line connecting those same two points. Consider, then, the fact that a chord of a concave function lies completely below it, while a secant line minus the chord lies completely above it. Therefore, we can set any number of pairs of points and draw the secant lines. Next, we cut the secant lines off where they intersect with their neighbours. This produces a concave piecewise linear surface that has a positive derivative almost everywhere and intersects the ray an even number of times, see Figure 2.9 (a).



(a) (b)

Figure 2.9: Pathological construction for an arbitrary ray (green, curvature is exaggerated for ease of presentation). (a) Two secant lines (blue) result in four intersections (red). The dashed line can be added to cause an odd number of intersections. (b) Replacing the intersection by a smooth connection (light blue).

Odd numbers of intersections can be achieved by adding a horizontal line after the right-most intersection point. Finally, the intersections of the secant lines can be replaced by smooth transitions, see Figure 2.9 (b). The end result of this construction is a smooth uniformly increasing concave surface that features any number of intersections with the ray. This construction works for one specific ray, and it's rather contrived, but the point is that in general, there's no bound on the number of intersections.

Naturally, the example can be flipped in the z-direction, so that the same

construction works for decreasing convex surfaces and rays with $z < z_w$. This motivates the last class of rays[7].

Skipping rays are not regular rays and intersect the bottom and top surfaces in a piece where $\zeta'' \cdot \zeta' < 0$.

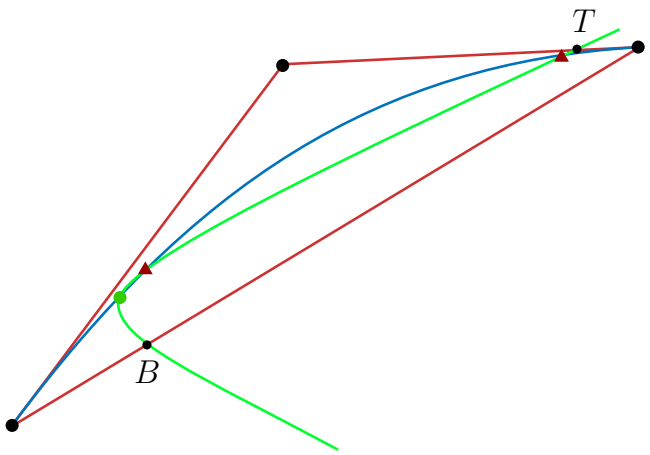


Figure 2.10: Sketch of a skipping ray with intersection points A and B indicated. The correct intersection is indicated with a green circle, while the other two intersections are indicated with a red triangle.

Lemma 2.8. *Skipping rays have at least one root of Δz in the interval $[s_B, s_T]$.*

Proof. The fact that both bottom and top surfaces are intersected implies that $\Delta z(s_B) > 0$ and $\Delta z(s_T) < 0$. Bolzano's Theorem implies that there is at least one root. □

An example of a skipping ray with multiple intersections is sketched in Figure 2.10. Unfortunately, just as with dipping rays, for skipping rays there's once again no guarantees that we'll find the correct root if there are multiple intersections. However, we can again guarantee that we find an intersection where $\Delta z'(s) < 0$. An algorithm that finds some intersection of skipping rays, or otherwise conclude that there are none, is given in Algorithm 9.

[7]The name 'skipping ray' is new too and it's motivated by the fact that a flat rock can skip multiple times if you throw it at oblique angles over a water surface.

Multiple *ad hoc* ways of dealing with skipping rays can be thought of, though none will in fact provide a guarantee to find the correct root. In other words, it's impossible, or at least it would be very hard indeed, to construct a method that allows a theorem like Theorem 2.2. In practice, however, skipping rays that actually have multiple roots are rare to encounter, the "probability" decreasing with the number of roots. Naturally, if a skipping ray only has a single intersection, we're guaranteed to find it using bisection. Here, we'll simply handle skipping rays by finding an odd root as outlined above.

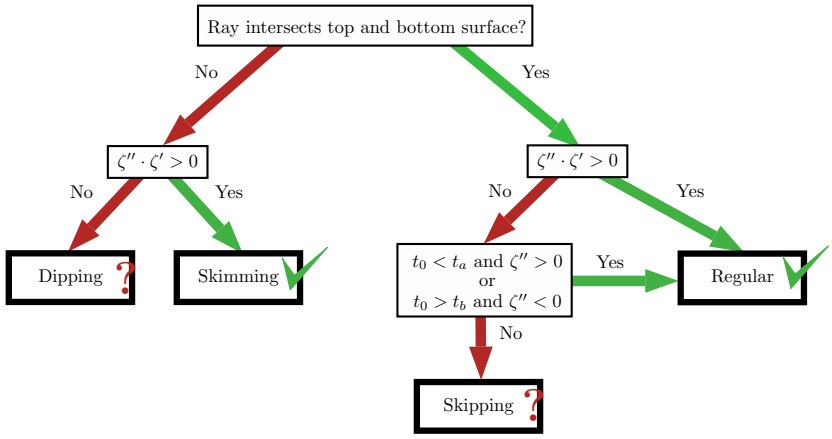


Figure 2.11: Logical structure of the ray zoo in a flow chart. Rays that for which we have a guarantee to find the correct root are indicated with a check mark. For skipping and dipping rays, we don't have a guarantee to find the correct root, hence the question mark.

Algorithm 9 Skipping ray subroutine

Set $[s_B, s_T]$ as the search bracket and find the root $\tilde{s}$ of Δz.
if $\Delta z'(\tilde{s}) \leq 0$ **and** $\tilde{s} > 0$ **then** $s \leftarrow \tilde{s}$
while $\Delta z'(\tilde{s}) > 0$ **do**
    Set $[s_B, \tilde{s} - \epsilon]$ as the search bracket
    and find the root $\tilde{s}$ of Δz.
    if $\Delta z'(\tilde{s}) \leq 0$ **and** $\tilde{s} > 0$ **then** $s \leftarrow \tilde{s}$
end while
return s

The logical structure of the ray zoo can be captured in a flow chart, which is shown in Figure 2.11. To the left, we have the two classes of rays that may or may not intersect the surface and we have to determine the minimum vertical signed distance first to decide. To the right are the two classes of rays that exhibit at least one intersection with the surface. For the regular rays, we know for sure there's only one intersection, while for the skipping rays all bets are off. The only thing we know about them is that there is at least one root and there have to be an odd number, counting multiplicity. With this logical structure in place, Magic Bullet can be composed of the earlier subroutines.

Much like the two-dimensional case, the Magic Bullet Algorithm is the amalgamation of the bounding triangle subroutine and the above subroutines dealing with each class of ray. The bounding triangle subroutine is run once for each surface and can be considered a preprocessing step. For each ray, we run through the triangles in ascending order of the arc length and classify the ray, after which the appropriate ray class subroutine is performed. In the algorithm, there are then roughly two ways of finding an intersection point: if there is a root we find it, if there is a minimum we find that one first to determine if there is a root. In the cases of regular and skimming rays this leads to a foolproof procedure. In the cases of dipping and skipping rays, whether or not we've found the correct intersection cannot be ascertained. Therefore, analogous to the two-dimensional case, we have a the following theorem, which is slightly weaker.

Theorem 2.3. ***(Magic Bullet for rotationally symmetric optics)*** *Given a piecewise smooth surface and a ray, the Magic Bullet Algorithm is guaranteed to find the correct intersection, or else return that the ray does not intersect the surface under the following conditions:*

- *There are no dipping rays with more than 2 intersections.*
- *There are no skipping rays with multiple intersections.*

Remark. *When dealing with a single point source on the z-axis, rays are in fact straight lines, so that the problem reduces to the two-dimensional case. Hence, dipping rays do not occur and skipping rays have at most one intersection. Consequently the Magic Bullet algorithm is guaranteed to succeed.*

In practice, the assumptions of the theorem aren't very restrictive. For instance, if the source is sufficiently far from the optic, the rays in the (r, z)-plane are quite closely approximated by straight lines, as hyperbolae converge rather quickly to their asymptotes. In terms of the theory developed here this means that z_w is usually well below the surface.

2.3 Example

Here, we show an example of a surface where more naive ray tracing algorithms occasionally fail. For instance, simply using Newton iteration without search bracket or using a single bounding box on the entire surface. For the surface, we'll use an aspheric surface with a very strong polynomial part, given by

$$\zeta(r) = \frac{cr^2}{1 + \sqrt{1 - (1+\kappa)c^2r^2}} + \sum_{i=1}^{p} a_i r^i, \tag{2.23}$$

where c is the curvature, κ is the conic constant and the sum represents the polynomial correction with a_i the polynomial coefficients. In this example, we choose $c = \frac{1}{21}$, $\kappa = 0$ and a polynomial order of $p = 4$ with coefficients $a_1 = 0$, $a_2 = -3 \cdot 10^{-2}$, $a_3 = 0$ and $a_4 = -10^{-5}$. The maximum radius of the surface is $r = 20$. As a root-finder for the Magic Bullet, we've used the bracketed method that will be introduced in the next chapter in Section 3.6.

For the special case where all the rays originate at the origin, rays are straight lines and the problem reduces to the two-dimensional version of Section 2.2.1. Incidentally, Newton iteration is now also guaranteed to converge as the vertical signed distance becomes a convex or concave function on each search bracket $[r_k, r_{k+1}]$. In this special case, Newton iteration converges monotonically, so that if we start at the smaller end of the search bracket, convergence to the proper intersection point is guaranteed. We therefore use the collection of rays given by $\mathbf{x}_0 = (0, 0, -3)^T$ and

$$\mathbf{v} = \frac{1}{\sqrt{(\frac{j}{4})^2 + 1}} \begin{pmatrix} \frac{j}{4} \\ 0 \\ 1 \end{pmatrix}, \tag{2.24}$$

with $j = 0, 1, \ldots, 40$, see Figure 2.12. Simply using Newton's method evidently converges to the same roots as bisection. For both algorithms, we've used the tolerance criterion of a relative difference in iterations of twice the machine precision, i.e., between two iterations s_k, we stop when $|s_k - s_{k-1}| \leq 2\epsilon_{\text{mach}}|s_k|$. To avoid infinite loops or uncontrollable runaway divergence, the maximum number of iterations for Newton's method was set to 100.

Next, we trace a collection of rays that do not originate from $r_0 = 0$, so that the rays in the (r, z)-plane become hyperbolae. For this example, Newton's method has no convergence guarantee. We set $\mathbf{x}_0 = (0, y_0, -1)^T$ and we use 40 equidistant points between 0 and 20 for y_0, see Figure 2.13.

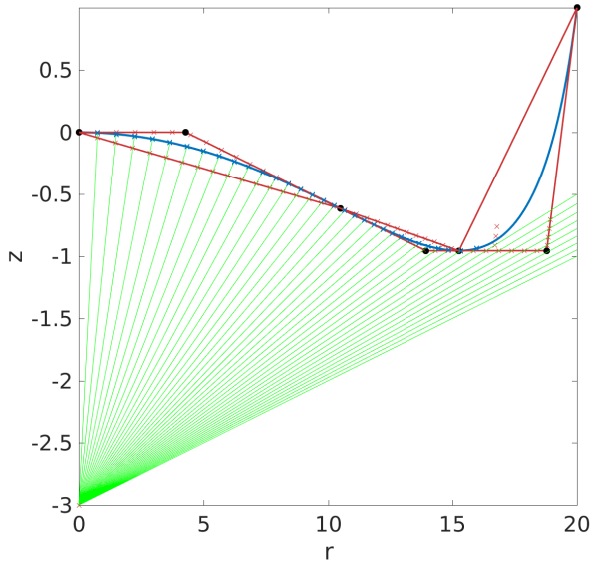


Figure 2.12: The Magic Bullet Algorithm applied to a collection of rays all starting with $(r_0, z_0) = (0, -3)$. Newton's method finds the same roots. The red crosses are the search bracket endpoints.

As advertised, the Magic Bullet Algorithm finds the correct intersection points for all rays. Newton's method is applied to the rays that are determined by the Magic Bullet Algorithm to hit the surface. However, Newton's method does not converge for all rays, even for what we've called regular rays. It should be noted that Newton iteration alone finds the majority of the correct intersection points. Therefore, Newton's method represents here the more naive ray tracing algorithms. The Magic Bullet Algorithm provides not only a guarantee that an intersection is present, but also an enclosing bracket with a sign change. As such, any bracketing method is guaranteed to find the root. The difference may be illustrated somewhat more dramatically by increasing the density of rays from 40 to 5000, see Figure 2.14.

The figure shows that Newton's method misses quite a few rays, 465 out of

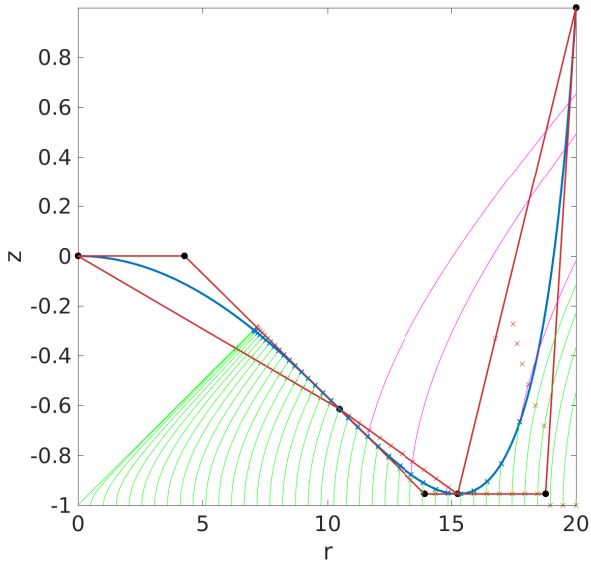


Figure 2.13: The Magic Bullet Algorithm applied to a collection of 40 rays and Newton's method applied to the same rays. Rays for which the proper root was found are plotted in green, while for purple rays Newton's method did not converge. The red crosses are the search bracket endpoints.

5000, about 9.3%. We furthermore see that Newton's method finds most of the rays in the region where the surface is concave. Where the surface is convex, quite a lot of intersection points are missed. Another interesting fact is that Newton's method simply diverges for these cases, instead of for instance finding the wrong intersection point. The Magic Bullet, on the other hand, finds all the correct intersection points.

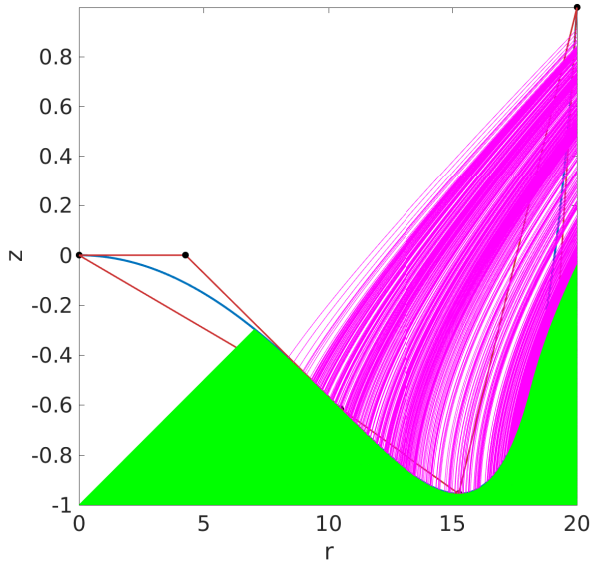


Figure 2.14: The Magic Bullet Algorithm applied to a collection of 5000 rays. Rays for which the proper root was found are plotted in green, while for purple coloured rays Newton's method did not converge.

Chapter 3

Finding roots fast

> Money, so they say, is the root of all evil today.
>
> Pink Floyd
> *Money*

In the previous chapter, we've focussed on producing a good search bracket to find a root of a sufficiently smooth nonlinear function, i.e., $f : \mathbb{R} \to \mathbb{R}$ and we're asked to solve the equation

$$f(x) = 0. \tag{3.1}$$

Provided with a good search bracket that encloses a root, bisection is guaranteed to converge. In the previous chapter, we've used the bisection algorithm mostly as a theoretical tool to prove theorems, rather than a practical root-finder. We'll now focus on the other side of the story, which is to find the root as efficiently as possible. The bisection method converges rather slowly, as every bisection step gains one bit of accuracy on the root. For double precision numbers therefore, the bisection algorithm takes at most 52 steps, since then the machine precision is reached. Typical runs of the bisection algorithm are close to the maximum number of steps. Obviously, we'd like to obtain a method that converges in fewer steps and ideally does less work.

Let's focus on the abstract problem posed by (3.1). This archetypical problem is ubiquitous in all fields of mathematics, science and engineering. Clearly, we have ray tracing in the backs of our minds, but the applications are legion.

For instance, implicit ODE solvers are often formulated like (3.1), after which a root-finder of some kind is applied [35]. Depending on the properties of the function f, there are several methods that present themselves. We'll discuss some well-known examples in the next section.

Recently, a new interpretation of root-finding methods in terms of ODEs has been proposed by Grau-Sánchez et al. [41–43]. Their idea is to consider the inverse function derivative rule as an ODE, so that any explicit ODE solver may be converted to a root-finding method. Indeed, Grau-Sánchez et al. have successfully introduced root-finders based on Adams-type linear multistep and Runge-Kutta integrators. It goes without saying that only explicit ODE solvers can be usefully converted to root-finding methods. However, predictor-corrector pairs are possible, as those methods are indeed explicit.

The main theoretical result of this chapter is a theorem on the convergence rate of root-finders based on explicit linear multistep methods (LMMs). We furthermore prove a barrier on the convergence rate of LMM-based root-finders. It turns out that adding a few history points quickly boosts the convergence rate close to the theoretical bound. However, adding many history points ultimately proves an exercise in futility due to diminishing returns in the convergence rate.

Looking back to applications such as ray tracing, we also discuss a robust LMM-based method combined with bisection to produce an algorithm that can been seen as an extension of Brent's method [44] with information from the derivative added. Similar to Brent's method, our algorithm is guaranteed to converge to a root whenever an enclosing starting bracket is provided[1], i.e., an interval $[a, b]$ with $f(a)f(b) < 0$.

3.1 Classes of root-finders

At the highest conceptual level, there are two major classes of root-finders: those with memory and those without. Over the last few decades, research effort has been mainly focusses on memoryless root-finders. As the name suggests, these methods are defined locally and make no reference to previous iterations. They can, however, work in multiple steps or use information from derivatives. Classical examples of memoryless root-finders include Newton's method [45], which we'll discuss in the next section, and Halley's method [46]. The memoryless

[1] While working on the Magic Bullet, I originally used Brent's method to close in on the roots after obtaining a good search bracket. Not knowing any better, I felt it was a bit of a waste not to use the derivative as well. This observation was the basis of the full LMM root-finders.

methods are epitomised by the Kung-Traub conjecture, which states that a memoryless method using w evaluations has an optimal order of 2^{w-1} [47]. An evaluation here simply means an evaluation of the function f or its derivatives. Methods that achieve this convergence rate are referred to as optimal methods and there is no shortage of them, see for instance [48–56]. We'll not be exploring this class of root-finders though, as we'll be focussing on the other class.

Root-finders with memory do store their previous iterations, or at least some part of it. The function values and possibly derivative values of the previous iterations are used to enhance the accuracy of the next iteration. As such, they can provide high accuracy with only minimal computational effort in terms of function evaluations. However, they often do require more storage compared to the memoryless methods. The proposed full LMM root-finders that we'll discuss over the course of this chapter fall in this class. Indeed, the distinguishing characteristic of LMM solvers for ODEs is that they use previous iterations of the numerical solution to increase accuracy. In the same way, we'll show that adding information from the iteration history also results in faster root-finders. First, however, we'll discuss some basic examples and concepts.

3.2 Some famous root-finders

For completeness, we'll discuss some of the more well-known specimens of root-finders. A more complete theory can be found in any textbook on numerical analysis, such as Gautschi [40] or Quarteroni[2] [57]. Central to root-finders is the rate of convergence, which is defined as the real number p such that

$$\lim_{k\to\infty} \frac{|x_k - x^\star|}{|x_{k-1} - x^\star|^p} = C > 0, \tag{3.2}$$

where x_k is the sequence generated by the root-finder and $x^\star$ is the root. The efficiency measure of a root-finder is defined as $p^{\frac{1}{w}}$, where w is the number of evaluations per iteration. Bisection, for example, converges linearly ($p = 1$) and costs one function evaluation per iteration, assuming we store the function value of the previous iteration, thus exhibiting an efficiency measure of 1.

[2]When on my second conference, to ICOSAHOM2016 in Rio de Janeiro, I was sitting next to Alfio on the bus to the cultural activity. He introduced himself to me and looked at me expecting some kind of response but all I had to say was "nice to meet you." Now, I've used several of the textbooks he wrote or co-authored and I imagine the conversation would have been quite different.

The most ubiquitous root-finder in computational practice is probably Newton's method, also known as the Newton-Raphson method [58, 59]. Geometrically, the method consists of successively finding the roots of tangent lines. The tangent line at the current iteration x_k is given by $L(x) = f(x_k)+f'(x_k)(x-x_k)$. Hence, we use the root of the tangent line at the current iteration as the next iteration, so that

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}. \tag{3.3}$$

Newton's method is an example of a memoryless method and as such, it requires a single point x_0 as an initial guess to the root. Provided the initial guess is sufficiently close to the root, the sequence generated by (3.3) converges quadratically ($p = 2$) to the root, which we'll prove later in a more general setting. Under the assumption that f and f' cost roughly the same to evaluate, Newton's method has an efficiency measure $\sqrt{2} \approx 1.41$.

Next, the secant method can be interpreted as Newton's method where the derivative is approximated by the secant line spanned by the current and previous iterations, thus

$$x_{k+1} = x_k - f(x_k)\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}. \tag{3.4}$$

Consequently, the secant method is a method with memory, one memory slot so to speak, hence it requires two points by way of an initial guess. The convergence rate is given by the golden ratio, $p = \frac{1+\sqrt{5}}{2} \approx 1.62$, which is slower than Newton's method, but faster than bisection. If we store the function value of the previous point, the secant method can do with a single function evaluation per iteration, producing an efficiency measure of 1.62.

Methods like Newton's and the secant method are called open root-finders because they don't start with a search bracket. They may exhibit runaway divergence for poorly behaved functions. Methods like bisection, on the other hand, require an enclosing bracket as an initial guess, those methods are called closed or bracketed methods. Dekker's method[3] was constructed to fix the drawback of the open secant method by combining it with bisection [60]. The result is a fast and robust root-finder. It requires two points, called a and b, as its initial guess. They span an interval $\text{int}[a, b]$ such that $f(a) \cdot f(b) < 0$, so they're said to bracket a sign change. The point b is the best estimate of the root so far, meaning $|f(b)|$ is the smallest function value up to the current iteration. The point a is the contrapoint, which simply means $f(a) \cdot f(b) < 0$.

[3] I had the privilege of meeting Theodorus Dekker at the 2015 WSC Woudschoten meeting.

Finally, the previous value of b is also stored and it's denoted c. Initially, c is set to a.

The basic idea is to perform a secant step unless it's outside the bracket $\text{int}[a, b]$. The method computes the midpoint of the interval $\text{int}[a, b]$ and a provisional point by the secant method applied to b and c. Given a tolerance ϵ, the algorithm is described by choosing one of the following three options:

- A minimal step if the secant step is within the tolerance of the interval endpoints.
- A secant step if it lies in the within the interval $\text{int}[a, b]$, but at least ϵ away from the endpoints.
- A bisection step otherwise.

A minimal step is a step of size ϵ. During each iteration, the definitions of a, b and c are maintained as invariants, so the appropriate swaps are made on occasion. The algorithm stops under the same condition as the bisection method, namely that $|a - b| \leq \epsilon|b|$. In essence, Dekker's method tries to make a secant step, but if it fails the fall-back is bisection.

Dekker's method is the progeny of the bisection and secant methods, inheriting all their favourable properties and few of the bad ones. As such, it's guaranteed to converge, see Lemma 2.1, while under reasonable circumstances it provides a convergence rate of approximately 1.62. If the function values of the points a, b and c are stored, the method furthermore uses only a single function evaluation per iteration, resulting in an efficiency measure of 1.62 as well in the best case.

Unfortunately, it's possible that Dekker's method takes a long time to converge, being far slower than the bisection method in the worst case. An example is $f(x) = x^3$ with a starting bracket $[-1, 1]$, where the secant step will exactly hit the origin, but the stopping criterion fails to recognise the root. The method then takes minimal steps until the interval is sufficiently small. In the worst case, therefore, Dekker's method converges quite slowly indeed, as it's essentially trying every number that can be represented with floating point arithmetic in the search interval.

Historically, Brent's method is again an improvement over Dekker's, where several other checks are performed [44]. It uses the same definitions of a, b and c, so that b is the best estimate of the root, c is the previous value of b and a is the contrapoint. Brent spotted that a quadratic can be constructed from the three available points, although the problem is that it may have zero, one or

two real roots. To fix this, Brent's method uses quadratic interpolation on the inverse, so that the root is always well-defined as long as $a \neq c$ and $f(a)$, $f(b)$ and $f(c)$ are different. The algorithm is described by a choice of the following four possibilities:

- If $a \neq c$ and $f(a)$, $f(b)$ and $f(c)$ are unique, use inverse quadratic interpolation to compute a provisional point. Else, use the secant method for this.
- Accept the provisional point if it lies within the interval $\text{int}[a, b]$, but at least ϵ away from the endpoints.
- Use a bisection step if the provisional point lies within tolerance of an endpoint or outside the interval $[a, b]$.
- If the step size of the provisional point is smaller than the tolerance, perform a minimal step.

The stopping criterion is again the same as Dekker's and bisection, but Brent's method additionally checks if it has accidentally hit an exact root. Brent's method comes with a stronger guarantee than Dekker's, as it converges to the root in at most twice the number of steps as bisection. For reasonably well-behaved functions, it will converge with a rate of approximately 1.84. By the same token as Dekker's method, it only needs one function evaluation per iteration, so that the efficiency measure is also 1.84 in the best case. Of course, it converges linearly in the worst case and thus the worst-case efficiency is 1.

3.3 Barriers on LMM root-finders

As mentioned earlier, Grau-Sánchez et al. introduced a new paradigm for the root-finding problem in [41]. It's based on reformulating the problem in such a way that it's equivalent to solving an ODE. Using this elegant reinterpretation, any ODE solver can be converted to a root-finder. The ODE formulation can be derived by assuming that the function f is sufficiently smooth and invertible in the vicinity of the root. Under these assumptions, the chain rule applied to $x = f^{-1}\big(f(x)\big)$ results in

$$\frac{\mathrm{d}x}{\mathrm{d}y} = [f^{-1}]'(y) = \frac{1}{f'\big(x\big)} = F(x), \tag{3.5}$$

which we may interpret as an autonomous ODE for the inverse with y as the independent variable[4]. Integrating (3.5) from an initial guess x_0 with function value $y_0 = f(x_0)$ to $y = 0$ yields

$$x(0) = x_0 + \int_{y_0}^{0} F\big(x(y)\big)\,\mathrm{d}y. \tag{3.6}$$

Immediately, we see that applying the forward Euler method to (3.5) with step size $h_{n+1} = y_{n+1} - y_n = -f(x_n)$ gives Newton's method. From (3.6), we see that the step size of the integrator should be taken as $0 - y_0 = -f(x_0)$. However, Newton's method may also be interpreted as an inverse linear Taylor method, i.e., a method where the inverse function is approximated by a first-order Taylor polynomial. Indeed, any linear numerical integration method applied to (3.5) can be interpreted as inverse polynomial interpolation.

As such, explicit linear multistep methods applied to (3.5) will also produce a polynomial approximation to the inverse function. Such methods have the form

$$x_{n+s} + \sum_{k=0}^{s-1} a_k^{(n)} x_{n+k} = h_{n+s} \sum_{k=0}^{s-1} b_k^{(n)} F(x_{n+k}). \tag{3.7}$$

The right-hand side cannot depend on x_{n+s}, otherwise we end up with an implicit root-finder, which would not be terribly useful. The coefficients of the method, $\{a_k^{(n)}\}_{k=0}^{s-1}$ and $\{b_k^{(n)}\}_{k=0}^{s-1}$, will depend on the previous step sizes and will therefore be different each step. The step sizes are given by $h_{n+k} = y_{n+k} - y_{n+k-1}$, that is the differences in y-coordinates. Since we wish to find the root, we set $y_{n+s} = 0$, leading to $h_{n+s} = y_{n+s} - y_{n+s-1} = -y_{n+s-1}$. Furthermore, the y-coordinates are of course given by the function values of the root estimates, $y_{n+k} = f(x_{n+k})$, so that

$$h_{n+k} = f(x_{n+k}) - f(x_{n+k-1}) \quad \text{for } k = 1, \ldots, s-1. \tag{3.8}$$

Like an ODE solver, we may use an implicit LMM in tandem with an explicit LMM to form a predictor-corrector pair, the whole forming an explicit method. Unlike an ODE solver, we may construct derivative-free root-finders based on the LMM approach by setting all $b_k^{(n)} = 0$ for $k = 0, \ldots s-1$ and for all $n > 0$, e.g., the secant method. For an ODE solver this would obviously not make sense. Similar to ODE solvers, we may introduce higher derivatives of f by

[4] I thought this was an incredibly clever formulation of the root-finding problem.

using

$$x_{n+s}+\sum_{k=0}^{s-1} a_k^{(n)} x_{n+k} = h_{n+s}\sum_{k=0}^{s-1} b_k^{(n)} F(x_{n+k})+h_{n+s}^2\sum_{k=0}^{s-1} c_k^{(n)} F'(x_{n+k})+\dots \quad (3.9)$$

The following theorem provides the convergence rate for any method of the form (3.9). Furthermore, it provides a fundamental barrier on the convergence rates of LMM-based root-finders. Under certain conditions, it also rewards our intuition in the sense that methods using more information converge faster to the root.

Let's introduce some notation first. We denote with d the number of derivatives of f used in the method (3.9). Higher derivatives of the inverse are found by iteratively applying the inverse function derivative rule. Methods defined by (3.7) are the special case of (3.9) with $d = 1$. We also introduce coefficients σ_k that indicate whether the coefficients $a_k^{(n)}$ are arbitrarily fixed from the outset or left free to maximise the order of convergence, i.e., $\sigma_k = 1$ if $a_k^{(n)}$ is free and $\sigma_k = 0$ otherwise.

Theorem 3.1. *For simple roots, the convergence rate p for any method of the form (3.9), where the coefficients are chosen so as to give the highest order of convergence, is given by the largest real root of*

$$p^s = \sum_{k=0}^{s-1} p^k(d+\sigma_k), \quad (3.10)$$

for all $s \geq 1$ and $d \geq 1$, or $s \geq 2$ and $d = 0$. The convergence rate is bounded by $p < d + 2$. Additionally, if $\sigma_k = 1$ for all $k = 0, \dots, s-1$, the convergence rates form a monotonically increasing sequence in s and $p \to d + 2$ as $s \to \infty$.

Proof. 1. Any method of the form (3.9) implicitly uses inverse polynomial (Hermite) interpolation, i.e., polynomial interpolation applied to the inverse function f^{-1}. Let's call the resulting interpolation H. Since H interpolates the inverse, $H(0)$ is an approximation to the root. Let y_{n+k}, $k = 0, \dots, s-1$ be the interpolation points. At each point y_{n+k}, $d+\sigma_k$ values are interpolated, the inverse function value x_k if $\sigma_k = 1$ and d derivative values. Thus, the polynomial interpolation error formula, see e.g. [40], gives

$$f^{-1}(y) - H(y) = \frac{[f^{-1}]^{(N+1)}(v)}{(N+1)!}\prod_{k=0}^{s-1}(y-y_{n+k})^{d+\sigma_k},$$

where υ is in the interval spanned by the interpolation points and $N = sd + \sum_{k=0}^{s-1} \sigma_k$. The approximation to the root is then computed as $x_{n+s} = H(0)$. Let's denote the exact value of the root as $x^\star$, then

$$|x_{n+s} - x^\star| = \frac{|[f^{-1}]^{(N+1)}(\upsilon)|}{(N+1)!} \prod_{k=0}^{s-1} |y_{n+k}|^{d+\sigma_k}.$$

Define $\varepsilon_{n+k} = x_{n+k} - x^\star$ and recognise that $y_{n+k} = f(x_{n+k}) = f(x^\star + \varepsilon_{n+k}) = f'(x^\star)\varepsilon_{n+k} + \mathcal{O}(\varepsilon_{n+k}^2)$, where $f'(x^\star) \neq 0$. Thus, we find

$$|\varepsilon_{n+s}| \approx A_0 |\varepsilon_{n+s-1}|^{d+\sigma_{s-1}} \cdots |\varepsilon_n|^{d+\sigma_0},$$

where $A_0 > 0$ is a constant depending on $[f^{-1}]^{(N+1)}(\upsilon)$, s and $f'(x^\star)$. The error behaviour is of the form

$$|\varepsilon_{l+1}| = C|\varepsilon_l|^p, \qquad (*)$$

asymptotically as $l \to \infty$. Here, $C > 0$ is a constant. Applying $(*)$ s times on the left and $s-1$ times on the right-hand side leads to

$$|\varepsilon_n|^{p^s} \approx A_1 |\varepsilon_n|^{\sum_{k=0}^{s-1} p^k (d+\sigma_k)},$$

where all the constants have been absorbed into A_1. Thus, (3.10) is established.

2. For methods that only use a single point, we have $s = 1$ and $\sigma_0 = 1$ so that (3.10) simplifies to $p = d + 1$. Hence, also in the case $s = 1$, we have $p < d + 2$.

3. Finally, by its definition we can bound $\sigma_k \leq 1$, so that we obtain

$$p^s \leq (d+1) \sum_{k=0}^{s-1} p^k = (d+1) \frac{p^s - 1}{p - 1}.$$

Simplifying, we obtain

$$p^{s+1} - (d+2)p^s + d + 1 \leq 0. \qquad (\star)$$

Note that $p = 1$ is always a solution if we impose equality. However, the maximal convergence rate is given by the largest real root, so that we look for solutions $p > 1$. Dividing by p^s yields,

$$p - (d+2) \leq -\frac{d+1}{p^s} < 0,$$

which holds for all $s \geq 1$. Hence, we obtain $p < d + 2$.

4. Suppose now that $\sigma_k = 1$ for all $k = 0, \ldots, s-1$, so that the convergence rate satisfies $(\star)$ with equality. Rewriting, we obtain

$$d + 2 - p = \frac{d+1}{p^s}.$$

We can interpret this equation geometrically, where each side is considered a function of p. Hence, we see that the convergence rate is given by the intersection point of a straight line and an inverse power law. First, we observe that there is always an intersection at $p = 1$. Furthermore, the slope of the straight line is -1, while the slope of the inverse power law at $p = 1$ is given by $-s(d+1)$. Therefore, the slope of the right-hand side is smaller than the slope of the left-hand side for all $s \geq 1$ and $d \geq 1$, or $d = 0$ and $s \geq 2$. Thus, immediately to the right of $p = 1$, the inverse power law is below the straight line. Combined with the fact that $\frac{d+1}{p^s}$ is a convex function for $p > 0$ and $s > 0$, we see that a second intersection point exists and is unique. Fix s and call the second intersection point, i.e., the convergence rate, $p^*(s)$. Finally, we note that for $p > 1$, we have

$$\frac{d+1}{p^{s+1}} < \frac{d+1}{p^s},$$

so that $p^*(s+1)$ will be moved towards the right compared to $p^*(s)$. Thus, we find

$$p^*(s+1) > p^*(s)$$

for all $d \geq 1$ and $s \geq 1$, or $d = 0$ and $s \geq 2$. The result now follows from the Monotone Convergence Theorem [61]. □

From Theorem 3.1, we find several special cases, such as the derivative-free interpolation root-finders, i.e., $d = 0$. Note that derivative-free root-finders with $s = 1$ simply don't exist, as then the inverse is interpolated with a constant function. Constant functions, of course, don't have roots unless the constant happens to be zero. However, for a root-finder, this means it would only work if the exact root is already found, which isn't terribly useful.

Corollary 3.1. *Inverse polynomial interpolation root-finders, i.e., $d = 0$ resulting in all $b_k^{(n)} = 0$ in (3.7), can attain at most a convergence rate that is quadratic. Their convergence rates are given by the largest real root of*

$$p^{s+1} - 2p^s + 1 = 0, \tag{3.11}$$

for all $s \geq 2$. The convergence rates are bounded by $p < 2$ and form a monotonically increasing sequence in s, with $p \to 2$ as $s \to \infty$.

Proof. The coefficients $\{a_k^{(n)}\}_{k=0}^{s-1}$ are chosen to maximise the order of convergence, so that $\sigma_k = 1$ for all $k = 0, \ldots, s-1$, while $d = 0$, leading to

$$p^s = \sum_{k=0}^{s-1} p^k = \frac{p^s - 1}{p - 1}.$$

Simplifying yields (3.11). Furthermore, the condition that $\sigma_k = 1$ for all $k = 0, \ldots, s-1$ is satisfied so that the convergence rates form a monotonically increasing sequence that converges to $d + 2 = 2$. □

Inverse polynomial root-finders such as the secant method ($s = 2$) or inverse quadratic interpolation ($s = 3$) are derivative-free, so that their highest convergence rate is 2 according to Theorem 3.1. The first few convergence rates for derivative-free inverse polynomial interpolation methods are presented in Table 3.1. The well-known convergence rates for the secant method and the inverse quadratic interpolation method are indeed reproduced. As becomes clear from the table, the rates quickly approach 2 but never quite get there. The increase in convergence rate becomes smaller and smaller as we increase the number of interpolation points. The law of diminishing returns is unrelenting, it seems.

Table 3.1: The first few convergence rates for s points for derivative-free methods.

s	p
2	1.62
3	1.84
4	1.92
5	1.97

Next, we cover the Adams-Bashforth methods also discussed in [41]. As ODE solvers, Adams-Bashforth methods are explicit integration methods that have order of accuracy s [31]. However, as Theorem 3.1 suggests, as root-finders they will have a convergence rate that is smaller than cubic, since $d = 1$. In fact, the convergence rate of Adams-Bashforth root-finders is bounded by $\frac{3+\sqrt{5}}{2} = 2.62$ as was proven by Grau-Sánchez et al. [41]. The following corollary is a generalisation of their result.

Corollary 3.2. *The Adams-Bashforth root-finder methods with $s \geq 2$ exhibit convergence rates given by the largest real root of*

$$p^{s+1} - 3p^s + p^{s-1} + 1 = 0, \tag{3.12}$$

for all $s \geq 1$. The convergence rates are bounded by $p < \frac{3+\sqrt{5}}{2}$ and form a monotonically increasing sequence in s, with $p \to \frac{3+\sqrt{5}}{2}$ as $s \to \infty$.

Proof. 1. Adams-Bashforth methods have $a_k^{(n)} = 0$ for $k = 0, \ldots, s-2$, and $a_{s-1}^{(n)} = 1$, resulting in $\sigma_k = 0$ for $k = 0, \ldots, s-2$ and $\sigma_{s-1} = 1$. We may write σ_k for simplicity as a Kronecker delta, i.e., $\sigma_k = \delta_{k,s-1}$. Furthermore, the methods use a single derivative of f^{-1} so that $d = 1$. The $s = 1$ method is equal to Newton's method, which has a quadratic convergence rate. For $s \geq 2$, we find from Theorem 3.1 that

$$p^s = p^{s-1} + \sum_{k=0}^{s-1} p^k = p^{s-1} + \frac{p^s - 1}{p - 1}.$$

Simplifying yields (3.12). Again, we assume that $p > 1$ and we divide by p^{s-1}, so that

$$p^2 - 3p + 1 = -\frac{1}{p^{s-1}} < 0,$$

which holds for all $s \geq 2$. This implies that $p < \frac{3+\sqrt{5}}{2}$ for all $s \geq 2$.

2. The proof that the convergence rates make up a monotonically increasing sequence is similar to the one given for Theorem 3.1. First, we note that (3.12) always has a root at $p = 1$. Next, we rewrite it to read

$$3 - p = \frac{1}{p} + \frac{1}{p^s}.$$

The left-hand side is a straight line with slope -1 while the right-hand side is a convex function that has slope $-(1 + s)$ at $p = 1$. Thus, immediately to the right of $p = 1$, the convex function is below the straight line. This implies that there is a unique intersection point with $p > 1$, which is the convergence rate. For fixed s, call the second intersection point $p^*(s)$. Finally, we note that

$$\frac{1}{p} + \frac{1}{p^{s+1}} < \frac{1}{p} + \frac{1}{p^s}, \tag{3.13}$$

for $p > 1$, from which we see that the intersection point is moved to the right for larger s, i.e., $p^*(s+1) > p^*(s)$. The result again follows from the Monotone Convergence Theorem. □

The first few convergence rates for the Adams-Bashforth root-finder methods are given in Table 3.2 and agree with the rates found by Grau-Sánchez et al. As becomes clear from the table, the convergence rates quickly draw near the bound of 2.62. Yet again we're met with steeply diminishing returns as we increase the number of history points s.

Table 3.2: The first few convergence rates for Adams-Bashforth root-finder method using s points.

s	p
1	2
2	2.41
3	2.55
4	2.59
5	2.61

The Adams-Bashforth root-finder methods cannot attain a convergence rate higher than 2.62, which is still some way off the cubic bound given by Theorem 3.1. Using another linear multistep method may therefore result in convergence rates closer to cubic. For ODE solvers, trying to obtain a higher convergence rate by increasing the number of points often leads to instabilities. In fact, polynomial interpolation on equispaced points can even lead to diverging results, e.g., Runge's phenomenon [57]. However, root-finders generate a convergent set of interpolation points, thereby avoiding instabilities. As root-finders, adding more history points *does* result in higher convergence rates.

Let's inspect the convergence rates of different LMM-based root-finders using Theorem 3.1, see Table 3.3. These convergence rates are computed under the assumption that all derivatives and point values are used, i.e., $\sigma_k = 1$ for $k = 0, \ldots, s-1$ in Theorem 3.1. The convergence rate of a d-derivative method can be boosted by at most 1, and the table shows that this mark is attained very quickly indeed. Adding a few history points raises the convergence rate significantly, but finding schemes with $s > 3$ is likely to be a waste of time. Adding derivatives, however, is of course the way to obtain arbitrary convergence rates.

3.4 Full LMM-based root-finders

Let's investigate full LMM-based root-finders that use a single derivative, thus methods of the form (3.7). Recall that the current step size is given by $h_{n+s} =$

Table 3.3: The first few convergence rates for s points (vertical) using all function values and the first d derivatives (horizontal).

$s\backslash d$	1	2	3	4
1	2	3	4	5
2	2.73	3.79	4.82	5.85
3	2.91	3.95	4.97	5.98
4	2.97	3.99	4.99	5.996

$-f(x_{n+s-1})$. Let's define $q_k^{(n)}$ as

$$q_k^{(n)} = \frac{f(x_{n+k})}{f(x_{n+s-1})}, \quad k = 0, \ldots, s-2, \tag{3.14}$$

so that $h_{n+s}q_k^{(n)} = -f(x_{n+k})$ is the total step between y_{n+k} and $y_{n+s} = 0$. The Taylor expansions of $x(y_{n+k})$ and $x'(y_{n+k})$ about y_{n+s} are then given by

$$x(y_{n+k}) = x(y_{n+s}) + \sum_{m=1}^{\infty} \frac{1}{m!}(-h_{n+s}q_k)^m x^{(m)}(y_{n+s}), \tag{3.15a}$$

$$x'(y_{n+k}) = x'(y_{n+s}) + \sum_{m=1}^{\infty} \frac{1}{m!}(-h_{n+s}q_k)^m x^{(m+1)}(y_{n+s}), \tag{3.15b}$$

where we've dropped the superscript (n) for brevity. Substituting these into (3.7) with $x'(y) = F(x)$, we obtain

$$\begin{aligned} &x(y_{n+s})\left[1 + \sum_{k=0}^{s-1} a_k\right] - h_{n+s}x'(y_{n+s})\left[\sum_{k=0}^{s-1} a_k q_k + b_k\right] \\ &+ \sum_{m=2}^{\infty} \frac{1}{(m-1)!}(-h_{n+s})^m x^{(m)}(y_{n+s}) \sum_{k=0}^{s-1} \left[\tfrac{1}{m}q_k^m a_k + q_k^{m-1}b_k\right] = 0. \end{aligned} \tag{3.16}$$

Eliminating the leading-order terms in (3.16) gives what are referred to in the ODE context as the consistency conditions, i.e.,

$$\sum_{k=0}^{s-1} a_k = -1, \tag{3.17a}$$

$$\sum_{k=0}^{s-1} a_k q_k + b_k = 0. \tag{3.17b}$$

This gives us two equations for $2s$ coefficients, so that we can eliminate another $2s-2$ remaining terms in (3.16), resulting in the order conditions

$$\sum_{k=0}^{s-1} \frac{q_k^m}{m} a_k + q_k^{m-1} b_k = 0, \tag{3.18}$$

where $m = 2, \ldots, 2s-1$.

3.4.1 The $s = 2$ method

The $s = 2$ LMM-based method is given by

$$x_{n+2} + a_1 x_{n+1} + a_0 x_n = h_{n+2}\Big(b_1 F(x_{n+1}) + b_0 F(x_n)\Big), \tag{3.19}$$

where we've again suppressed the superscript (n) on the coefficients. Here, $h_{n+2} = -f(x_{n+1})$ so that we may write $q = q_0$, i.e.

$$q = \frac{f(x_n)}{f(x_{n+1})}. \tag{3.20}$$

Applying (3.17) and (3.18), we find a set of linear equations, i.e.,

$$a_1 + a_0 = -1, \tag{3.21a}$$
$$a_1 + q a_0 + b_1 + b_0 = 0, \tag{3.21b}$$
$$\tfrac{1}{2} a_1 + \tfrac{1}{2} q^2 a_0 + b_1 + q b_0 = 0, \tag{3.21c}$$
$$\tfrac{1}{3} a_1 + \tfrac{1}{3} q^3 a_0 + b_1 + q^2 b_0 = 0. \tag{3.21d}$$

These equations are solved, provided $q \neq 1$, to yield

$$a_0 = \frac{1-3q}{(q-1)^3} \qquad a_1 = -1 - a_0, \tag{3.22a}$$
$$b_0 = \frac{q}{(q-1)^2} \qquad b_1 = q b_0. \tag{3.22b}$$

The condition $q \neq 1$ is equivalent to $f(x_{n+1}) \neq f(x_n)$. This condition isn't very restrictive, as stronger conditions are needed to ensure convergence.

The above method may also be derived from the inverse polynomial interpolation perspective, using the ansatz

$$H(y) = h_3\big(y - f(x_{n+1})\big)^3 + h_2\big(y - f(x_{n+1})\big)^2 + h_1\big(y - f(x_{n+1})\big) + h_0, \tag{3.23}$$

where h_i, $i = 0, 1, 2, 3$ are undetermined coefficients. The coefficients are fixed by demanding that H interpolates f^{-1} and its derivative at $y = f(x_{n+1})$ and $y = f(x_n)$, i.e.,

$$H\big(f(x_n)\big) = x_n, \tag{3.24a}$$

$$H\big(f(x_{n+1})\big) = x_{n+1}, \tag{3.24b}$$

$$H'\big(f(x_n)\big) = \frac{1}{f'(x_n)}, \tag{3.24c}$$

$$H'\big(f(x_{n+1})\big) = \frac{1}{f'(x_{n+1})}. \tag{3.24d}$$

Solving for h_i, $i = 0, 1, 2, 3$ and setting $y = 0$, we find the same update x_{n+2} as (3.19) with coefficients given by (3.22).

The stability of the $s = 2$ LMM method depends on the coefficients of the LMM in much the same way as an ODE solver. Indeed, we can set the sequence $\tilde{x}_n = x_n + z_n$ where x_n is the sequence generated by exact arithmetic we wish to find while z_n is a parasitic mode. The undesired parasitic mode is the homogeneous solution of (3.19), i.e.,

$$z_{n+2} + a_1 z_{n+1} + a_0 z_n = 0, \tag{3.25}$$

so that it will grow unbounded if the roots are greater than 1 in modulus. Using the ansatz $z_n = B\lambda^n$ and the fact that $a_1 = -(1+a_0)$, we find the characteristic polynomial of the $s = 2$ method, i.e.,

$$\rho(\lambda) = \lambda^2 - \lambda\,(1 + a_0) + a_0 = (\lambda - 1)\,(\lambda - a_0)\,, \tag{3.26}$$

where the roots can simply be read off. Stability of the root-finder is ensured if the stability polynomial of the method has a single root with $\lambda = 1$, while the other roots satisfy $|\lambda| < 1$. This property is called zero-stability for linear multistep ODE solvers. Thus, to suppress parasitic modes we need

$$|a_0| = \left|\frac{1 - 3q}{(q-1)^3}\right| < 1. \tag{3.27}$$

This reduces to q being either $q < 0$, or $q > 3$, so that $|q| > 3$ is a sufficient condition. Thus, if the sequence $\{|f(x_n)|\}_{n=1}^{\infty}$ is decreasing fast enough, any parasitic mode is suppressed. We estimate q as a ratio of errors, since for simple roots we have $f(x_n) = f'(x^\star)\varepsilon_n + \mathcal{O}(\varepsilon_n^2)$, so that

$$q \approx \frac{\varepsilon_n}{\varepsilon_{n+1}}. \tag{3.28}$$

Using $\varepsilon_{n+1} = C\varepsilon_n^p$ with $p = 1 + \sqrt{3}$ given by Theorem 3.1, we find that

$$\left|\frac{\varepsilon_n}{\varepsilon_{n+1}}\right| = \frac{1}{C}|\varepsilon|^{-\sqrt{3}} > 3. \tag{3.29}$$

Rearranging terms, we discover that we should satisfy the condition

$$|\varepsilon_n| < C_1, \tag{3.30}$$

with $C_1 = \left(\frac{1}{3C}\right)^{\frac{1}{\sqrt{3}}}$. Furthermore, for each iteration, $|q|$ will increase since

$$\left|\frac{\varepsilon_{n+1}}{\varepsilon_{n+2}}\right| = \left|\frac{\varepsilon_n}{\varepsilon_{n+1}}\right|^{1+\sqrt{3}}.$$

Hence, if $|q| > 3$ for the first iteration, all following iterations will have $|q| > 3$. We conclude that the method will be stable if the initial errors are smaller than the constant C_1, which depends on the details of the function f in the vicinity of the root. This condition translates to having the starting values sufficiently close to the root. This is a rather typical stability condition for root-finders.

3.4.2 The $s = 3$ method

We again apply (3.17) - (3.18) to find a method with $s = 3$, this time there are 6 coefficients, given by

$$a_0 = \frac{q_1^2(q_0(3 + 3q_1 - 5q_0) - q_1)}{(q_0 - 1)^3(q_0 - q_1)^3}, \qquad b_0 = \frac{q_0 q_1^2}{(q_0 - 1)^2(q_0 - q_1)^2}, \tag{3.31a}$$

$$a_1 = \frac{q_0^2(q_1(5q_1 - 3q_0 - 3) + q_0)}{(q_1 - 1)^3(q_0 - q_1)^3}, \qquad b_1 = \frac{q_0^2 q_1}{(q_0 - q_1)^2(q_1 - 1)^2}, \tag{3.31b}$$

$$a_2 = \frac{q_0^2 q_1^2(3q_1 - q_0(q_1 - 3) - 5)}{(q_0 - 1)^3(q_1 - 1^3)}, \qquad b_2 = \frac{q_0^2 q_1^2}{(q_0 - 1)^2(q_1 - 1)^2}, \tag{3.31c}$$

where $q_0 = \frac{f(x_n)}{f(x_{n+2})}$ and $q_1 = \frac{f(x_{n+1})}{f(x_{n+2})}$. Here, we have the conditions $q_0 \neq 1$ and $q_1 \neq 1$, reducing to the condition that all y-coordinates must be unique. Again, this condition isn't very restrictive for reasons detailed above.

Methods with a greater number of history points are possible, however, the gain in convergence rate from $s = 3$ to $s = 4$ is rather slim, as indicated by Table 3.3. If such methods are desirable, they can be derived by selecting coefficients that satisfy (3.17) - (3.18).

3.5 Comparison with Newton's method

Like the secant method, the $s = 2$ full LMM root-finding method needs two starting points for the iteration. However, as the analytical derivative is available, we choose to simply start the LMM-based method using Newton's method on the first point, say x_0, giving x_1. The $s = 3$ method needs three starting values, therefore the next value x_2 is obtained from the $s = 2$ LMM method. The LMM-based methods can be efficiently implemented by storing the function value and derivative value of the previous step, thus resulting in a need for only one function and one derivative evaluation per iteration.

Recall that the efficiency measure is defined as $p^{\frac{1}{w}}$ with p the order of convergence and w the number of evaluations per iteration [40]. Assuming the function itself and the derivative cost the same to evaluate, the $s = 3$ LMM-based method has an efficiency measure of $\sqrt{2.91} \approx 1.71$, using the convergence rate from Table 3.3. Compared to Newton's method, with an efficiency measure of $\sqrt{2} \approx 1.41$, this is certainly an improvement. Even compared to memoryless methods that have an optimal efficiency of $2^{\frac{w-1}{w}}$ according to the Kung-Traub conjecture, our $s = 3$ method still holds up for $w = 4$. Hence, in terms of the efficiency measure, our method should weigh up to eighth-order optimal methods. It would take a sixteenth-order optimal method, such as Geum and Kim's [62], to produce a higher efficiency.

3.5.1 Numerical examples

Here, we provide a number of test cases and show the number of iterations LMM-based root-finders take versus Newton's method, see Table 3.4. We've used a selection of different test cases with polynomials, exponentials, trigonometric functions, square roots and combinations thereof. For each of the test problems shown, the methods converged within a few iterations. Some problems were deliberately started near a maximum or minimum to see the behaviour when the derivatives are small.

The test computations were performed using the variable-precision arithmetic of MATLAB's Symbolic Math Toolbox. The number of digits was set to 300 while the convergence criterion used was

$$|x_{l+1} - x_l| \leq 10^{-\eta}, \tag{3.32}$$

with $\eta = 250$. The numerical convergence rates were computed with the error behaviour

$$|\varepsilon_{l+1}| = C|\varepsilon_l|^p, \tag{3.33}$$

asymptotically as $l \to \infty$. The limiting value of the estimates for p is displayed in Table 3.4.

Table 3.4: Test cases with iterations taken for Newton's method (subscript N) and the LMM-based method with $s = 2$ (subscript 2) and $s = 3$ (subscript 3) method.

function	root	x_0	$\#\text{its}_N$	$\#\text{its}_2$	$\#\text{its}_3$	p_2	p_3
$x + e^x$	-0.57	1.50	11	8	8	2.73	2.93
$\sqrt{x} - \cos(x)$	0.64	0.50	9	7	8	2.74	2.91
$e^x - x^2 + 3x - 2$	0.26	0.00	10	8	7	2.72	2.94
$x^4 - 3x^2 - 3$	1.95	1.30	17	14	14	2.73	2.92
$x^3 - x - 1$	1.32	1.00	12	9	9	2.73	2.64
$e^{-x} - x^3$	0.77	2.00	13	10	10	2.73	2.92
$5\big(\sin(x) + \cos(x)\big) - x$	2.06	1.50	11	9	9	2.73	2.92
$x - \cos(x)$	0.74	1.00	9	7	7	2.72	2.93
$\log(x-1) + \cos(x-1)$	1.40	1.60	12	9	9	2.73	2.92
$\sqrt{1+x} - x$	1.62	1.00	9	7	7	2.73	2.92
$\sqrt{e^x - x} - 2x$	0.54	1.00	11	8	7	2.73	2.92
Total number of iterations			124	96	95		

We didn't display Newton's method in the table since it consistently exhibited a quadratic convergence rate. The LMM-based methods, on the other hand, generally have a higher convergence rate that may vary somewhat from problem to problem. This is due to the fact that the step sizes may vary slightly while the convergence rates of the LMM-based methods only holds asymptotically, even with so many digits.

3.5.2 Pathological functions

A classical example of a pathological function for Newton's method is the hyperbolic tangent $\tanh(x)$. Undergraduates newly introduced to Newton's method often believe that it converges for any monotone function until they're asked to find the root of $\tanh(x)$. Hence, we've used this function as a test case using standard double precision floating point arithmetic and a convergence criterion reading

$$|x_{l+1} - x_l| \leq 2\epsilon_{\text{mach}}, \tag{3.34}$$

with ϵ_{mach} the machine precision. Newton's method fails to converge for starting values with approximately $|x_0| \geq 1.089$, see Table 3.5. Our $s = 2$ LMM-based

Table 3.5: Convergence history for $\tanh(x)$ of the three methods: Newton (subscript N) and the LMM-based methods with $s = 2$ and $s = 3$. Note that the root of $\tanh(x)$ is at $x = 0$.

x_N	$x_{(s=2)}$	$x_{(s=3)}$
1.239	1.239	1.239
-1.719	-1.719	-1.719
6.059	0.8045	0.8045
$-4.583 \cdot 10^4$	0.7925	-0.6806
Inf	-0.7386	1.377
	$-6.783 \cdot 10^{-3}$	-0.7730
	$9.323 \cdot 10^{-6}$	$3.466 \cdot 10^{-2}$
	$\lvert x_{(s=2)}\rvert < \epsilon$	$-3.032 \cdot 10^{-4}$
		$1.831 \cdot 10^{-11}$
		$\lvert x_{(s=3)}\rvert < \epsilon$

method extends this range somewhat more and converges for any starting value with roughly $|x_0| \leq 1.239$. This value was determined experimentally by trial-and-error. The behaviour of the $s = 3$ LMM-based method is similar, though it does take two more iterations to converge.

Starting at $x_0 = 1.239$, Newton's method diverges quickly, returning $-\text{inf}$ after only 4 iterations. The LMM-based method, on the other hand, bounces around positive and negative values for about 5 iterations until it's close enough to the root. After that, the asymptotic convergence rate sets in and the root is quickly found, reaching the root within machine precision at 7 iterations.

Donovan et al. [63] developed another test function for which Newton's method fails. The test produces a false convergence result for any initial guess other than the root $x = 0$, satisfying (3.34) while the sequence x_k itself diverges very slowly. The test function is given by

$$h(x) = \sqrt[3]{x} e^{-x^2}, \tag{3.35}$$

which is, in fact, infinitely steep near the root $x = 0$, though it's continuous, see Figure 3.1. Again, we used double precision arithmetic and (3.34) as a stopping criterion. Newton's method slowly moves away from the root with a decreasing step size, eventually resulting in satisfying (3.34) without having found the root. The $s = 2$ and $s = 3$ LMM-based methods converge when starting with $|x_0| \leq 0.1147$ for this problem, see Table 3.6. Again, this starting value was determined experimentally by trial-and-error.

Table 3.6: Convergence history for $h(x)$ from (3.35) of the three methods: Newton (subscript N) and the LMM-based methods with $s = 2$ and $s = 3$. Note that the root is at $x = 0$.

x_N	$x_{(s=2)}$	$x_{(s=3)}$
0.1147	0.1147	0.1147
-0.2589	-0.2589	-0.2589
1.0402	0.1016	0.1016
1.6084	$9.993 \cdot 10^{-2}$	$-5.648 \cdot 10^{-2}$
1.9407	-0.2581	0.1959
2.2102	$9.840 \cdot 10^{-2}$	-0.1611
2.4445	$9.810 \cdot 10^{-2}$	$5.021 \cdot 10^{-2}$
2.6549	-0.2344	$-7.190 \cdot 10^{-2}$
2.8478	$6.602 \cdot 10^{-2}$	$4.947 \cdot 10^{-2}$
3.0270	$6.021 \cdot 10^{-2}$	$-3.777 \cdot 10^{-3}$
3.1953	$-4.939 \cdot 10^{-2}$	$3.027 \cdot 10^{-4}$
3.3543	$-4.019 \cdot 10^{-4}$	$-6.875 \cdot 10^{-6}$
3.5056	$1.288 \cdot 10^{-4}$	$1.216 \cdot 10^{-9}$
3.6502	$2.028 \cdot 10^{-10}$	$-4.652 \cdot 10^{-15}$
3.7889	$-5.308 \cdot 10^{-15}$	$\lvert x_{(s=2)} \rvert < \epsilon$
3.9225	$\lvert x_{(s=2)} \rvert < \epsilon$	

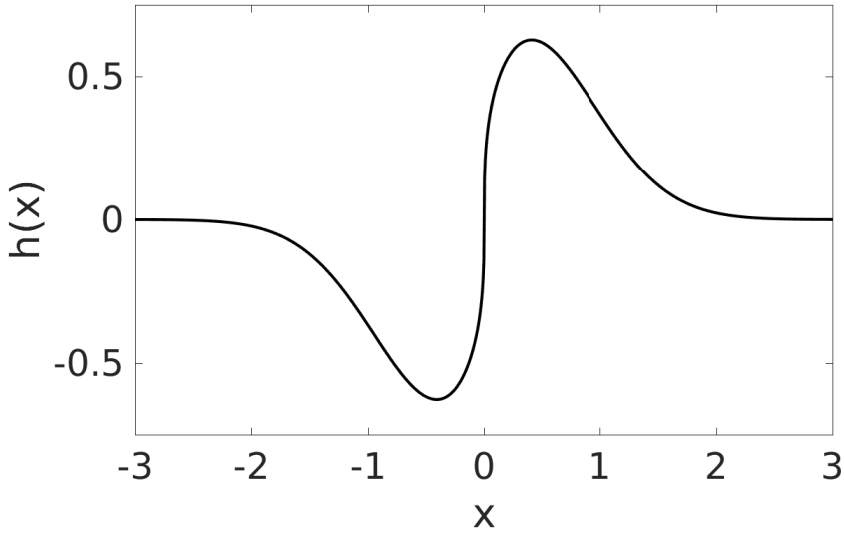


Figure 3.1: Pathological test function $h(x)$ from (3.35).

Starting at the maximal $x_0 = 0.1147$ for instance, the LMM-based methods bounce several times between positive and negative x-values without making much headway. After that, the root is close enough and asymptotic convergence sets in, reaching the root to within machine precision in a few steps.

We believe the reason that the LMM-based method has increased stability is due to the fact that it uses two points to evaluate the function and its derivative. In both cases, the iterations jump between positive and negative values, enclosing the root. In this fashion, the LMM-based method acts much like the *regula falsi* method. Once the iterates are close enough to the root, the asymptotic convergence rate sets in and the iterates converge in but a few steps.

3.6 A robust implementation

As with most open root-finding algorithms, the conditions under which the method is guaranteed to converge are rather restrictive. Therefore, we've designed a bracketed version of the LMM-based method that's guaranteed to converge. The algorithm is based on Brent's method, using similar conditions to

catch either slow convergence or runaway divergence. This version of the LMM-based method does, however, require an enclosing bracket int$[a, b]$ on which the function changes sign, i.e., $f(a)f(b) < 0$. Alternatively, such a method can start out as an open method, switching to the bracketed method once a sign change is detected.

The algorithm consists of a cascade of methods increasing in accuracy but decreasing in robustness, again analogous to Brent's method. At the lowest level stands the most robust method, bisection, guarding against steps outside the current search bracket. On the highest level we use the full $s = 3$ LMM-based method discussed in the previous section. Thus, in the best possible case, the method will converge with a rate of 2.91 with an efficiency measure of 1.71. The method is, by virtue of the bisection method, guaranteed to converge to a root.

Like Brent's method and Dekker's method, the LMM-based method keeps track of three points a, b and c. Here, b is the best estimate of the root so far, c is the previous value for b while a is the contrapoint so that int$[a, b]$ encloses the root. Ideally, all three values are used to compute the next value for b. However, extra conditions are added to ensure the inverse actually makes sense on the interval int$[a, c]$.

Consider the case where the sign of $f'(c)$ is not equal to the sign of $\frac{f(b)-f(a)}{b-a}$, but the sign of $f'(b)$ is, see Figure 3.2. It follows that there is an extremum between b and c, and the inverse function does not exist in the entire interval int$[a, c]$, leading to an error if we were to compute the inverse interpolation. By discarding derivative information at c, we can define an inverse function that makes sense on the entire interval.

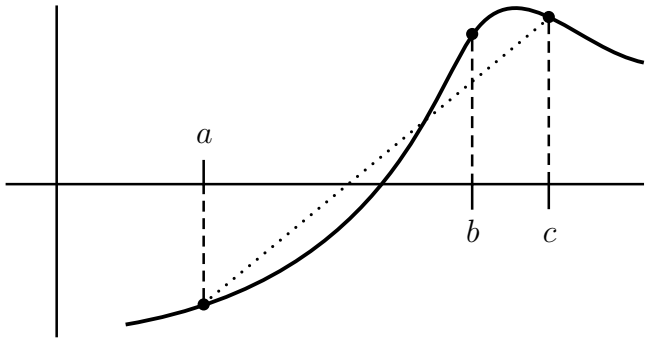


Figure 3.2: Sketch of the three points and the chord (dotted line) over $[a, c]$. The function f has no inverse on the entire interval.

From the previous argument, we see that the following condition should be applied to each derivative value: the sign of the derivative needs to be the same as the sign of the secant slope on $\text{int}[a, b]$, i.e.,

$$\text{sgn}\left(f'(z)\right) = \text{sgn}\left(\frac{f(b) - f(a)}{b - a}\right), \tag{3.36}$$

where $z = a, b, c$. The derivative at each point can only contribute sensibly if the derivative at a point satisfies (3.36). Otherwise we'd be trying to compute an inverse to a function that doesn't have one. If (3.36) is violated, the derivative information should be discarded, leading to a lower-order interpolation and a root-finder with a smaller convergence rate. If all derivatives are discarded, the algorithm switches to inverse quadratic interpolation or the secant method.

Ultimately, the method provides an interval on which the function f changes sign with a relative size of some given tolerance ϵ, i.e.,

$$|a - b| \leq \epsilon |b|. \tag{3.37}$$

We'll use $\epsilon = 2\epsilon_{\text{mach}}$ in all our examples, with ϵ_{mach} the machine precision. As an input, the algorithm has f, f', a and b such that $f(a) \cdot f(b) < 0$. The algorithm can be described in the following way:

1. If all three function values are different, use $s = 3$, otherwise use $s = 2$.
2. Check the sign of the derivatives at each point a, b and c, discard the derivative if (3.36) is violated.
3. Try interpolation. If interpolation is worse than bisection, or outside the interval $\text{int}[a, b]$, use bisection.
4. If the step is smaller than the tolerance, use the tolerance as step size.
5. If the convergence criterion is met, exit, otherwise go to 1.

Moreover, the usual definitions of a, b and c are maintained as invariants throughout the algorithm by making suitable swaps. In the algorithm, the first step determines the number of history points that can be used. The second step determines which derivative values should be taken into account. In effect, only the second step is essentially different from Brent's method, with all the following steps exactly the same. The details of determining if the interpolation step is worse than bisection are rather technical, but they allow for a worst-case performance of twice the number of steps as bisection [44].

The conditions on the derivatives gives rise to a veritable smörgåsbord of possible root-finders, including inverse quadratic interpolation and the secant method. For each point, there are two alternatives, either the derivative is used or not. Hence, for $s = 3$, there are $2^3 = 8$ possible combinations and for $s = 2$ there are $2^2 = 4$ possible combinations. Each combination defines a separate root-finder, resulting in a total of 12 possibilities. Our method can therefore be seen as stacking another 10 options on top of Brent's method. The methods are in fact exactly the same when none of the derivatives are suitable for use. Naturally, sufficiently close to a simple root, the derivative conditions will be satisfied at all three points and the method will use the full LMM method with $s = 3$.

3.6.1 Comparison with Brent's method

Here, we give a few examples of the robust LMM-based root-finding algorithm discussed above compared to Brent's method. As a performance measure, we use the number of iterations. Standard double precision arithmetic is employed, as that provides sufficient material for comparison. For both methods, the stopping criterion is given by (3.37), i.e., the relative size of the interval must be sufficiently small.

Table 3.7 shows that for most functions, both Brent's method and the LMM-based method take a comparable number of iterations. However, in some cases, the difference is considerable. In the worst case considered here, or rather the most favourable for our algorithm, Brent's method takes 7.5 times as many iterations to converge. In terms of efficiency index, Brent's method should be superior with an efficiency index of 1.84 against 1.71 of the LMM-based method. Taken over the whole set of test functions, however, Brent's method takes more than three times as many iterations in total, leading to a significant increase in function evaluations. We conclude therefore that practically, the LMM-based root-finder is a better choice.

3.7 Concluding remarks on ray tracing

Our discussion on algorithms for ray tracing has come to an end. We started the previous chapter by presenting a philosophy of high reliability. As such, our aim was focussed at building robust and efficient tools for ray tracing. In the previous chapter we've explored algorithms that reliably provide a good search bracket for the correct intersection between a ray and a surface. In this chapter,

Table 3.7: Test cases with iterations taken for Brent's method and the LMM-based method. Subscript B represents Brent while subscript LMM represents the LMM-based method.

function	root	$[a, b]$	$\#\text{its}_B$	$\#\text{its}_{LMM}$
$x + e^x$	-0.57	$[-1, 1]$	6	4
$\sqrt{x} - \cos(x)$	0.64	$[0, 2]$	8	4
$e^x - x^2 + 3x - 2$	0.26	$[-1, 1]$	5	3
$x^4 - 3x^2 - 3$	1.95	$[1, 3]$	10	8
$x^3 - x - 1$	1.32	$[0, 2]$	29	6
$e^{-x} - x^3$	0.77	$[0, 2]$	9	4
$5\big(\sin(x) + \cos(x)\big) - x$	2.06	$[0, 4]$	45	6
$x - \cos(x)$	0.74	$[0, 1]$	7	3
$\log(x - 1) + \cos(x - 1)$	1.39	$[1.2, 1.6]$	31	4
$\sqrt{1 + x} - x$	1.62	$[0, 2]$	5	3
$\sqrt{e^x - x} - 2x$	0.54	$[-1, 2]$	9	4
Total number of iterations			164	49
Total number of function evaluations			164	98

we've provided an equally impeccable root-finder that can swiftly close in on the root.

Overall, then, our foray into the classical approach to illumination optics, namely ray tracing, has been quite successful. However, to truly discover new lands and virgin territories, we must look towards new avenues of attack. The rest of this thesis is therefore dedicated to Liouville's equation, which is the topic of the next chapter.

Chapter 4

Liouville's equation

> Ich unterstelle natürlich Leser, die etwas Neues lernen, also auch selbst denken wollen.
>
> Karl Marx

Joseph Liouville.

So far, we've looked at a ray-tracing approach to computational illumination optics. In the industry, ray tracing is indeed the go-to method for solving such problems. However, if we're serious in our desire to find new, and hopefully faster, methods in computational optics, we need to try and break away from this paradigm. Alternative approaches may avoid the fundamental problems that plague ray tracing.

The first issue to avoid is sensitivity to the set of initial conditions. For any method based on ray tracing, the choice of initial conditions can have a big influence on the result. Approaches where the initial conditions of the rays are chosen in a regular grid on phase space often give bad results. Randomly drawing from a uniform distribution, called Monte Carlo ray tracing, is a popular option [23, 24, 64]. However, Monte Carlo ray tracing converges rather slowly. Moreover, using a stochastic method to approximate a deterministic quantity always results in a random error. The solutions computed by Monte Carlo ray tracing exhibit some

amount of noise.

The second issue is energy conservation. Methods based on ray tracing will always need a reconstruction technique to compute the energy distribution. One reasonable request for the reconstruction would be that it's conservative, i.e., the total reconstructed energy should equal the total source energy. However, typical reconstruction techniques, such as bin-count, don't conserve energy! Using forward ray tracing, it would be very hard, if not impossible, to construct a conservative method. Conservative ray tracing schemes do exist, but they use a combination of forward and backward ray tracing [65].

These two issues can be circumvented entirely by switching to methods based on Liouville's equation, which in the context of geometric optics can be interpreted as the statement of conservation of energy. In the absence of attenuation or diffusion, the energy carried by a ray is constant. If we assume that the optical situation is static and time doesn't play a role, power is also constant along rays. As mentioned already in Section 1.2.4, the light source may also move slowly with respect to the time it takes light to go through the optic, which practically means any lighting situation. Indeed, the speed of light may as well be infinite as far as human perception is concerned.

We can define a power density on phase space, ρ, which is power per unit area per solid angle. The power carried by a ray is then given by $\rho\,\mathrm{d}y$ with $\mathrm{d}y$ the phase space volume element of the ray. Conservation of étendue implies that the volume of $\mathrm{d}y$ is constant. Therefore, since power is constant along a ray, we should have that

$$\rho\big(z+\Delta z, \mathbf{q}(z+\Delta z), \mathbf{p}(z+\Delta z)\big) = \rho\big(z, \mathbf{q}(z), \mathbf{p}(z)\big), \tag{4.1}$$

which must hold for any Δz. In radiometric terms, ρ is the radiance, while in photometric terms it's called the *brightness* or *luminance*. Photometric quantities are scaled with the sensitivity of the human eye. Hence, we can interpret ρ either way, the difference being only a constant factor.

This equation, (4.1), is probably the most important equation in this thesis, because it characterises physical solutions to Liouville's equation. Its most significant feature is that it also holds even when the rays are refracted or reflected, since it's derived from the very physics of rays itself. Assuming sufficient smoothness, we may subtract the right-hand side and divide by Δz. The limit as $\Delta z \to 0$, yields

$$\frac{\mathrm{d}}{\mathrm{d}z}\rho\big(z, \mathbf{q}(z), \mathbf{p}(z)\big) = \frac{\partial \rho}{\partial z} + \frac{\partial h}{\partial \mathbf{p}} \cdot \frac{\partial \rho}{\partial \mathbf{q}} - \frac{\partial h}{\partial \mathbf{q}} \cdot \frac{\partial \rho}{\partial \mathbf{p}} = 0, \tag{4.2}$$

where we've applied Hamilton's equations (1.33).

Now we can interpret $\mathbf{q}$ and $\mathbf{p}$ as fixed coordinates and (4.2) becomes a partial differential equation that determines the evolution of ρ. Another way to put it's that Hamilton's equations provide the characteristics of Liouville's equation. This is the strong form of Liouville's equation, which requires ρ and h to be differentiable to make any sense[1]. Near an optical interface, (4.2) is not valid as everything ceases to be smooth, but (4.1) provides us with the correct physical solution. In particular, taking limits from both sides of the interface yields

$$\rho\big(z^+, \mathbf{q}^+, \mathbf{p}^+\big) = \rho\big(z^-, \mathbf{q}^-, \mathcal{S}(\mathbf{p}^-)\big), \tag{4.3}$$

where $\mathcal{S}$ is Snell's function from Theorem 1.2. The pluses and minuses in the superscript denote one-sided limits toward the interface. This equation simply says that across an interface, the ray carries along its energy from one side to the other, even though the ray itself is affected by Snell's law or the law of specular reflection.

Finally, (4.1) also allows us to find the exact solution to (4.2) based on the initial conditions. In particular, we set $\Delta z = -z$ and find

$$\rho\big(z, \mathbf{q}(z), \mathbf{p}(z)\big) = \rho_0\big(\mathbf{q}(0), \mathbf{p}(0)\big), \tag{4.4}$$

where ρ_0 is the initial brightness distribution. Again, we note that (4.4) is valid even when optical interfaces are present. We'll use it to derive analytical solutions to Liouville's equation for certain special cases in Section 4.2.

The brightness, ρ, is one of the most fundamental quantities of illumination optics. If we know ρ for a given source, we can pretty much compute anything we'd like to know. For instance, the *luminous intensity*, the luminous power per solid angle, is obtained by integrating ρ over all positions, i.e.,

$$I(\mathbf{p}) = \int_Q \rho \, \mathrm{d}q. \tag{4.5}$$

Recall that the momenta are related to angles defined with respect to the optical axis, so that (4.5) can be converted to an angular distribution if so desired. On the other hand, if we're interested in the *illuminance*, the luminous power per squared meter, we simply integrate over all momenta, yielding

$$E(\mathbf{q}) = \int_P \rho \, \mathrm{d}p. \tag{4.6}$$

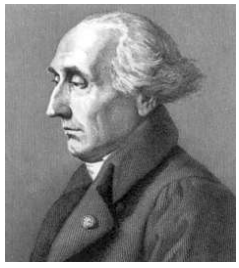
[1]It's funny that Liouville's equation was never even considered by Joseph Liouville himself. It was first derived by Josiah Willard Gibbs in [66]. He used an identity discovered by Liouville in [67].

Note that the converse is not necessarily true, if we know both the luminous intensity I and the illuminance E, the brightness ρ can in general not be reconstructed. In some sense, therefore, ρ provides more information than both I and E combined. The difference is that ρ is a phase space distribution, while phase space provides a complete description of geometric optics. Just consider the fact that Hamilton's equations (1.33) compose a first-order system of ODEs. Therefore, a single point in phase space provides all necessary and sufficient information to recover the entire ray. On the other hand, the ray equation (1.19) is a second-order ODE, so that simply providing an initial position is insufficient.

Information on phase space, therefore, provides everything there is to know about a luminaire in the context of geometric optics. As such, considering illumination optics on phase space brings certain advantages over considering only intensity or illuminance or even both. Rausch and Herkommer were to first to point this out [68, 69].

4.1 From Lagrange to Euler

What we did in going from Hamilton's equations (1.33) to Liouville's equation (4.2) was switch from a moving frame of reference to a fixed reference frame in describing what happens to the energy. In the moving frame situation, we travel along with the ray, tracing its position in phase space while keeping the energy carried by the ray in the back of our minds. In the other case, we have an independent coordinate system and we see some energy pass by as it's carried along by rays.

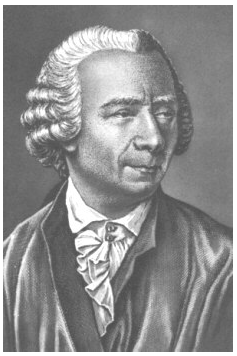


Joseph-Louis Lagrange.

The moving frame picture is called a Lagrangian description while the other is the Eulerian description. One old saw of mathematics holds that Lagrange is the kind of person who likes to sit in a boat adrift on a river and describe what happens directly around him, while Euler likes to stand on the shore describing what comes by. Another way of putting it's that Lagrangian information is local, while Eulerian information is global.

In illumination optics, we are interested in global information. We wish to know the entire energy distribution on a wall, for instance. If we compute Lagrangian (local) information by ray tracing, we therefore have to somehow convert it to Eulerian (global) information. This is precisely the reconstruction

step. Solving Liouville's equation (4.2) directly means we do all the work in the Eulerian frame and consequently there is no need for a reconstruction step.



Leonhard Euler.

If we think about it a little bit, we see that the conversion step to global information is what introduces all the errors for Lagrangian methods. The ray location itself can be computed exactly when dealing with optical interfaces, given exact arithmetic. On a computer, this means the error will be close to machine precision. However, performing a bin-count, for instance, wastes all this exactness since it's a first-order reconstruction method. It produces a piecewise constant approximation indiscriminate of the accuracy of each individual ray.

In the Eulerian picture, Liouville's equation cannot be solved exactly, but it can be numerically approximated to high accuracy. By trading near-exact computation for high-order accuracy, we can let go of the reconstruction step entirely. Eulerian methods produce higher-order global information directly. For illumination optics, therefore, it makes much more sense to focus on Eulerian methods.

Another consequence is that no special integrators are needed. For Eulerian methods the reference frame is fixed, so the structure of phase space is preserved no matter what. By switching to the Eulerian picture, we only need to worry about energy conservation. The reasoning is the following, consider some subset of phase space, Ω and its indicator function $\mathbb{1}_\Omega$, i.e., the function that has value 1 for every point in Ω and 0 elsewhere. Conservation of étendue, volume on phase space implies, among other things, that the integral over phase space is constant when $\mathbb{1}_\Omega$ is used as an initial condition for Liouville's equation. Furthermore, this condition must hold for all possible Ω. Hence, conservative numerical methods in the Eulerian frame fulfil more or less the same role as symplectic integrators in the Lagrangian frame.

4.2 Analytical solutions

Liouville's equation admits analytical solutions only in the very simplest of problems. These are constructed by means of an analytical version of ray tracing, known as the method of characteristics in the context of hyperbolic PDEs [70, 71]. It is, in principle, possible to wrench out a closed-form expres-

sion for any number of optical interfaces, provided the exact intersection with an arbitrary ray can be found. In practice, however, the staggering complications in cases occurring in the solution will compound swiftly and without mercy. For this reason, we consider here only simple propagation in a constant medium and a single flat interface. Such analytical solutions are nevertheless important as test cases to verify the numerical solvers we propose in later chapters.

4.2.1 Propagation in a constant medium

The first special case we consider is a constant refractive index. In this case, ρ is simply propagated along a single direction with constant speed. For a constant refractive index, the Hamiltonian is given by $h(\mathbf{p}) = -\sqrt{n^2 - |\mathbf{p}|^2}$, so that Liouville's equation reads

$$\frac{\partial \rho}{\partial z} + \frac{\mathbf{p}}{\sqrt{n^2 - |\mathbf{p}|^2}} \cdot \frac{\partial \rho}{\partial \mathbf{q}} = 0. \tag{4.7}$$

Note that this a linear advection equation with advection speed $\frac{\mathbf{p}}{\sqrt{n^2-|\mathbf{p}|^2}}$. This means that for every $\mathbf{p}$, the advection speed will be different. If we denote the initial condition with ρ_0, then the solution is given by

$$\rho(z, \mathbf{q}, \mathbf{p}) = \rho_0 \left(\mathbf{q} - z \frac{\mathbf{p}}{\sqrt{n^2 - |\mathbf{p}|^2}}, \mathbf{p} \right), \tag{4.8}$$

which can be easily checked.

4.2.2 The bucket of water

Another case which may be solved exactly is what we dub the bucket of water problem: a simple transition from refractive index n_1 to n_2 with a flat interface. The refractive index field is therefore given by

$$n(\mathbf{q}) = \begin{cases} n_1 & \text{if } q_1 \leq 0 \\ n_2 & \text{if } q_1 > 0, \end{cases} \tag{4.9}$$

where q_1 is the first component of $\mathbf{q}$ and $1 \leq n_2 < n_1$. This problem, for $n_1 = 1.4$ and $n_2 = 1$, resembles a torch being shone under water towards the flat water-air transition. As such, it represents the simplest problem that contains all of the important physics of light: propagation, refraction and total internal reflection.

Even though the problem is relatively simple and of no real practical interest, it represents an important test case that we'll use again and again as a benchmark throughout this work.

Clearly, this problem reduces to a two-dimensional optics problem, as the q_2-direction is parallel to the flat interface. The q_2-direction is therefore straightforward propagation described by a one-dimensional version of (4.8). To simplify the presentation, we'll switch to a two-dimensional optics setting, i.e., ignoring the q_2 and p_2-directions. We'll write $q = q_1$ and $p = p_1$, whence Liouville's equation becomes

$$\frac{\partial \rho}{\partial z} + \frac{p}{\sqrt{n(q)^2 - p^2}} \frac{\partial \rho}{\partial q} = 0, \tag{4.10}$$

for $q \neq 0$.

Remark. *A special note for optical engineers and physicists: the momentum is related to the choice of the optical axis, i.e., $p = n \sin \phi$, where ϕ is measured with respect to the optical axis, see Figure 4.1. In this case, the optical axis is parallel to the surface, rather than perpendicular to it*[2]*. Snell's law is formulated in terms of angles measured with respect to the surface normal. Therefore, in terms of ϕ, Snell's law becomes a cosine law.*

We formulate the bucket of water problem with this choice of optical axis since otherwise there'd be nothing much to see. Consider the more "natural" choice of choosing the z-axis normal to the interface. Since p is parallel to the flat interface at the point of impact, we have that $\frac{\partial n}{\partial q} = 0$ and Snell's law reduces to $p' = p$. In such a case, we'd apply Snell's law instantaneously to the whole distribution ρ in crossing the interface. As a consequence, the only difference between the phase space distributions before and after the interface would be that some part is reflected due to total internal reflection. Clearly, this situation wouldn't be much of an example. Hence the "strange" choice of axes.

For simplicity we'll also assume that there is only one source region, given by $\{q < 0, p \geq 0\}$. The initial condition can only have non-zero values in the source region. To find how the two regions left and right of the interface are linked, we have to apply Snell's law at the interface. For completeness, we'll also quote Snell's law in its two-dimensional form, given by

$$\mathcal{S}(p; n_1, n_2, \nu) := \begin{cases} p - \left(\psi + \operatorname{sgn}(n_2)\sqrt{\delta}\right)\nu & \text{if } \delta \geq 0, \\ p - 2\psi\nu & \text{if } \delta < 0, \end{cases} \tag{4.11a}$$

[2] You don't want to know how many times I've had to explain this over the years. Comments ranged from "counterintuitive" to plain "stupid".

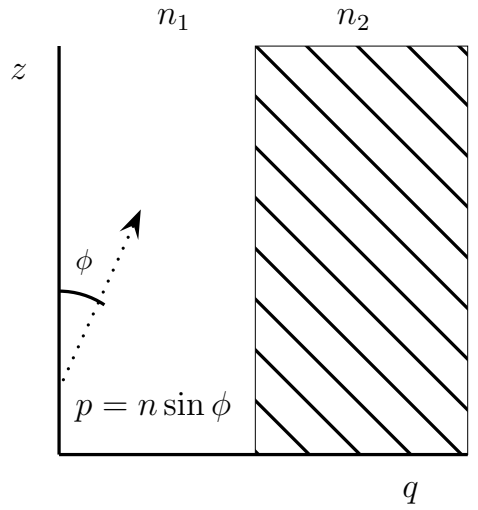


Figure 4.1: The momentum and its relation to the angle with respect to the optical axis ϕ.

where

$$\delta := n_2^2 - n_1^2 + \psi^2 \quad \text{and} \quad \psi := p\nu \pm \sqrt{n_1^2 - p^2}\sqrt{1-\nu^2}. \tag{4.11b}$$

Recall that the sign has to be chosen such that $\psi \leq 0$.

Let 's consider the source region first, $\{q < 0, p \geq 0\}$. In this region, only simple propagation occurs as the light beam doesn't encounter the interface. Second, we consider the region $\{q > 0, p < 0\}$, which is the lower-right quadrant of phase space. The advection speed in this region is negative, the flow is towards the left. This region therefore also exhibits only simple propagation. The only way any light could come here is by coming from further towards the right, but it happens to be completely dark there by assumption, i.e., $\rho = 0$. This region will be called the dark region. The situation is sketched in Figure 4.2.

Next, consider the region $\{q > 0, p \geq 0\}$, which we call the refracted region for reasons that will soon become apparent. We are looking for a ray with initial coordinates $(q(0), p(0))$ such that the ray passes through phase space coordinates $(q(z), p(z)) = (q, p)$ at z. The advection speed is positive, so that light can only come from the left, i.e., the source region. Any light from the source region will first encounter the interface. Hence, we need to figure out

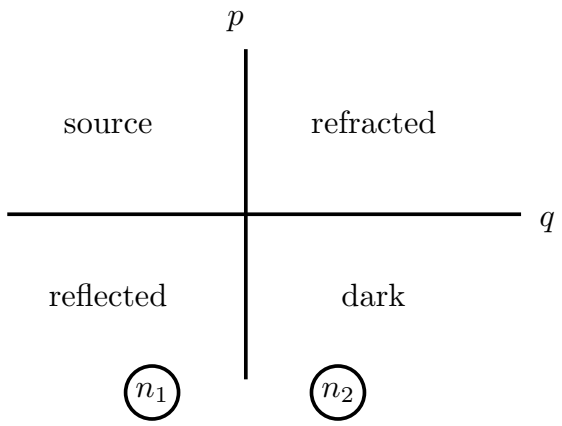


Figure 4.2: Sketch of phase space for the bucket of water problem.

what distance along the optical axis the ray travels to reach the point (q, p) from the interface, call it Δz. Hence, if we trace the ray back from its terminal position $(q(z), p(z)) = (q, p)$ to the interface, this also takes a z-interval of length Δz. From the interface to the final position, the rays are subject to simple propagation in medium n_2, so that

$$0 = q(z - \Delta z) = q - \Delta z \frac{p}{\sqrt{n_2^2 - p^2}}, \tag{4.12}$$

from which Δz can be easily found, i.e.,

$$\Delta z = \frac{q}{p}\sqrt{n_2^2 - p^2}. \tag{4.13}$$

Note that Δz depends on the terminal position in phase space. If $z - \Delta z > 0$, we can trace the ray further back to some initial coordinates in the source region, so that

$$0 = q(z - \Delta z) = q(0) + (z - \Delta z)\frac{p'}{\sqrt{n_1^2 - p'^2}}, \tag{4.14}$$

where p' is the momentum in the source region such that $\mathcal{S}(p') = p$, using Corollary 1.1. The initial position $q(0)$ can be easily isolated. The solution for ρ in this region can be found by first computing Δz from (4.12) and then using

(4.4), yielding

$$\rho(z,q,p) = \begin{cases} \rho_0\big((\Delta z - z)\frac{p'}{\sqrt{n_1^2 - p'^2}}, p'\big) & \text{if } \Delta z < z, \\ 0 & \text{otherwise.} \end{cases} \tag{4.15}$$

Finally, we consider the region $\{q \leq 0, p < 0\}$, which we call the reflected region. The advection speed is negative, while there's no light further towards the right, in the dark region. Therefore, light can only enter due to total internal reflection off the interface. Recalling Snell's function from Theorem 1.2, we find first that $\nu = -1$ so that $\psi = -p$ for $p > 0$. Total internal reflection therefore occurs whenever $n_2^2 - n_1^2 + p^2 < 0$, so that we can define the critical momentum

$$p_c = \sqrt{n_1^2 - n_2^2}, \tag{4.16}$$

while we need $p > -p_c$ when tracing backwards for TIR to occur. For a water-air transition, the critical momentum is $p_c \approx 0.9798$. Again applying the same trick, we first propagate the ray backwards from the location $(q(z), p(z)) = (q, p)$ to the interface. This leads to

$$0 = q(z - \Delta z) = q - \Delta z \frac{p}{\sqrt{n_1^2 - p^2}}, \tag{4.17}$$

whence we again can find $\Delta z = \frac{q}{p}\sqrt{n_1^2 - p^2}$. The reflection is particularly easy to compute, since $\nu = -1$, so that $p(0) = p' = -p$. Hence, if $z - \Delta z > 0$, we can again propagate backward from the interface to some initial condition in the source region. This yields

$$0 = q(z - \Delta z) = q(0) + (z - \Delta z)\frac{-p}{\sqrt{n_1^2 - p^2}}, \tag{4.18}$$

from which $q(0)$ can be found quite easily. Simplifying and using (4.4), we find

$$\rho(z,q,p) = \begin{cases} \rho_0\big(z\frac{p}{\sqrt{n_1^2 - p^2}} - q, -p\big) & \text{if } p > -p_c \text{ and } \Delta z < z, \\ 0 & \text{otherwise.} \end{cases} \tag{4.19}$$

It turns out that due to the compact support of ρ_0 in the source region, we can simply ignore the cases where $\Delta z > z$. In either case, even though the answer is wrong, the initial condition is computed to lie outside the source region, thereby

returning $\rho = 0$ anyway. The global solution can therefore be compiled as

$$\rho(z,q,p) = \begin{cases} \rho_0\left(q - z\frac{p}{\sqrt{n_1^2-p^2}}, p\right) & \text{if } q < 0, p \geq 0, \\ \rho_0\left(z\frac{p}{\sqrt{n_1^2-p^2}} - q, -p\right) & \text{if } q < 0, -p_c < p < 0, \\ \rho_0\left((\Delta z - z)\frac{p'}{\sqrt{n_1^2-p'^2}}, p'\right) & \text{if } q > 0, p \geq 0, \\ 0 & \text{otherwise,} \end{cases} \tag{4.20}$$

with p' such that $\mathcal{S}(p') = p$, p_c defined by (4.16) and Δz given by

$$\Delta z = \frac{q}{p}\sqrt{n_2^2 - p^2}. \tag{4.21}$$

4.2.3 Compound parabolic concentrator

The compound parabolic concentrator (CPC) is a pair of mirrors that have a parabolic shape. We'll discuss the two-dimensional CPC, as the three-dimensional version is usually constructed from rotating the two-dimensional CPC around the z-axis. A CPC is fixed by two parameters: the exit aperture a and the acceptance angle θ, see Figure 4.3. The spatial aperture width is $2a$, while the angular aperture width is 2θ. In two dimensions, the CPC is a perfect concentrator, which means any light falling into the entrance aperture with an angle less than the acceptance angle is directed to the exit aperture. Any ray with a larger angle is rejected by various reflections [2]. The exit aperture looks like a *Lambertian light* source, which means the brightness is constant on $[-a, a] \times [-\frac{\pi}{2}, \frac{\pi}{2}]$.

The right wall of the CPC can be constructed by tilting a parabola with respect to the z-axis over angle θ and shifting it such that its focal point is at $(-a, 0)$. Finally, the focal distance is such that the parabola goes through the point $(a, 0)$. The right wall consists of the part of the parabola with $q > 0$ and $z > 0$. The left wall is simply the right wall reflected in the z-axis, see Figure 4.3. The right wall of the CPC is given by $q = Q_r(z)$, with

$$Q_r(z) = \frac{c_1 z + b_1 + \sqrt{c_2 z + b_2}}{2\cos^2\theta}. \tag{4.22a}$$

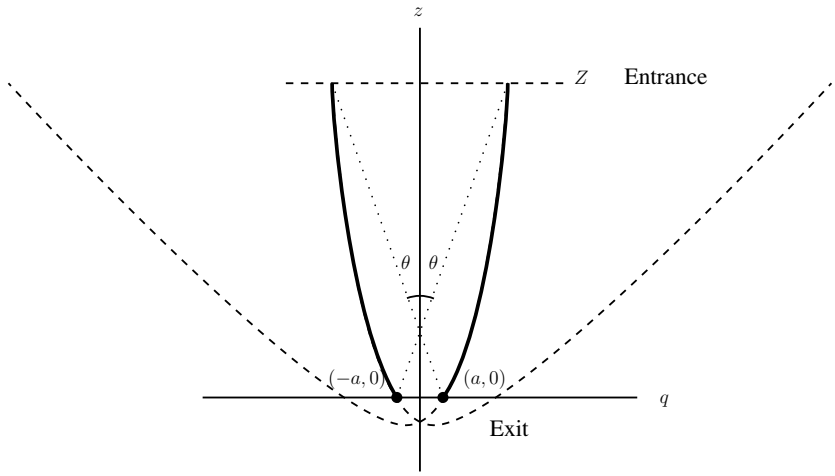


Figure 4.3: The CPC is constructed by tilting and shifting two parabolas.

where the coefficients are given by

$$c_1 = -\sin(2\theta), \tag{4.22b}$$

$$b_2 = a\big(\cos(2\theta) - 3 - 4\sin\theta\big), \tag{4.22c}$$

$$c_2 = 8a\big(2\cos\theta + \sin(2\theta)\big), \tag{4.22d}$$

$$b_2 = 8ab_1. \tag{4.22e}$$

The left wall is given by $q = -Q_r(z)$. The normal can be found straightforwardly, since Q_r is differentiable. The shape of a CPC with exit aperture a and acceptance angle θ is completely fixed. The optic has a length Z, given by

$$Z = a\frac{(1+\sin\theta)\cos\theta}{\sin^2\theta}. \tag{4.23}$$

In two dimensions the CPC is a perfect concentrator, which means the spatial concentration is as high as possible. However, this entails also maximal dilution in the angular coordinate, due to étendue conservation, see Theorem 1.1.

We can apply étendue conservation to find how a distribution that is constant on the entrance aperture is transformed when traversing the CPC. The exit aperture has width $2a$, while maximal angular dilution implies that the angular range of the exit is $[-\frac{\pi}{2}, \frac{\pi}{2}]$, meaning $[-1, 1]$ in terms of momentum. Thus, the

exit aperture has étendue $4a$. The entrance aperture has angular range $[-\theta, \theta]$, or $[-\sin\theta, \sin\theta]$ in terms of momentum, whereby the half-width of the entrance aperture is given by

$$a' = \frac{a}{\sin\theta}. \tag{4.24}$$

Thus, if the initial condition is given by the characteristic function of $[a, a] \times [-1, 1]$ at the exit aperture, i.e.,

$$\rho_0(q, p) = \begin{cases} 1, & \text{for } -a \le q \le a, -1 \le p \le 1, \\ 0, & \text{otherwise,} \end{cases} \tag{4.25}$$

it follows that the solution at $z = Z$, the entrance aperture, is given by the characteristic function of $[-a', a'] \times [-\sin\theta, \sin\theta]$, i.e.,

$$\rho(Z, q, p) = \begin{cases} 1, & \text{for } -a \le q\sin\theta \le a, -\sin\theta \le p \le \sin\theta, \\ 0, & \text{otherwise.} \end{cases} \tag{4.26}$$

4.3 Scaling arguments

So far, we've only argued with hand-waving arguments that Eulerian methods are more suitable to solve illumination problems than ray tracing. To recap: ray tracers are usually not energy conserving; information is generated at a local level, which requires a conversion to the global level; and ray tracers are wasteful of the computational accuracy of individual rays. We'd like quantify the effects of these objections by looking at scaling arguments, in particular time scaling and error scaling. At the same time, we'll also take a closer look at the scaling behaviour of Eulerian methods.

4.3.1 Ray tracing

Ray tracing with a bin-count reconstruction works by laying down a collection of E bins on d-dimensional phase space, where $d \in \{2, 4\}$, and determining the averages by a Monte Carlo process. Here, $d = 2$ for two-dimensional optics and $d = 4$ for three-dimensional optics, since we choose d as the phase space dimension. For each bin, the average is approximated by the Monte Carlo average, i.e.,

$$\frac{1}{|\Omega_i|} \int_{\Omega_i} \rho \, \mathrm{d}y \approx \frac{1}{N_i} \sum_{k=0}^{N_i} \rho_k. \tag{4.27}$$

where N_i is the number of rays incident to bin i, represented by a domain Ω_i in phase space. The initial condition of each ray is random, so that N_i is random, as well as each ρ_k. The estimate for the average on bin i converges with a standard deviation of $\mathcal{O}\big(N_i^{-1/2}\big)$ [64]. Although this is a statistical measure for the error, we'll simply use the standard deviation as an estimate of the error of the Monte Carlo method.

After obtaining the averages, the local approximation on the bin is completed by defining the numerical solution of ρ to be constant hence equal to its average over the bin. This results in a piecewise constant approximation to the brightness ρ, which is first-order accurate. Let's say there are E bins and let's assume that they're roughly the same size. This means the number of rays in each bin will be close to the average number of rays per bin, i.e., $N_i \approx \frac{N}{E}$ with N the total number of rays. The total error of the Monte Carlo ray tracing method is the sum of the two contributions and will therefore scale as

$$e_{\mathrm{RT}} = \mathcal{O}\left(\sqrt{\frac{E}{N}}\right) + \mathcal{O}\left(\frac{1}{E^{1/d}}\right). \tag{4.28}$$

This error will be minimal when the two terms scale the same way. The logic is straightforward: if the Monte Carlo error dominates, we should use more rays, while if the discretisation error dominates, we're using too many rays. There is, therefore, an optimal scaling for bins versus rays, which occurs whenever the two error contributions scale alike[3]. The result is that the number of rays should scale as

$$N = \mathcal{O}\left(E^{1+2/d}\right). \tag{4.29}$$

For two-dimensional optics $d = 2$ and $N \sim E^2$, while for three-dimensional optics $d = 4$ and $N \sim E^{3/2}$.

Computing the location of a single ray on the target usually takes constant time, meaning $\mathcal{O}(1)$, after which the correct bin has to be found. We're ignoring some pathological cases here, such as the whispering modes of a CPC, which reflect an infinite number of times. Once a ray location has been computed, a naive search will typically require linear time $\mathcal{O}(E)$, while a more advanced strategy might achieve logarithmic time. For instance, a binary structure may be imposed on the bins, allowing the right bin to be found in logarithmic time, $\mathcal{O}(\log E)$. However, it's unclear how to do such things on unstructured grids.

[3]You can also obtain this condition by setting the derivative of e_{RT} with respect to E to zero after assuming some constants for each term.

Therefore, we'll assume a simple linear search over the elements, so that each ray will take linear time to process, resulting in

$$t_{\mathrm{RT}} = \mathcal{O}\left(E^{2+2/d}\right), \tag{4.30}$$

which is simply the number of rays multiplied by the computation time per ray. For two-dimensional optics, we find $t_{\mathrm{RT}} \sim E^3$, while for three-dimensional optics we have $t_{\mathrm{RT}} \sim E^{5/2}$. This looks like it suggests that ray tracing in four dimensions is actually faster, but remember that E is the total number of elements. Choosing a product grid of M points per dimension, i.e., $E \sim M^d$, the computation time on the regular grid becomes $\mathcal{O}(M^{2d+2})$, which does increase for higher dimensions.

Second-order accurate ray tracing

Whereas our previous illustration of a ray tracing method produces a first-order accurate ray tracing scheme, it's actually possible to define a second-order ray tracer. This is due to the fact that the Monte Carlo sum (4.27) supplies averages and not point values directly. The midpoint rule, applied in reverse, tells us that the midpoints of each bin actually converges with second order. Indeed, for any smooth function $f : [a, b] \to \mathbb{R}$ with average value $\bar{f} = \frac{1}{b-a}\int_a^b f(x)\,\mathrm{d}x$, we have

$$\begin{aligned}\bar{f} - f\left(\frac{a+b}{2}\right) &= \frac{1}{b-a}\int_a^b f'\left(\frac{a+b}{2}\right)\left(x - \frac{a+b}{2}\right)\mathrm{d}x + \mathcal{O}((b-a)^2) \\ &= \mathcal{O}\left((b-a)^2\right).\end{aligned} \tag{4.31}$$

This estimate also holds for higher dimensions under some mild conditions on the shape of the domain. However, in the context of our ray tracer, this means that defining the result of the Monte Carlo process as the midpoint value of each bin, the result is second-order accurate. Linearly interpolating between nearest neighbours then results in second-order accuracy globally on phase space.

The Monte Carlo process is not changed, hence its contribution to the error is unchanged. Therefore, using this alternative reconstruction procedure, we find a different error scaling for second-order ray tracing, namely

$$e_{\mathrm{RT2}} = \mathcal{O}\left(\sqrt{\frac{E}{N}}\right) + \mathcal{O}\left(\frac{1}{E^{2/d}}\right). \tag{4.32}$$

Performing the same analysis as above and finding the optimal scaling for the number of rays results in

$$N = \mathcal{O}\left(E^{4/d+1}\right). \tag{4.33}$$

In particular, this yields $N \sim E^3$ for $d = 2$ and $N \sim E^2$ for $d = 4$. Assuming a constant computation time per ray and a linear search to find the correct element, we find therefore that

$$t_{\mathrm{RT2}} = \mathcal{O}\left(E^{4/d+2}\right). \tag{4.34}$$

Again, it may seem that ray tracing on a four-dimensional phase space is faster than on a two-dimensional phase space, but if we use a product grid of M points per dimension, we have $E \sim M^d$ and consequently $N \sim M^{2d+4}$.

Though we remark that second-order ray tracing is indeed an option, we've chosen not to use it, because the time scaling is simply too unfavourable. Indeed, throughout this work we'll focus on two-dimensional optics problems, where we'd suffer $t_{\mathrm{RT2}} \sim E^4$. Even though this is polynomial, practically speaking computation times will become unwieldy very quickly. For example, gaining another decimal point in accuracy on a two-dimensional phase space would increase the computation time by a factor of 10.000. As such, we accept a lower accuracy at the benefit of a better time scaling.

Low-dimensional ray tracing

The ray tracers illustrated above are valid methods for solving Liouville's equation, or in other words, to compute the brightness distribution. However, ray tracing is more commonly employed to compute either the luminous intensity or the illuminance. To jog the memory, intensity is the power per solid angle and illuminance is the power per unit area. Hence, compared to the brightness, computing these profiles provides lower-dimensional information. For phase space dimension $d \in \{2, 4\}$, these distributions live in a $\frac{d}{2}$-dimensional world. Consequently, the number of bins will be roughly $\sqrt{E}$ if E bins are used for the phase space distribution.

Let's denote by F the number of bins in the lower-dimensional problem. Assuming a second-order ray tracer is used, the discretisation error of the numerical solution will be $\mathcal{O}(F^{2/\tilde{d}})$, where $\tilde{d} = \frac{d}{2}$, with d the phase space dimension. Therefore, denoting together with the Monte Carlo random error, the low-dimensional

ray tracer will achieve an error of

$$e_{\mathrm{RTL}} = \mathcal{O}\left(\sqrt{\frac{F}{N}}\right) + \mathcal{O}\left(\frac{1}{F^{4/d}}\right), \tag{4.35}$$

assuming a second-order ray tracer as discussed above. The minimum error is achieved when

$$N \sim F^{8/d+1}, \tag{4.36}$$

so that $N \sim F^5$ for $d = 2$ and $N = F^3$ for $d = 4$. In terms of the number of bins F, a two-dimensional ray tracer admits a truly horrendous scaling. The time scaling is of course worse, since an additional bin search is needed. However, for two-dimensional optics problems a one-dimensional distribution is computed, which means the correct bin can be found in logarithmic time. Moreover, the correct bin can be found in constant time when the grid is uniform. In the case that $d = 4$, we assume for simplicity that a linear search is used, yielding

$$t_{\mathrm{LRT}} = \begin{cases} \mathcal{O}(F^5 \log F), & \text{for a general grid and } d = 2, \\ \mathcal{O}(F^5), & \text{for a uniform grid and } d = 2, \\ \mathcal{O}(F^4), & \text{for } d = 4. \end{cases} \tag{4.37}$$

The time scaling is pretty bad, but it's useful to compare it to first-order ray tracing on phase space, i.e., using $F \sim \sqrt{E}$, so that the time scaling gives $t_{\mathrm{LRT}} \sim E^{5/2}$ ($d = 2$, uniform grid) or $t_{\mathrm{LRT}} \sim E^2$ ($d = 4$). In both cases, it's therefore quicker by a square root factor of E. The difference will, of course, be more pronounced as the number of bins increases.

Low-dimensional ray tracers of any order can be defined by adjusting the number of rays. For instance, choosing $N \sim F^{1+2/d}$ leads to a first-order accurate method, i.e., the error scales as $\mathcal{O}(F^{-2/d})$. As expected, choosing a lower-order ray tracer improves the computation time. Redoing the maths of before leads to times scaling with $t \sim F^2 \log F$ for $d = 2$ and $t \sim F^{5/2}$ for $d = 4$.

4.3.2 Liouville solvers

Now let's look at the Eulerian approach, where we solve Liouville's equation with a discretisation method of order $\gamma \geq 1$. For a fair comparison, we take as the number of elements or grid points the number of bins, E. Therefore, by definition the discretisation error of such a method is given by

$$e_{\mathrm{LS}} = \mathcal{O}\left(\frac{1}{\left(E^{1/d}\right)^{\gamma}}\right). \tag{4.38}$$

The time scaling is the effort per z-step times the number of z-steps. With explicit Eulerian methods, we usually have to observe a CFL condition, restricting z-steps to a size $\Delta z \leq C\frac{h_{\min}}{b_{\max}}$, with $C > 0$, $b_{\max}$ the maximum velocity and $h_{\min}$ the smallest size. The constant C depends on the discretisation method, while $b_{\max}$ is a property of the problem. Therefore, both C and $b_{\max}$ do not depend on E. The smallest size, however, does depend on the grid and scales as $h_{\min} = \mathcal{O}(E^{-1/d})$, while Δz should scale as $h_{\min}$ by the CFL condition. For some fixed integration time Z, the number of z-steps, $\frac{Z}{\Delta z}$, will therefore be $\mathcal{O}(E^{1/d})$.

The effort per element or grid point may depend on the order of accuracy of the discretisation and the dimension. For instance, a γth order finite difference scheme can be constructed by a stencil of size $\gamma + 1$ for each dimension, while the central point can be used by each dimension, so that the total effort of a z-step would be proportional to $\gamma d + 1$. In general, for each element on grid point, the update requires information to be drawn from a finite number of neighbours. The exact number of neighbours in communication with an element is a constant that can depend on the order of accuracy γ. Therefore, the time cost per element will be a polynomial of the dimension d and the order γ. The dimension is either 2 or 4, so that assuming a fixed order γ leads to a z-step taking effort $\mathcal{O}(E)$.

In the presence of optical interfaces, momentum is transferred from one part of phase space to another, while the position is continuous. Therefore, a space with half the dimensionality of phase space will have to be searched for every element or grid point near the interface. Again, we might use some kind of binary structure to make the search faster, but to keep the comparison fair we'll also assume a linear search here. The number of elements that need to be searched will depend on the dimension d. In particular, an interface manifests itself a surface in position space, so that the dimension of the interface in phase space is $d - 1$. The number of elements that need to be searched will therefore scale as $\mathcal{O}(E^{\frac{d-1}{d}})$. The total time scaling is the product of the number of elements E, the effort per element per time step, $\mathcal{O}(E^{\frac{d-1}{d}})$ for interface elements and constant otherwise, and the number of time steps, i.e., $t_{\text{LS}} \sim E(C + E^{\frac{d-1}{d}})E^{\frac{1}{d}}$. Of course, the effort for interface elements will dominate the constant, so that we find

$$t_{\text{LS}} = \mathcal{O}\left(E^2\right). \tag{4.39}$$

Again a gentle reminder: E is the total number of bins, which scales as $\mathcal{O}(M^d)$ for a d-dimensional regular grid of M points per dimension. On a regular grid, therefore, Eulerian methods should scale with $\mathcal{O}(M^{2d})$.

Collapsing to lower-dimensional distributions

As we've pointed out in the beginning of this chapter, lower-dimensional information such the intensity and illuminance can be obtained from the brightness. This is done by integrating out the unwanted dimensions. For instance, to obtain the illuminance, the brightness is integrated over all momenta. It's therefore possible to obtain lower-dimensional information by means of a Liouville solver. This is especially useful for three-dimensional optics, since humanity hasn't mastered the trick of plotting distributions in four dimensions yet.

Integrating the brightness over $\frac{d}{2}$ dimensions should be considered a post-processing technique applied to the numerical solution. However, it's still useful to know how long this will take. Suppose we want to integrate over the momenta and we fix, for the moment, the position. The fixed position defines a surface in phase space where $\frac{d}{2}$ dimensions are fixed and $\frac{d}{2}$ are left free. There are, therefore, $\mathcal{O}(\sqrt{E})$ elements that will connect to the surface. For each position we have to integrate over $\mathcal{O}(\sqrt{E})$ elements, while there are of course also $\mathcal{O}(\sqrt{E})$ positions to consider. Hence, the computation will scale linearly with the number of elements. Of course, if we fix the momentum and integrate over the positions, we obtain the same scaling. Next, we need to figure out how much effort each element takes.

Integration over an element typically takes an effort that depends on the order γ and the dimension d. Using as an example again a finite difference method of order γ, we'd also need to use a quadrature rule of order γ for each grid point. If $d = 2$, we need to apply the quadrature over one direction, while if $d = 4$, we need to apply the quadrature over two dimensions. In general the effort per element will depend polynomially on the order γ and the dimension d. Hence, we find that computing either intensity or illuminance from the brightness takes linear time in the number of elements, i.e., $\mathcal{O}(E)$.

The accuracy of this approach also has order γ, so that we find an error of $\mathcal{O}\big(E^{-\frac{\gamma}{d}}\big)$. In terms of the number of elements on the lower-dimensional space, $F \sim E^2$, this provides an accuracy of $\mathcal{O}(F^{-\frac{2\gamma}{d}})$. As a method for finding both intensity and illuminance, the computation is dominated by solving Liouville's equation, which is done in quadratic time $\mathcal{O}(E^2)$. Since $E \sim F^2$, the computation time scales quartic $\mathcal{O}(F^4)$ with the number of elements on the lower-dimensional grid. As a consequence, even if the desired quantities are intensity and illuminance, it's theoretically faster, or equally fast, to first compute the numerical solution to Liouville's equation compared to second-order ray tracing.

4.3.3 Summary of scaling arguments

We summarise the scaling arguments for the lower-dimensional ray-tracers in Table 4.1. These methods should be used if one wishes to know the luminous intensity or the illuminance. The ray tracers are compared to first solving Liouville's equation and integrating over either all positions or all momenta to obtain the desired information.

Table 4.1: Overview of the scaling behaviour for lower-dimensional methods as a function of the number of elements in the low-dimensional grid F, the dimension of phase space $d \in \{2, 4\}$ and the order γ.

	error	time
first-order ray tracing	$F^{-2/d}$	$F^2 \log F$ $(d = 2)$ or $F^{5/2}$ $(d = 4)$
second-order ray tracing	$F^{-4/d}$	$F^5 \log F$ $(d = 2)$ or F^4 $(d = 2)$
Liouville + integration	$F^{-\frac{2\gamma}{d}}$	F^4

In this case, as can be read from the table, solving Liouville's equation first and then integrating to obtain the lower-dimensional information is equally fast or faster than second-order ray tracing. First-order ray tracing is faster by quite a margin, even if we choose a first-order Eulerian method. We'll mostly be comparing the Liouville solvers to the first-order ray tracer on phase space, but we'll also see one example of the lower-dimensional approach.

Table 4.2 shows the conclusion of each the scaling arguments for phase space methods at a single glance. As advertised, Eulerian methods have much better scaling behaviour. We should note for Liouville solvers, the time scaling constant will depend polynomially on the order γ and the dimension of phase space d. However, γ and d are usually fixed, so that we are mostly interested in the scaling behaviour with respect to the number of elements E.

Table 4.2: Overview of the scaling behaviour for phase-space methods as a function of the number of elements E, the dimension $d \in \{2, 4\}$ and the order γ.

	error	time
first-order ray tracing	$E^{-1/d}$	$E^{2+2/d}$
second-order ray tracing	$E^{-2/d}$	$E^{2+4/d}$
Liouville solver	$E^{-\gamma/d}$	E^2

Perhaps surprisingly, the scaling arguments suggest that any Eulerian method should perform better than first-order ray tracing. In particular, even a first-order Eulerian method, with $\gamma = 1$, should achieve the same accuracy as the ray tracer while providing an improved time scaling. Higher-order methods should give even better accuracy while still exhibiting a superior time scaling. We'll therefore start our investigation with a first-order method as a proof of concept.

Part II

Computational methods for hyperbolic PDEs

Before we discuss implementation of Eulerian schemes for Liouville's equation, we'll show a few methods in their native environment, i.e., hyperbolic PDEs. A typical example of a hyperbolic PDE is given by

$$\frac{\partial u}{\partial t} + \nabla \cdot \mathbf{f}(u) = 0,$$

for a given flux function $\mathbf{f}$, where t denotes time and $\nabla \cdot$ is the divergence operator. Liouville's equation fits in this form for smooth Hamiltonians, provided we allow $\mathbf{f}$ to depend on space and time as well. Our basic idea is to use off-the-shelf methods away from an interface. Close to an interface, we locally adjust the schemes to incorporate Snell's law via (4.3).

For evolution equations, PDEs that depend on time, one common strategy is to discretise space first, leading to a large number of coupled ODEs. This approach is known as the *method of lines* (MOL), see e.g. [70]. The large set of coupled ODEs is then solved using a general-purpose numerical integrator like Runge-Kutta methods. All methods we present here fit into this paradigm.

Chapter 5

Upwind and WENO

Everybody knows.

Leonard Cohen

Perhaps the simplest numerical methods for partial differential equations are finite difference methods. In these methods derivatives are approximated by, as the name suggests, finite differences. Usually, such methods are applied on a regular grid where all grid points are equidistant. Taylor expansions provide a way to design and analyse such methods. The collection of grid points needed to approximate the derivative at a particular grid point is called the *stencil* [70].

We showcase here two numerical schemes that operate on regular grids. The first one is the first-order upwind method, which is elementary in the sense that it's simple yet contains all the major concepts that more elaborate methods also use. The second is the more sophisticated WENO scheme, which still clearly has its roots in finite differences.

5.1 First-order upwind

To illustrate the first-order *upwind method*, we consider a one-dimensional linear advection equation

$$\frac{\partial u}{\partial t} + b\frac{\partial u}{\partial x} = 0, \tag{5.1}$$

for $t \in [0, T]$ and $x \in [0, 1]$. The velocity field b is allowed to depend on time and space. We'd like to solve this PDE numerically with first-order accuracy on

a regular grid. Before we describe the method in detail, we make a small detour to a slightly older method.

5.1.1 The CIR scheme

The roots of the upwind method can be traced back to a method proposed by Courant, Isaacson and Rees in 1952 [72]. The CIR scheme integrates backward along the characteristics that terminate in a grid point. Linear interpolation in space is then performed to update the approximate values on the grid points. Consider the quantity $u^\star$ defined by

$$u^\star(t) = u\big(t, x(t)\big), \tag{5.2}$$

for some curve $x : \mathbb{R} \to \mathbb{R}$. The time derivative of $u^\star$ is given by

$$\frac{\mathrm{d}u^\star}{\mathrm{d}t} = \frac{\partial u}{\partial t} + \frac{\mathrm{d}x}{\mathrm{d}t}\frac{\partial u}{\partial x}. \tag{5.3}$$

A special choice of the curve x turns it into a *characteristic*, namely, the choice that x satisfies

$$\frac{\mathrm{d}x}{\mathrm{d}t} = b\big(t, x(t)\big). \tag{5.4}$$

The result is that (5.3) becomes identically zero due to (5.1), whence $u^\star$ becomes a constant. Thus, the solution to the advection equation is constant along characteristics, which are curves defined by (5.4).

We now introduce a spatial discretisation as

$$x_j = (j-1)\Delta x, \tag{5.5}$$

with $\Delta x = \frac{1}{N-1}$ and $j = 1, \ldots, N$ and a time discretisation

$$t^n = n\Delta t, \tag{5.6}$$

with $\Delta t = \frac{T}{M}$, with $n = 0, \ldots, M$. The approximation to the exact solution $u(\cdot, \cdot)$ is denoted $u_j^n \approx u(t^n, x_j)$. Using the forward Euler method to integrate (5.4) backward in time from x_j, we find that

$$u(t^{n+1}, x_j) \approx u(t^n, x_j - b_j^{n+1}\Delta t), \tag{5.7}$$

where b_j^{n+1} is shorthand for $b(t^{n+1}, x_j)$. We restrict the time step Δt to be such that $x_{j-1} \leq x_j - b_j^{n+1}\Delta t \leq x_{j+1}$. The stencil therefore consists of

$\{x_{j-1}, x_j, x_{j+1}\}$. Note that the sign of b determines whether $x_j - b_j^{n+1}\Delta t$ lies to the left or right of x_j. It's in this sense that integrating backwards along characteristics leads to upwinding. Going backwards in time leads to travelling in some definite direction in space, i.e., against the direction of the velocity field b.

Clearly, we'd like to have the numerical solution satisfy (5.7) in a discrete sense. For the moment, therefore, assume that we know the exact solution on the grid points. Since we don't know the exact solution anywhere else, we must perform an interpolation to find the actual update. For instance, the interpolation at time t^n between x_{j-1} and x_j is given by

$$\tilde{u}^n(x) = u_{j-1}^n + \frac{x - x_{j-1}}{x_j - x_{j-1}} \left(u_j^n - u_{j-1}^n\right), \tag{5.8}$$

where a similar expression is used for the interpolation between x_j and x_{j+1}. Using the interpolation, we simply define the time stepping as $u_j^{n+1} = \tilde{u}^n(x_j - b_j^{n+1}\Delta t)$. The time update therefore depends on the sign of b, and is given by

$$u_j^{n+1} = \begin{cases} u_j^n - \frac{b_j^{n+1}\Delta t}{\Delta x}\left(u_j^n - u_{j-1}^n\right), & \text{if } b_j^{n+1} \geq 0, \\ u_j^n - \frac{b_j^{n+1}\Delta t}{\Delta x}\left(u_{j+1}^n - u_j^n\right), & \text{if } b_j^{n+1} < 0. \end{cases} \tag{5.9}$$

Note that this is an explicit method, as we don't have to solve any algebraic system of equations to advance in time. Furthermore, whenever b is a constant, the forward Euler method gives the exact solution for the characteristics. The error in the numerical solution in this case therefore comes completely from the linear interpolation.

5.1.2 Upwinding differences

The upwind scheme is more easily derived than the CIR scheme. It's based on finite differences, though the previous discussion is useful as it shows the underlying ideas explicitly. The main insight is to use upwind-biased differences to approximate the spatial derivative, i.e., $\frac{\partial u}{\partial x}(t^n, x_j)$ is approximated by

$$\begin{cases} \frac{u_j^n - u_{j-1}^n}{\Delta x} & \text{if } b_j^n \geq 0, \\ \frac{u_{j+1}^n - u_j^n}{\Delta x} & \text{if } b_j^n < 0. \end{cases} \tag{5.10}$$

Together with a forward Euler time integrator, this produces the scheme

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \begin{cases} -b_j^n \frac{u_j^n - u_{j-1}^n}{\Delta x} & \text{if } b_j^n \geq 0, \\ -b_j^n \frac{u_{j+1}^n - u_j^n}{\Delta x} & \text{if } b_j^n < 0. \end{cases} \tag{5.11}$$

which, after isolating u_j^{n+1}, is almost identical to the CIR scheme except for the fact that the velocity field is evaluated at the current time level t^n. This doesn't change the order of accuracy since we have $b_j^n = b_j^{n+1} + \mathcal{O}(\Delta t)$. The major insight of upwinding the differences, however, is harder to explain without a discussion of characteristics.

The global error behaviour of this upwind method is first order in space in time, thus

$$e_{\text{global}} = \mathcal{O}(\Delta x) + \mathcal{O}(\Delta t). \tag{5.12}$$

This error behaviour can be derived more rigorously, but here it suffices to note that forward Euler is first-order accurate in time while the biased difference is first-order accurate in space. The idea of an upwind solver can be generalised to stencils of any size, giving schemes of any order. The time integrator has to be chosen accordingly. The guiding principle to constructing upwind schemes is that the number of grid points upwind of x_j should be larger than the number of downwind points.

Upwind schemes, like most explicit hyperbolic discretisation methods, are subject to the Courant-Friedrichs-Lewy (CFL) condition, first discussed in [73], which roughly states that the highest velocity has to fit on the grid. To be more precise, the numerical domain of dependence for any grid point must contain the analytical domain of dependence. This relation is usually expressed in terms of the CFL number $c = \frac{b_{\max}\Delta t}{\Delta x}$, where $b_{\max}$ is the maximum velocity occurring in the system. The CFL condition then implies that

$$|c| \leq 1. \tag{5.13}$$

If the CFL condition is violated, the scheme can become unstable, meaning the numerical solution blows up.

5.2 WENO schemes

As mentioned earlier, constructing higher-order upwind methods isn't hard in principle, simply using larger stencils will get you there. However, obtaining a good numerical solution is not all that straightforward, as Godunov pointed out

in his famous theorem [74]. It states that linear schemes that preserve monotonicity of the numerical solution can be at most first-order accurate. Therefore, nonlinear interpolation schemes are needed, such as weighted essentially non-oscillatory schemes. What follows is a near verbatim version of our article on embedded WENO schemes [75].

In a seminal paper in 1987, Harten and Osher introduced the essentially non-oscillatory (ENO) reconstruction technique [76]. The basic idea of ENO is to construct several different candidate polynomial interpolations of the numerical solution and to choose the smoothest approximation to work with. The choice is facilitated by means of smoothness indicators, which become larger as the interpolation varies more rapidly.

Building further on the ENO scheme, Liu, Osher and Chan introduced the weighted essentially non-oscillatory (WENO) reconstruction technique in 1994 [77]. The WENO technique comes from the realisation that the several approximations of ENO can be combined to construct a higher-order approximation. Instead of the logical statements inherent in the ENO scheme, the WENO scheme weighs every lower-order approximation according to its smoothness indicator. Thus, in smooth regions, WENO gives a better approximation, while reducing to ENO near discontinuities.

WENO schemes are commonly used in science and engineering, with applications in fluid dynamics, astrophysics, or any other application involving convection-dominated dynamics [78, 79]. The technique is mainly applied in the context of hyperbolic and convection-dominated parabolic PDEs. However, since it's a highly advanced interpolation technique, it also has applications in fields that don't use it as part of a PDE solver, such as computer vision and image processing [80, 81].

The standard WENO scheme as it's most commonly used today was devised by Jiang and Shu [82], and is sometimes referred to as the WENO-JS scheme. Recently, several variants of the WENO scheme have appeared that improve the order of accuracy near points where the first derivative vanishes. Examples include the WENO-M [83, 84], WENO-Z [85–87] and WENO-NS [88] schemes. For a comparison of the performance of these schemes, see Zhao et al. [89]. Other efforts have focused on creating energy-stable WENO schemes such as those constructed by Yamaleev et al. [90,91], or decreasing numerical dissipation by using central discretisations such as considered by Hu et al. [92].

The most common implementations of WENO schemes use a five-point stencil, which can be subdivided into three three-point stencils. WENO schemes switch seamlessly between the third- and fifth-order reconstructions that are

possible on the five-point stencil. The idea is straightforward: when all three smoothness indicators are roughly equal, a WENO scheme switches to the fifth-order mode. When one or more smoothness indicators are large, a WENO scheme switches to the third-order mode.

In this formulation, it seems obvious that information is discarded when only one out of three smoothness indicators is large. When this happens, the two smooth approximations could still be used to obtain better accuracy. The current WENO methods don't allow for control over the numerical solution in this situation. However, one very recent scheme which does feature this type of functionality is the targeted ENO scheme of Fu et al. [93]. Their approach is completely novel and uses a combination of ideas from ENO and WENO schemes. Here, we propose a design strategy that aims to adapt existing WENO schemes such that they utilise the maximum number of grid points that form a smooth substencil. Moreover, we'll explicitly construct variants of two existing WENO schemes that exhibit this property.

Apart from the order of convergence, one can also analyse a WENO scheme in terms of its spectral properties [94]. WENO schemes switch non-linearly between linear modes of operation and as such, it's possible to investigate the spectral properties by analysing the underlying linear schemes [95]. We'll also show that our method allows for tuning of spectral properties such as dispersion and dissipation.

5.2.1 The classical WENO scheme

The WENO method is an advanced interpolation technique that aims to suppress spurious oscillations. It's commonly used as part of a high-resolution scheme for hyperbolic conservation laws, e.g.,

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} f(u) = 0, \tag{5.14}$$

where f is the flux function. We once again use a regular grid introduced in (5.5). With each point x_j, we associate a cell centred on x_j of width Δx, i.e., the interval $(x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}})$. Taking the average of the conservation law over a cell leads to

$$\frac{\mathrm{d}\bar{u}_j}{\mathrm{d}t} + \frac{f(u(t, x_{j+\frac{1}{2}})) - f(u(t, x_{j-\frac{1}{2}}))}{\Delta x} = 0, \tag{5.15}$$

where $u_{j\pm\frac{1}{2}}(t) = u(t, x_{j\pm\frac{1}{2}})$. Note that the representation (5.15) is exact and therefore conservative. In this setting, we approximate point values $u(t, x_{j\pm\frac{1}{2}})$ from their average values $\bar{u}_j(t)$.

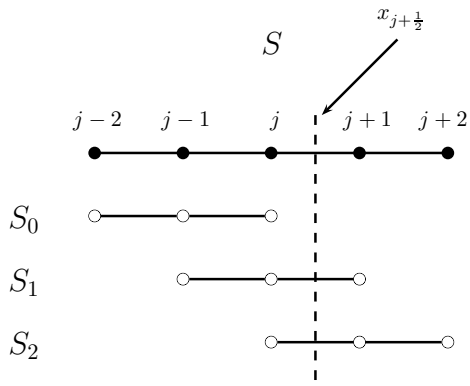


Figure 5.1: The five-point stencil S, with substencils S_0, S_1 and S_2. Note that the stencil is asymmetric around the interpolation point.

In the following, we'll suppress the explicit time dependence of u, as we interpolate in space for fixed time. The first order of business is to replace the exact flux by a numerical flux $F(u, v)$, e.g. the Lax-Friedrichs flux [96]. The numerical flux usually depends on two values, which are the left and right of u limits toward the cell edges, yielding as the semi-discrete numerical scheme

$$\frac{\mathrm{d}\bar{u}_j}{\mathrm{d}t} = -\frac{1}{\Delta x}\left(F(u^+_{j+\frac{1}{2}}(t), u^-_{j+\frac{1}{2}}(t)) - F(u^+_{j-\frac{1}{2}}(t), u^-_{j-\frac{1}{2}}(t))\right), \tag{5.16}$$

where $u^+_{j\pm\frac{1}{2}}(t)$ are the cell edge right limits and $u^-_{j\pm\frac{1}{2}}(t)$ are the cell edge left limits. This way, discontinuities at the cell edges are handles by the numerical flux. The second thing to do is to introduce some kind of interpolation scheme to approximate u on the cell edges. If we'd naively use polynomial interpolation, this would inadvertently introduce spurious oscillations. To avoid them, we need a more advanced interpolation technique. WENO is such a technique especially designed to suppress oscillations.

The classical WENO scheme, or WENO-JS, can be constructed by considering a five-point stencil around x_j, i.e., $S = \{x_{j-2}, x_{j-1}, x_j, x_{j+1}, x_{j+2}\}$. The large stencil can be divided into three smaller substencils, viz., $S_0 = \{x_{j-2}, x_{j-1}, x_j\}$, $S_1 = \{x_{j-1}, x_j, x_{j+1}\}$ and $S_2 = \{x_j, x_{j+1}, x_{j+2}\}$; see Figure 5.1.

On each of these substencils, S_k with $k = 0, 1, 2$, we can approximate $u_{j\pm\frac{1}{2}} \approx u(x_{j\pm\frac{1}{2}})$ by constructing a second-degree polynomial p_k that has cell averages

as $\bar{u}_j$, i.e.,

$$\frac{1}{\Delta x}\int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} p_k(x)\,\mathrm{d}x = \bar{u}_j. \tag{5.17}$$

Introducing the auxiliary vector $\mathbf{v} = (\bar{u}_{j-2}, \bar{u}_{j-1}, \bar{u}_j, \bar{u}_{j+1}, \bar{u}_{j+2})^T$ representing the averages on the large stencil S, the three lower-order approximations of $u_{j+\frac{1}{2}}$ can be represented as

$$\mathbf{u}_{j+\frac{1}{2}} = C\mathbf{v}, \tag{5.18}$$

with C a 3×5 matrix. It's straightforward to show that one can obtain a fifth-order approximation by taking a linear combination of the third-order approximations in (5.18). The fifth-order upwind approximation is therefore given by

$$u_{j+\frac{1}{2}}^{(UW5)} = \boldsymbol{\gamma}^T C\mathbf{v}, \tag{5.19}$$

with $\boldsymbol{\gamma}$ being a column vector. The matrix C and the linear, or optimal, weights $\boldsymbol{\gamma}$, can be represented in a tableau inspired by Butcher Tableaux [35],

$$\begin{array}{c|c} C & \boldsymbol{\gamma} \end{array}. \tag{5.20}$$

Organised this way, it contains all the coefficients involved in a WENO scheme, thus giving a concise overview of the underlying linear method. The tableau for the five-point WENO scheme looks as follows [82],

$$\begin{array}{ccccc|c} \frac{2}{6} & -\frac{7}{6} & \frac{11}{6} & & & \frac{1}{10} \\ & -\frac{1}{6} & \frac{5}{6} & \frac{2}{6} & & \frac{6}{10} \\ & & \frac{2}{6} & \frac{5}{6} & -\frac{1}{6} & \frac{3}{10} \end{array}, \tag{5.21}$$

where zero entries have been left out for clarity.

The previous discussion shows how a fifth-order approximation can be constructed from three third-order underlying approximations. However, whenever there is a discontinuity on the stencil, the fifth-order approximation incurs spurious oscillations and a third-order approximation might actually be more robust, i.e., be less oscillatory.

This idea can be realised by introducing smoothness indicators β_k, $k = 0, 1, 2$. There are several smoothness indicators available, each one exhibiting some desirable property [97, 98]. A very popular set of indicators, however, was

introduced by Jiang and Shu, see [82], they're given by

$$\beta_k := \int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} \left(p_k''(x)\right)^2 \Delta x^3 + \left(p_k'(x)\right)^2 \Delta x \,\mathrm{d}x. \tag{5.22}$$

A tedious but straightforward calculus exercise shows that

$$\beta_0 = \frac{13}{12}(\bar{u}_{j-2} - 2\bar{u}_{j-1} + \bar{u}_j)^2 + \frac{1}{4}(\bar{u}_{j-2} - 4\bar{u}_{j-1} + 3\bar{u}_j)^2, \tag{5.23a}$$

$$\beta_1 = \frac{13}{12}(\bar{u}_{j-1} - 2\bar{u}_j + \bar{u}_{j+1})^2 + \frac{1}{4}(\bar{u}_{j-1} - \bar{u}_{j+1})^2, \tag{5.23b}$$

$$\beta_2 = \frac{13}{12}(\bar{u}_j - 2\bar{u}_{j+1} + \bar{u}_{j+2})^2 + \frac{1}{4}(3\bar{u}_j - 4\bar{u}_{j+1} + \bar{u}_{j+2})^2, \tag{5.23c}$$

where one can recognise undivided finite differences. Provided that u is sufficiently smooth, a Taylor expansion reveals that $\beta_k = \mathcal{O}(\Delta x^2)$, where the coefficients of the expansion contain various derivatives of u, either squared or multiplied with higher order derivatives, i.e.,

$$\beta_0 = (u_j')^2\Delta x^2 + \left(\tfrac{13}{12}(u_j'')^2 - \tfrac{2}{3}u_j'u_j'''\right)\Delta x^4 - \left(\tfrac{13}{6}u_j''u_j''' - \tfrac{1}{2}u_j'u_j''''\right)\Delta x^5 + \mathcal{O}(\Delta x^6), \tag{5.24a}$$

$$\beta_1 = (u_j')^2\Delta x^2 + \left(\tfrac{13}{12}(u_j'')^2 + \tfrac{1}{3}u_j'u_j'''\right)\Delta x^4 + \mathcal{O}(\Delta x^6), \tag{5.24b}$$

$$\beta_2 = (u_j')^2\Delta x^2 + \left(\tfrac{13}{12}(u_j'')^2 - \tfrac{2}{3}u_j'u_j'''\right)\Delta x^4 + \left(\tfrac{13}{6}u_j''u_j''' + \tfrac{1}{2}u_j'u_j''''\right)\Delta x^5 + \mathcal{O}(\Delta x^6), \tag{5.24c}$$

where u_j' is shorthand for $\partial_x u(x_j)$, etc. Whereas an ENO scheme uses a logical statement to select the interpolation with the lowest smoothness indicator, a WENO scheme proposes to use a convex combination of the third-order interpolations, much like (5.19). To this end, the nonlinear weights ω_k are introduced, which are functions of the smoothness indicators. Collecting the nonlinear weights into a column vector $\boldsymbol{\omega}$, a WENO scheme uses a linear combination of the form

$$u_{j+\frac{1}{2}}^{(\mathrm{WENO})} = \boldsymbol{\omega}^T C \mathbf{v}. \tag{5.25}$$

Consistency requires that the nonlinear weights ω_k $(k = 0, 1, 2)$ sum to unity. Hence, to construct nonlinear weights that satisfy the requirements discussed earlier, we first compute the unnormalised nonlinear JS weights, indicated with a superscript JS, as

$$\tilde{\omega}_k^{\mathrm{JS}} = \frac{\gamma_k}{(\beta_k + \varepsilon)^p}, \tag{5.26}$$

with $\varepsilon > 0$ a small number to avoid division by zero and $p > 0$. Typical values are $\varepsilon = 10^{-6}$ and $p = 2$. In any WENO scheme the unnormalised weights are subsequently normalised to obtain the nonlinear weights

$$\omega_k = \frac{\tilde{\omega}_k}{\sum_{l=0}^{2} \tilde{\omega}_l}. \tag{5.27}$$

The WENO-JS scheme gives fifth-order accuracy whenever u is smooth, i.e., $u'_j = \mathcal{O}(1)$ and consequently $\beta_k = \mathcal{O}(\Delta x^2)$, or if ε is sufficiently large compared to the second-order terms in the expansions (5.24), otherwise only third-order is attained [83]. At the same time, it gives third-order accuracy whenever a substencil contains a discontinuity, since then the corresponding smoothness indicator becomes large. By using only one of the smooth substencils instead of all three, oscillations are suppressed.

A modern incarnation of the WENO scheme is given by the WENO-Z scheme of Borges et al. [85], who showed that a sufficient condition for fifth-order accuracy is

$$\omega_k = \gamma_k + \mathcal{O}(\Delta x^3). \tag{5.28}$$

The WENO-JS scheme attains $\omega_k = \gamma_k + \mathcal{O}(\Delta x^2)$, however it does satisfy a more complicated condition ensuring fifth-order accuracy. Unfortunately, near critical points, i.e., where a (higher) derivative vanishes, the WENO-JS scheme only provides third-order accuracy as pointed out by Henrick et al. [83]. WENO-Z was designed to satisfy (5.28) and thereby restore optimal convergence near critical points. The unnormalised weights, indicated with a superscript Z, are given by

$$\tilde{\omega}_k^Z = \gamma_k \left(1 + \left(\frac{\tau}{\beta_k + \varepsilon}\right)^p\right), \tag{5.29}$$

where $\tau = |\beta_2 - \beta_0|$ is called the global smoothness indicator. Using (5.24), one can show that $\tau = \mathcal{O}(\Delta x^5)$ and so WENO-Z satisfies the sufficient condition (5.28) for any $p \geq 1$.

WENO schemes fit perfectly into the method of lines paradigm, where one leaves time continuous while discretising space. The result of the scheme can be represented by the following system of ODEs,

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = L(\mathbf{u}), \tag{5.30}$$

where L represents the action of the WENO scheme. After the spatial discretisation, one discretises time by setting time levels $t^n = n\Delta t$, $n = 0, 1, \ldots$. The time

integrators of choice are the strong stability preserving Runge-Kutta methods (SSPRK) [99, 100]. These are explicit Runge-Kutta methods that have a high order of accuracy and do not incur spurious oscillations due to time integration. The golden standard for five-point WENO schemes is the SSPRK(3,3) method, a three-stage third-order SSP time integrator. One time step of this method is given by

$$\mathbf{u}^{(1)} = \mathbf{u}^n + \Delta t L(\mathbf{u}^n), \tag{5.31a}$$

$$\mathbf{u}^{(2)} = \tfrac{3}{4}\mathbf{u}^n + \tfrac{1}{4}\mathbf{u}^{(1)} + \tfrac{1}{4}\Delta t L(\mathbf{u}^{(1)}), \tag{5.31b}$$

$$\mathbf{u}^{n+1} = \tfrac{1}{3}\mathbf{u}^n + \tfrac{2}{3}\mathbf{u}^{(2)} + \tfrac{2}{3}\Delta t L(\mathbf{u}^{(2)}). \tag{5.31c}$$

Wang and Rong have shown that this method is also linearly stable when combined with a five-point WENO scheme [101].

5.2.2 Embedded WENO

We now pose the question of what happens when the solution on two adjacent substencils is smooth with no critical points and the third one contains a discontinuity. Specifically, either the solution is smooth on S_0 and S_1 and not smooth on S_2, or the solution is smooth on S_1 and S_2 and not on S_0. The answer is that the WENO-JS scheme provides third-order accuracy while suppressing oscillations. However, the scheme generates a linear combination of the two smooth substencils that is forced by the fifth-order mode, i.e., the user cannot choose the resulting weights. Being able to choose the resulting weights results in direct control over the truncation error and the numerical dissipation and dispersion.

As a shorthand whenever the solution is smooth on a substencil S_k, we call the substencil smooth. In the following, we ignore critical points for the moment. Let's examine the normalised JS weights, indicated with the superscript JS, from the definition (5.26) - (5.27) we find that

$$\frac{\omega_k^{\mathrm{JS}}}{\omega_l^{\mathrm{JS}}} = \frac{\tilde{\omega}_k^{\mathrm{JS}}}{\tilde{\omega}_l^{\mathrm{JS}}} = \frac{\gamma_k}{\gamma_l}\left(\frac{\beta_l}{\beta_k}\right)^p, \tag{5.32}$$

where we've assumed ε is negligible compared to the smoothness indicators. Thus, the proportions of the nonlinear weights only depend on the local smoothness of S_k and S_l, as one has for $k \neq l$,

$$\frac{\beta_l}{\beta_k} = \begin{cases} 1 + \mathcal{O}(\Delta x^3) & \text{if } k = 0, l = 2 \text{ or } k = 2, l = 0, \\ 1 + \mathcal{O}(\Delta x^2) & \text{otherwise,} \end{cases} \tag{5.33}$$

which follows from the Taylor expansions of the smoothness indicators (5.24). Therefore, the ratios of the nonlinear weights satisfy

$$\frac{\omega_k^{\mathrm{JS}}}{\omega_l^{\mathrm{JS}}} = \frac{\gamma_k}{\gamma_l}\left(1 + \mathcal{O}\left(\Delta x^s\right)\right), \tag{5.34}$$

with $s \geq 2$, provided ε is much smaller than β_k and β_l.

A similar computation for the WENO-Z weights, indicated with a superscript Z, shows that this relation also holds, i.e.,

$$\frac{\omega_k^{Z}}{\omega_l^{Z}} = \frac{\gamma_k\left(1 + (\frac{\tau}{\beta_k})^p\right)}{\gamma_l\left(1 + (\frac{\tau}{\beta_l})^p\right)} = \frac{\gamma_k\left(\beta_l^p + \tau^p(\frac{\beta_l}{\beta_k})^p\right)}{\gamma_l\left(\beta_l^p + \tau^p\right)}. \tag{5.35}$$

Again using (5.33), we find that now independent of the value of τ,

$$\frac{\omega_k^{Z}}{\omega_l^{Z}} = \frac{\gamma_k}{\gamma_l}\left(1 + \mathcal{O}\left(\Delta x^s\right)\right), \tag{5.36}$$

again with $s \geq 2$. Like with WENO-JS, this relation only depends on the local smoothness of S_k and S_l. Note that when the entire stencil S is smooth, the lower bound is increased to $s \geq 3p$. This is due to the fact that $\tau = \mathcal{O}(\Delta x^5)$ in this situation and hence $\left(\frac{\tau}{\beta_k}\right)^p = \mathcal{O}(\Delta x^{3p})$.

Now consider S_0 and S_1 being smooth, but S_2 contains a discontinuity. In this case, both JS and Z schemes will result in $\frac{\omega_0}{\omega_1} = \frac{\gamma_0}{\gamma_1} + \mathcal{O}(\Delta x^2)$. This leads to $\omega_0 \approx \frac{1}{7}$ and $\omega_1 \approx \frac{6}{7}$ for both JS and Z schemes, from which the WENO approximation becomes

$$u_{j+\frac{1}{2}}^{(\mathrm{WENO})} - u(x_{j+\frac{1}{2}}) = \tfrac{1}{28}u_j'''\Delta x^3 + \mathcal{O}(\Delta x^4), \tag{5.37}$$

by a Taylor expansion of the third-order approximations. However, this may not be the optimal choice of weights, as it leads to a third-order linear combination while, for instance, a fourth-order combination is possible if $\omega_0 \approx \frac{1}{4}$ and $\omega_1 \approx \frac{3}{4}$ for this situation.

Relations (5.34) and (5.36) therefore show the flaw that we address: when the large stencil S is not smooth, WENO-JS and WENO-Z immediately revert to lowest-order modes, even when there are multiple adjacent smooth substencils. In such cases, being able to choose the resulting linear combination has some advantages. If anything, it allows for more control over the numerical solution.

More control over the numerical solution in this case means reducing dissipation and increasing resolution.

We propose a technique that allows for a choice of the resulting nonlinear weights in the situation when either $\beta_0 = \mathcal{O}(1)$ or $\beta_2 = \mathcal{O}(1)$ and the other substencils are smooth. Consequently, this allows for direct control over the truncation error of the numerical solution in these situations. We call this new type of scheme an embedded WENO scheme. Similarly to conventional WENO schemes, fifth-order accuracy is demanded whenever the numerical solution is smooth on the entire stencil S. Moreover, it should reduce to an ENO scheme when two out of three substencils contain a discontinuity.

Let's set the question of how to achieve this aside for the moment and first introduce some terminology. The overall third-to-fifth-order accurate scheme is called the outer scheme. The resulting scheme when there are only two adjacent smooth substencils is called the inner scheme, see Figure 5.2. For instance, an obvious choice is a fourth-order inner scheme in combination with WENO-JS as the outer scheme.

Examining Figure 5.2 more closely, it becomes clear that if S_2 contains the discontinuity and S_0 and S_1 are smooth, then the discontinuity must lie in the interval (x_{j+1}, x_{j+2}). Consequently, there are four grid points on which there is a smooth solution to interpolate. From the two remaining substencils, a four-point stencil can be constructed where the inner scheme is defined. When S_2 contains the discontinuity, the available four-point stencil is $S_{0,1} := S_0 \cup S_1$. When S_0 contains the discontinuity, the four-point stencil is $S_{1,2} := S_1 \cup S_2$ to use for the inner scheme.

Even though a higher formal order of convergence may be obtained, Banks et al. [102] have pointed out that one often obtains sublinear convergence near linearly degenerate discontinuities, such as discontinuities in linear hyperbolic equations or the contact waves of the Euler equations. They estimate that the convergence rate becomes $\frac{m}{m+1}$ for a scheme with formal convergence rate m. In our case, this suggests the convergence rate is increased from $\frac{3}{4}$ to $\frac{4}{5}$. Thus, the benefits might be less great as a naive estimate would suggest. However, aside from the increased convergence rate, embedded WENO also allows more control over spectral properties.

With the terminology in place we can turn to the basic question: how to embed one WENO scheme into another. Thus, we'd like the nonlinear weights to converge to the inner scheme whenever there are two adjacent smooth substencils and the third one is not smooth. Otherwise, they should remain approximately equal to the nonlinear weights of the outer scheme. This suggests that

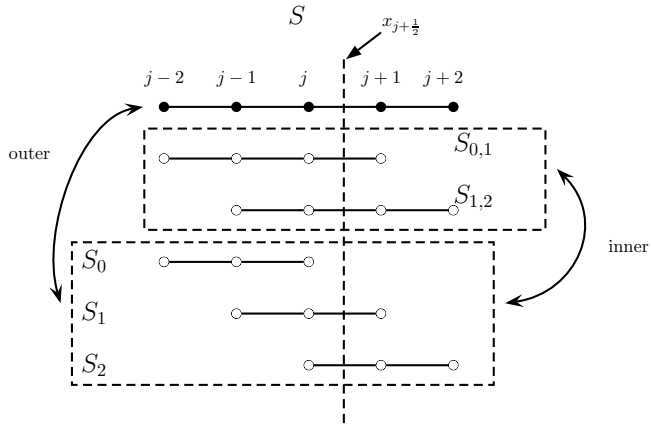


Figure 5.2: The five-point stencil S, with substencils S_0, S_1 and S_2, and inner scheme stencils $S_{0,1}$ and $S_{1,2}$.

we multiply the unnormalised weights $\tilde{\omega}_k$ of the outer scheme by a correction that is ordinarily close to unity, but activates when either β_0 or β_2 becomes $\mathcal{O}(1)$. The correction is constructed such that it adjusts the proportions found in (5.34) and (5.36).

Suppose the inner scheme is given by the linear weights $\alpha_0^{(2)}$, $\alpha_1^{(2)}$, $\alpha_1^{(0)}$ and $\alpha_2^{(0)}$. We write the stencils containing a discontinuity in parenthesis in the superscript and the substencil index in the subscript. The desired convex combination then becomes

$$u_{j+\frac{1}{2}}^{(0,1)} := \alpha_0^{(2)} u_{j+\frac{1}{2}}^{(0)} + \alpha_1^{(2)} u_{j+\frac{1}{2}}^{(1)}, \tag{5.38a}$$

$$u_{j+\frac{1}{2}}^{(1,2)} := \alpha_1^{(0)} u_{j+\frac{1}{2}}^{(1)} + \alpha_2^{(0)} u_{j+\frac{1}{2}}^{(2)}. \tag{5.38b}$$

We consider two possible choices for the linear weights of the inner scheme, see Table 5.1. The first is the fourth-order linear combination which is possible on the four-point stencil. The second choice consists of placing the superfluous weight onto the middle substencil, i.e., using the approximation $u_{j+\frac{1}{2}}^{(k)} \approx u_{j+\frac{1}{2}}^{(1)}$ for $k = 0, 2$. The fourth-order choice is motivated from an order-of-convergence perspective, while the third-order choice comes from a spectral point of view.

The nonlinear weights must at all times sum to unity to ensure consistency. Therefore, any correction we introduce must be incorporated into the unnor-

Table 5.1: Possible choices for the inner scheme.

	4th	3rd
$\alpha_0^{(2)}$	$\frac{1}{4}$	$\frac{1}{10}$
$\alpha_1^{(2)}$	$\frac{3}{4}$	$\frac{9}{10}$
$\alpha_1^{(0)}$	$\frac{1}{2}$	$\frac{7}{10}$
$\alpha_2^{(0)}$	$\frac{1}{2}$	$\frac{3}{10}$

malised nonlinear weights and still work after normalisation.

Furthermore, what's happening in substencil S_2 must influence both substencils S_0 and S_1 and *mutatis mutandis* substencil S_0 must influence both S_1 and S_2. It follows that the corrections must be functions of multiple smoothness indicators and thus enforce that the resulting WENO scheme is nonlocal. As a final note, we've seen from (5.34) and (5.36) that the nonlinear weights are simply a redistribution of the linear weights. We therefore have to influence the proportions of the linear weights, hence the linear weights should be multiplied with some relative proportions c_2 and c_0. The role of the relative proportions will become clear later, they're defined as

$$\alpha_0^{(2)} : \alpha_1^{(2)} = c_2\gamma_0 : \gamma_1, \tag{5.39a}$$

$$\alpha_1^{(0)} : \alpha_2^{(0)} = \gamma_1 : c_0\gamma_2. \tag{5.39b}$$

The naming convention is again to label the relative proportions with the index of the substencil that is not smooth. We can thus compute the relative proportions using the inner weights suggested in Table 5.1, see Table 5.2.

Table 5.2: Relative proportions for the 4th order and 3rd order inner schemes. The 3rd order inner scheme places the superfluous weight on the middle stencil.

	4th	3rd
c_2	2	$\frac{2}{3}$
c_0	2	$\frac{6}{7}$

We'll now briefly summarise the conditions that should be satisfied by an embedding correction.

1. *(Implementation)* The unnormalised nonlinear weights must be multiplied with a correction.

2. *(Nonlocality)* The corrections cannot be functions of only the local smoothness indicators.

3. *(Consistency)* Wherever the solution is smooth on the full stencil, the embedded scheme must reproduce the original scheme.

4. *(Embedding)* When there is a discontinuity present, the scheme must produce the inner weights on the smooth substencils.

General framework

Here, we construct a general framework for embedded WENO schemes. The implementation and consistency conditions suggest that our correction is ordinarily close to a constant, while according to the nonlocality condition it may not be a function of a single smoothness indicator. Therefore, we propose using a general starting point given by

$$\tilde{\omega}_k^{(\mathrm{E})} = \tilde{\omega}_k^{(\mathrm{O})} \left(a_{kk} + \sum_{l \neq k} \frac{a_{kl}\beta_l}{\beta_k + \varepsilon} \right), \tag{5.40}$$

where the outer scheme is denoted with superscript (O) and the embedded scheme with (E). This is probably the simplest possible nonlocal correction: a linear combination of ratios. Here, a_{kl} (k and l in the range $0, 1, 2$) are a collection of undetermined coefficients and ε is small constant to avoid division by zero. We'll refer to (5.40) as the general form of an embedded WENO scheme and the term in parenthesis as the general form of a correction.

The consistency condition will give us a set of equations that has to be satisfied by the coefficients a_{kl}. It tells us that when the solution is smooth all corrections must be close to 1. Let's assume that the outer scheme satisfies, whenever the solution is smooth with no critical points, $\omega_i^{(\mathrm{O})} = \gamma_i + \mathcal{O}(\Delta x^q)$, then the corrections must satisfy

$$a_{kk} + \sum_{l \neq k} \frac{a_{kl}\beta_l}{\beta_k + \varepsilon} = 1 + \mathcal{O}(\Delta x^q), \tag{5.41}$$

which must hold for all $k = 0, 1, 2$. Using (5.33), and assuming $\beta_k = \mathcal{O}(\Delta x^2)$ for all $k = 0, \ldots, r-1$, where r is the number of substencils, this yields

$$\sum_{l=0}^{r-1} a_{kl} = 1, \quad k = 0, 1, \ldots, r-1. \tag{5.42}$$

If $q = 2$, this is sufficient to satisfy the consistency condition. If $q > 2$, for instance for WENO-Z, the coefficients a_{kl} must also provide linear combinations of smoothness indicators that cancel out the lower order terms in the Taylor expansions (5.24).

To achieve this, the general form (5.40) is adjusted. In this case, at least the first term in the error expansion of (5.41) must vanish. The constant term in the correction must still equal 1, which suggests we adjust the general form to read

$$\tilde{\omega}_k^{(\mathrm{E})} = \tilde{\omega}_k^{(\mathrm{O})} \left(1 + \left(\frac{\left| \sum_{l=0}^{r-1} a_{kl} \beta_l \right|}{\beta_k + \varepsilon} \right)^p \right), \tag{5.43}$$

where r is again the number of substencils, $p \geq 1$ and ε is a small constant to avoid division by 0. We'll refer to (5.43) as the second general form. Setting $\tilde{\omega}_k^{(\mathrm{O})} = \gamma_k$, it becomes clear this form can be considered as a generalisation of the WENO-Z weights (5.29). Here, at least the lowest-order term from the smoothness indicators (5.24) must vanish, leading to

$$\sum_{l=0}^{r-1} a_{kl} = 0, \quad k = 0, 1, \ldots, r-1. \tag{5.44}$$

Regardless of which general form is chosen, (5.40) or (5.43), further equations for the coefficients a_{kl} are obtained by the embedding condition. These can be derived by examining the possible positions of a discontinuity and setting the resulting weights equal to the inner weights.

The conditions (5.42) and (5.44) hold for any number of substencils r. In deriving the embedding equations, we'll also take a more general view.

Theorem 5.1. (Embedding equations) *Let $\tilde{\omega}_k$ be the unnormalised nonlinear weights of a WENO scheme that has r substencils and satisfies $\frac{\tilde{\omega}_k}{\tilde{\omega}_l} \to \frac{\gamma_k}{\gamma_l}$ as $\Delta x \to 0$ whenever S_k and S_l are smooth. Let the unnormalised embedded WENO weights be given by*

$$\tilde{\omega}_k^{(\mathrm{E})} = \tilde{\omega}_k g_k, \quad g_k = a_{kk} + \sum_{l \neq k} \frac{a_{kl} \beta_l}{\beta_k + \varepsilon}, \quad k = 0, 1, 2, \ldots, r-1, \tag{5.45}$$

where g_k is the correction factor from the first general form (5.40)*. Let K be the set of indices such that $\beta_n = \mathcal{O}(1)$ for $n \in K$, i.e. S_n is not smooth, and let $\beta_k \downarrow 0$ for $k \notin K$ as $\Delta x \to 0$. Then, the embedding equations are given by*

$$\frac{\gamma_k}{\alpha_k^{(K)}} \sum_{m \in K} a_{km} = \frac{\gamma_l}{\alpha_l^{(K)}} \sum_{n \in K} a_{ln}, \tag{5.46}$$

with $k, l \notin K$, $n, m \in K$. Here $\alpha_k^{(K)}$ are the desired inner weights and γ_k the linear weights.

Proof. 1. Fix some set K and assume that ε is so small it may be safely ignored. Let S_k and S_l be smooth, then the ratio of their two embedded weights is given by

$$\frac{\omega_k^{(\mathrm{E})}}{\omega_l^{(\mathrm{E})}} = \frac{\tilde{\omega}_k^{(\mathrm{E})}}{\tilde{\omega}_l^{(\mathrm{E})}} = \frac{\tilde{\omega}_k}{\tilde{\omega}_l} \frac{g_k}{g_l}. \tag{$*$}$$

2. For S_k not smooth, the correction becomes

$$g_k = \sum_{n \notin K} a_{kn} + \sum_{n \in K} a_{kn} \frac{\beta_n}{\beta_k} + \mathcal{O}(\Delta x^s),$$

where $s \geq 2$ from (5.33). Next, we use that $\beta_n = \mathcal{O}(1)$ if S_n is not smooth, so that we obtain

$$g_k = \frac{C}{\Delta x^2} \sum_{n \in K} a_{kn} + \mathcal{O}(1), \tag{$\star$}$$

where $C > 0$ is some constant.
3. Next, ($\star$) is substituted into ($*$) leading to

$$\frac{\tilde{\omega}_k^{(\mathrm{E})}}{\tilde{\omega}_l^{(\mathrm{E})}} = \frac{\tilde{\omega}_k}{\tilde{\omega}_l} \frac{\sum_{n \in K} a_{kn} + \mathcal{O}(\Delta x^2)}{\sum_{m \in K} a_{lm} + \mathcal{O}(\Delta x^2)}.$$

We let $\Delta x \downarrow 0$ so that $\frac{\tilde{\omega}_k}{\tilde{\omega}_l} \to \frac{\gamma_k}{\gamma_l}$, therefore

$$\frac{\omega_k^{(\mathrm{E})}}{\omega_l^{(\mathrm{E})}} = \frac{\gamma_k}{\gamma_l} \frac{\sum_{n \in K} a_{kn}}{\sum_{m \in K} a_{lm}}.$$

This ratio has to be equal to the ratio of inner weights, so that

$$\frac{\alpha_k^{(\mathrm{K})}}{\alpha_l^{(\mathrm{K})}} = \frac{\gamma_k}{\gamma_l} \frac{\sum_{n \in K} a_{kn}}{\sum_{m \in K} a_{lm}},$$

which can be simplified to (5.46). □

Remark. *The assumption that the outer weights should satisfy* $\frac{\tilde{\omega}_k}{\tilde{\omega}_l} \to \frac{\gamma_k}{\gamma_l}$ *as* $\Delta x \to 0$ *includes the choices of JS weights, Z weights and simply using the linear weights* $\tilde{\omega}_k = \gamma_k$.

Theorem 5.2. *Under the same assumptions as Theorem 5.1 and using the second general form* (5.43)*, the embedding equations are given by*

$$\left(\frac{\gamma_k}{\alpha_k^{(K)}}\right)^{\frac{1}{p}} \sum_{m\in K} a_{km} = \pm \left(\frac{\gamma_l}{\alpha_l^{(K)}}\right)^{\frac{1}{p}} \sum_{n\in K} a_{ln}, \tag{5.47}$$

Proof. 1. Repeating the steps of the previous proof with $k \notin K$, the correction for the second form now equals

$$g_k = 1 + \left| \frac{C}{\Delta x^2} \sum_{n\in K} a_{kn} + \sum_{n\notin K} a_{kn} \left(1 + \mathcal{O}(\Delta x^2)\right) \right|^p,$$

so that

$$\frac{g_k}{g_l} = \frac{|\sum_{n\in K} a_{kn} + \mathcal{O}(\Delta x^2)|^p}{|\sum_{m\in K} a_{lm} + \mathcal{O}(\Delta x^2)|^p}.$$

2. Passing to the limit $\Delta x \downarrow 0$, we find

$$\frac{\omega_k^{(\mathrm{E})}}{\omega_l^{(\mathrm{E})}} = \frac{\gamma_k}{\gamma_l} \left| \frac{\sum_{n\in K} a_{kn}}{\sum_{m\in K} a_{lm}} \right|^p.$$

Setting the left-hand side equal to $\frac{\alpha_k^{(\mathrm{K})}}{\alpha_l^{(\mathrm{K})}}$, this yields

$$\left(\frac{\gamma_k}{\alpha_k^{(\mathrm{K})}}\right)^{\frac{1}{p}} \left| \sum_{n\in K} a_{kn} \right| = \left(\frac{\gamma_l}{\alpha_l^{(\mathrm{K})}}\right)^{\frac{1}{p}} \left| \sum_{m\in K} a_{lm} \right|.$$

Thus, if the coefficients satisfy (5.47), the embedded weights will converge to the inner weights. □

Remark. *There is a freedom in the choice of sign of the coefficients for the second form* (5.43)*. Interpreting the coefficients as elements of a matrix A, any row may be multiplied with* -1 *with impunity. From here on out, we use the positive sign in* (5.47)*.*

The embedding equations are a set of linear equations for the coefficients a_{kl}, since the inner weights are given or rather chosen by the user. The embedding equations relate the weights of the inner scheme to the linear weights. Together with the equations coming from the consistency condition, this will provide a number of linear equations for the coefficients a_{kl}. For five-point WENO schemes, we find that K can be either $\{0\}$ or $\{2\}$, the other cases being already included in the WENO weights. In each case for K there are only two remaining smooth substencils. We thus end up with two equations

$$\left(\frac{\gamma_0}{\alpha_0^{(2)}}\right)^{\frac{1}{p}} a_{02} = \left(\frac{\gamma_1}{\alpha_1^{(2)}}\right)^{\frac{1}{p}} a_{12}, \tag{5.48a}$$

$$\left(\frac{\gamma_2}{\alpha_2^{(0)}}\right)^{\frac{1}{p}} a_{20} = \left(\frac{\gamma_1}{\alpha_1^{(0)}}\right)^{\frac{1}{p}} a_{10}, \tag{5.48b}$$

where the $p = 1$ equations also apply to the first form (5.40). These may be simplified using our earlier definition of the relative proportions c_0 and c_2 (5.39), i.e.,

$$\frac{a_{02}}{a_{12}} = (c_2)^{\frac{1}{p}}, \tag{5.49a}$$

$$\frac{a_{20}}{a_{10}} = (c_0)^{\frac{1}{p}}, \tag{5.49b}$$

where once again, the $p = 1$ equations apply to both forms provided $c_2 > 0$ and $c_0 > 0$, for the second form one can use $p > 1$.

5.2.3 Implementation

Embedded WENO-JS

We'll now show how to construct embedded WENO schemes using the WENO-JS scheme as an outer scheme. We'll assume the inner weights $\alpha_0^{(2)}$, $\alpha_1^{(2)}$, $\alpha_1^{(0)}$ and $\alpha_2^{(0)}$ are given, e.g., chosen from Table 5.1. From the inner weights, we can find their relative proportions as measured against the outer weights by (5.39), see Table 5.2. We'll use the general form (5.40) as a template. Furthermore, we have that $q = 2$, so that (5.42) provides three equations that are sufficient to ensure that the scheme is unaltered when the solution is smooth. The two embedding equations for a five-point WENO scheme are given by (5.49). Hence,

we have five equations for nine coefficients that can be solved to yield a four-parameter family of embedded schemes, given by

$$a_{00} = 1 - a_{01} - a_{02}, \tag{5.50a}$$

$$a_{11} = 1 - \frac{a_{20}}{c_0} - \frac{a_{02}}{c_2}, \tag{5.50b}$$

$$a_{22} = 1 - a_{20} - a_{21}, \tag{5.50c}$$

$$a_{12} = \frac{a_{02}}{c_2}, \tag{5.50d}$$

$$a_{10} = \frac{a_{20}}{c_0}, \tag{5.50e}$$

where a_{01}, a_{02}, a_{20} and a_{21} can be chosen freely. We've experimented with a number of possible choices, all seemed to provide improvements over the WENO-JS scheme. However, different choices resulted in schemes with different behaviour, much like choosing a different flux limiter in a total variation diminishing (TVD) scheme.

We'll continue with the embedded scheme that appears to have the best all-round performance, it can be constructed using the choices $a_{01} = a_{21} = 0$, $a_{20} = \frac{c_0}{3}$ and $a_{02} = \frac{c_2}{3}$. For this particular choice of embedded WENO, we may even choose the linear weights as the outer scheme, such that we obtain

$$\tilde{\omega}_0 = \tfrac{1}{3}\gamma_0 \left(3 - c_2 + c_2 \frac{\beta_2}{\beta_0 + \varepsilon}\right), \tag{5.51a}$$

$$\tilde{\omega}_1 = \tfrac{1}{3}\gamma_1 \left(1 + \frac{\beta_2}{\beta_1 + \varepsilon} + \frac{\beta_0}{\beta_1 + \varepsilon}\right), \tag{5.51b}$$

$$\tilde{\omega}_2 = \tfrac{1}{3}\gamma_2 \left(3 - c_0 + c_0 \frac{\beta_0}{\beta_2 + \varepsilon}\right). \tag{5.51c}$$

This scheme yields a convex combination of the underlying third-order approximations when all weights are positive, thus we must have $c_0 < 3$ and $c_2 < 3$, which includes the choices presented in Table 5.2.

To show that we may use this choice, we apply a Taylor expansion to the normalised weights, under the assumption of smooth solutions without critical points. A lengthy computation or symbolic calculation will show that the weights from (5.51) satisfy

$$\omega_k - \gamma_k = \tfrac{1}{3}\left(\omega_k^{\mathrm{JS}} - \gamma_k\right) + \mathcal{O}(\Delta x^3), \tag{5.52}$$

regardless of the values of c_0 and c_2. Therefore, the embedded WENO scheme (5.51) also provides fifth-order convergence. Under the assumption that only

one substencil is smooth, the weights (5.51) also provide the proper behaviour. Indeed, fix k and set $\beta_k = \mathcal{O}(\Delta x^2)$ and $\beta_l = \mathcal{O}(1)$ with $l \neq k$, then we find that $\tilde{\omega}_k = \mathcal{O}(\frac{1}{\Delta x^2})$ and $\tilde{\omega}_l = \mathcal{O}(1)$. Therefore, with only one smooth substencil, we find $\omega_k = 1 + \mathcal{O}(\Delta x^2)$ and $\omega_l = \mathcal{O}(\Delta x^2)$.

We conclude that the embedded WENO scheme given by (5.51) is equivalent to the standard WENO-JS scheme for smooth solutions without critical points or having only a single smooth substencil. When there are two adjacent smooth substencils and the third one is not smooth, we obtain the inner scheme.

To verify that the embedded schemes have the same order of convergence for smooth functions, a short test is performed. As was shown by Borges et al., the conservative difference in (5.15) can be interpreted as a differentiation operator, call it D, if applied to the original function instead of its averages [85]. This provides an easy to implement convergence test for WENO schemes. WENO-JS only features optimal convergence for smooth functions without critical points, therefore the first test consists of applying WENO differentiation to a test function given by

$$u_1(x) = \tanh(10x). \tag{5.53}$$

The boundary conditions are supplied exactly using fictitious grid points. For functions featuring first-order critical points, WENO-JS provides fourth-order accuracy. Therefore, the second test function is given by

$$u_2(x) = \sin\left(\pi x - \frac{\sin(\pi x)}{\pi}\right). \tag{5.54}$$

In both tests, the error is computed using the scaled absolute sum

$$e = \sum_{j=1}^{N} \big|Du_j - u'(x)\big|\Delta x, \tag{5.55}$$

where D is the WENO differentiation operator. The parameter ε is set to 10^{-40} and $c_2 = c_0 = 2$ in this example. The number of grid points N is repeatedly chosen such that the grid size Δx is halved each time. The results are given in Table 5.3 and clearly demonstrate optimal convergence for smooth functions with no critical points and fourth-order convergence for smooth functions with first-order critical points. Thus, the embedded WENO-JS scheme provides the same or similar performance as the original scheme for smooth functions.

Table 5.3: Convergence test for the embedded WENO-JS scheme applied to (5.53) (subscript 1) and (5.54) (subscript 2) with $c_2 = c_0 = 2$ and $\varepsilon = 10^{-40}$.

N	error_1	order_1	error_2	order_2
101	$2.0 \cdot 10^{-4}$		$1.6 \cdot 10^{-6}$	
201	$7.1 \cdot 10^{-6}$	4.8	$7.2 \cdot 10^{-8}$	4.5
401	$2.3 \cdot 10^{-7}$	4.9	$3.8 \cdot 10^{-9}$	4.2
801	$7.3 \cdot 10^{-9}$	5.0	$2.1 \cdot 10^{-10}$	4.2
1601	$2.3 \cdot 10^{-10}$	5.0	$1.3 \cdot 10^{-11}$	4.0

Embedded WENO-Z

A more contemporary version of WENO schemes is represented by the WENO-Z scheme of Borges et al. [85]. As mentioned earlier, the WENO-JS scheme has the property that $\omega_k = \gamma_k + \mathcal{O}(\Delta x^2)$ for smooth solutions, whereas the WENO-Z weights given by (5.29) satisfy $\omega_k^{\text{Z}} = \gamma_k + \mathcal{O}(\Delta x^{3p})$, with p the power parameter. Consequently, at critical points, the WENO-Z scheme avoids loss of convergence. A side-effect of the new weights is faster convergence to the linear weights in smooth regions. This also results in sharper resolution of discontinuities. The unnormalised weights for the WENO-Z scheme are defined as in (5.29).

Embedding an inner scheme into the WENO-Z scheme is somewhat easier, since the second general form (5.43) is a generalisation of WENO-Z. In the context of our framework, we have to satisfy the consistency conditions (5.44), i.e., $\sum_l a_{kl} = 0$ for $k = 0, 1, 2$. At the same time, we can obtain extra equations from (5.24), where we find the fourth-order term must cancel out as well, i.e.,

$$a_{k0} - \tfrac{1}{2} a_{k1} + a_{k2} = 0, \quad k = 0, 1, 2. \tag{5.56}$$

By Theorem 5.2, the two embedding equations are now given by (5.49). Thus, for an embedded version of WENO-Z, we have six equations from consistency and two embedding equations to solve for nine coefficients, yielding a one-parameter family of schemes, given by

$$\tilde{\omega}_0 = \gamma_0 \left(1 + \mu c_2 \left(\frac{\tau}{\beta_0 + \varepsilon} \right)^p \right), \tag{5.57a}$$

$$\tilde{\omega}_1 = \gamma_1 \left(1 + \mu \left(\frac{\tau}{\beta_1 + \varepsilon} \right)^p \right), \tag{5.57b}$$

$$\tilde{\omega}_2 = \gamma_2 \left(1 + \mu c_0 \left(\frac{\tau}{\beta_2 + \varepsilon} \right)^p \right), \tag{5.57c}$$

where μ is the free parameter, set to $\frac{1}{4}$ unless mentioned otherwise, and again $\tau = |\beta_0 - \beta_2|$. The scheme given by (5.57) is stable for $\mu > 0$, $c_0 > 0$ and $c_2 > 0$, which includes the options presented in Table 5.2. The power parameter is set to $p = 2$ throughout the rest of this work.

As earlier, we perform a convergence test to verify that the embedded WENO-Z scheme has the same order of convergence as its standard counterpart for smooth functions. The details can be found in the previous subsection as the test procedure is exactly the same. WENO-Z with $p = 2$ attains optimal convergence for smooth functions that may have first-order critical points. Only third-order accuracy is attained when the order of the critical points is higher. Therefore, only the second test is performed with initial condition (5.54), which features two first-order critical points. We furthermore set $c_2 = c_0 = 2$ and $\varepsilon = 10^{-40}$. The results are given in Table 5.4 and once again show optimal convergence.

Table 5.4: Convergence test for the embedded WENO-Z scheme applied to (5.54) (subscript 2) with $c_2 = c_0 = 2$ and $\varepsilon = 10^{-40}$.

N	error_2	order_2
101	$6.0 \cdot 10^{-7}$	
201	$1.8 \cdot 10^{-8}$	5.1
401	$5.9 \cdot 10^{-10}$	5.0
801	$1.8 \cdot 10^{-11}$	5.0
1601	$6.0 \cdot 10^{-13}$	5.0

5.2.4 Some last remarks on WENO

To show more of the article on embedded WENO at this point would be overkill. The rest of the article demonstrates the embedded WENO scheme applied to the linear advection equation and the Euler equations. Both show significant improvements over the standard versions. In another article, we've also shown how to apply the same strategy to seven-point stencils [103]. There, it turns out that for relatively easy test problems the performance is comparable, being to within a percent or so of the WENO-Z scheme. However, for a harder problem, the embedded WENO really shines and produces almost 25% reduction in the global error.

5.3 Application to Liouville's equation

Methods with a regular grid and stencil approach to hyperbolic PDEs can be applied to Liouville's equation as well, as we'll show in Chapter 8. The upwind method especially is a natural and obvious first try when encountering a hyperbolic problem. This entails placing a regular grid on phase space and using an upwind-biased difference to approximate the derivative on grid points. Away from any interfaces, the scheme is then already correct as the luminance distribution simply satisfies Liouville's equation.

Near the interface, some local adjustments to the scheme have to be made. This basic idea is reminiscent of front tracking methods [104–108]. However, front tracking is usually employed when the interface changes dynamically depending on the solution of the PDE on either side. In geometric optics problems, movement of the interface is given, as it's directly related to the shape of the optic. The major challenge in Liouville's equation is that the effects of the interface are nonlocal. Indeed, through Snell's law, completely different parts of phase space are in communication with each other. As we'll see, the consequence of this is that stencils become quite tangled and complicated.

5.4 Conclusion

In the following two chapters, we'll discuss two unstructured methods: the active flux scheme and the discontinuous Galerkin method. Both work with elements that have internal degrees of freedom. Elements are no longer defined on a regular grid but on a unstructured mesh. In particular, this means neither methods uses a stencil to define the approximation or to achieve higher orders. This has great advantages for optics problems. Whereas a stencil has to be locally adjusted when the interface cuts it, methods defined on elements don't have this problem provided the elements are suitably placed. Another possible advantage is mesh refinement. In places where high gradients are expected, the elements can be made smaller locally, resulting in higher accuracy around smaller elements and a more uniform accuracy globally. It's even possible to adapt the mesh dynamically, with the mesh responding to the solution [109–111].

Chapter 6

The active flux scheme

$\pi\alpha\nu\tau\alpha\ \rho\epsilon\iota$
(Everything is in flux)

Heraclitus

In the previous chapter, we've looked at two methods that work on a regular grid. Obtaining higher-order for those methods is accomplished by expanding the stencil. This may run into some complications when dealing with optical interfaces, as we'll argue at the end of Chapter 8. As a first alternative, we will investigate a high-order method defined on an unstructured mesh: the active flux scheme[1]. This necessarily makes it somewhat more complicated than methods defined on a regular grid. We'll illustrate the active flux scheme in a two-dimensional setting for scalar hyperbolic conservation laws. The active flux scheme has been successfully applied to systems like the Euler equations[2] and the shallow water equations [112–114].

The active flux scheme can be interpreted as adding point values to a finite volume scheme and solving for those separately. Interestingly, one can also interpret the active flux scheme as adding average values to the original scheme

[1] I first learned about the active flux scheme from Phil Roe at the 2015 Woudschoten conference. Phil was kind enough to help me along in trying to get the active flux scheme working for Liouville's equation.

[2] I find it truly astonishing how many things in science and mathematics are named after Euler. This is illustrated well by what an instructor I had on a fluid dynamics course once said: "This system of PDEs was discovered by Euler as well, but we can't go about calling everything Euler's equations."

proposed by Courant, Isaacson and Rees discussed in the previous chapter [72]. That scheme used only point values that are updated by integrating backwards along characteristics. However, being derived from the advective form, the CIR scheme is not conservative. By explicitly adding average values and demanding conservation, the active flux scheme results.

Consider a scalar hyperbolic conservation law with flux vector $\mathbf{f}$, i.e.,

$$\frac{\partial u}{\partial t} + \nabla \cdot \mathbf{f}(u) = 0. \tag{6.1}$$

One approach to the numerical solution of this problem is by a finite volume approach. We average (6.1) over a domain Ω and apply the divergence theorem, i.e.,

$$\frac{\mathrm{d}\bar{u}}{\mathrm{d}t} = -\frac{1}{|\Omega|}\int_{\Omega} \nabla \cdot \mathbf{f}(u)\,\mathrm{d}A = -\frac{1}{|\Omega|}\oint_{\partial\Omega} \mathbf{f}(u)\cdot\mathbf{n}\,\mathrm{d}s, \tag{6.2}$$

where we've divided by $|\Omega|$, the area of Ω. Here, $\bar{u}$ is the average of u over the volume while $\mathbf{n}$ is the outward unit normal. The integral over the closed curve is to be traversed counter-clockwise as per the right-hand rule. It's important to note that the flux across the boundary completely determines the rate of change of the average value of u inside the volume. This is the essence of any finite volume method.

Pressing (6.2) a bit more, the boundary fluxes are in turn determined by point values on the boundary. Telescoping the argument, the rate of change of the average value is determined by the point values on the boundary. This key observation is the basis of the active flux scheme, where averages and point values are kept track of *separately*. In fact, the name "active flux" originates from this distinction. Other finite volume schemes that don't actively keep track of boundary values may be classified as "passive flux" schemes.

6.1 Finding average values

As a start, we'll assume that point values are somehow known (by magic or sorcery, take your pick). We'll first discuss the time stepping of the average values. How to advance the point values is shown afterwards. For those who cannot contain their curiosity: the point values are updated using a semi-Lagrangian step.

6.1.1 The standard domain

The computational mesh is assumed to be a triangulation, possibly of some complex shape, for instance using Ruppert's Algorithm [115]. The shape of Ω is thereby a triangle, let's fix the vertices as $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ where $\mathbf{x} = (x, y)$. Any triangle may be mapped to a reference triangle χ by an affine transformation, whose vertices are $(0, 0)$, $(1, 0)$ and $(0, 1)$, see Figure 6.1. The Cartesian coordinates in the reference domain are denoted $\boldsymbol{\xi} = (\xi, \eta)$. The affine transformation from the reference domain to Ω reads

$$\begin{pmatrix} x \\ y \end{pmatrix} = \boldsymbol{A} \begin{pmatrix} \xi \\ \eta \end{pmatrix} + \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}, \tag{6.3a}$$

where $\boldsymbol{A}$ is given by

$$\boldsymbol{A} = \begin{pmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{pmatrix}. \tag{6.3b}$$

Note both $\mathbf{x}$ and $\boldsymbol{\xi}$ are Cartesian coordinates. Let's denote the determinant of $\boldsymbol{A}$ as $\mathcal{J}$, then $|\mathcal{J}| = |\Omega|/|\chi| = 2|\Omega|$. We label the vertices in such a way that $\mathcal{J} > 0$, corresponding to a counter-clockwise orientation of the labels. Furthermore, $\mathcal{J} = 0$ occurs only when the vertices are collinear, clearly a situation we'd like to avoid.

The derivatives in the reference coordinates are related to the derivatives on Ω by

$$\frac{\partial}{\partial x} = \frac{\partial \xi}{\partial x}\frac{\partial}{\partial \xi} + \frac{\partial \eta}{\partial x}\frac{\partial}{\partial \eta}, \tag{6.4a}$$

$$\frac{\partial}{\partial y} = \frac{\partial \xi}{\partial y}\frac{\partial}{\partial \xi} + \frac{\partial \eta}{\partial y}\frac{\partial}{\partial \eta}. \tag{6.4b}$$

This component-wise expression can be converted to a matrix-vector form by realising that $\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{x}} = \boldsymbol{A}^{-1}$, hence

$$\nabla = \nabla_{\mathbf{x}} = \boldsymbol{A}^{-T} \nabla_{\boldsymbol{\xi}}, \tag{6.5}$$

where $\nabla_{\boldsymbol{\xi}} = (\frac{\partial}{\partial \xi}, \frac{\partial}{\partial \eta})^T$ is the gradient operator in the standard coordinates.

The area integral in (6.2) is transformed to the reference domain, i.e.,

$$\frac{\mathrm{d}\bar{u}}{\mathrm{d}t} = -2 \int_{\chi} \left(\boldsymbol{A}^{-T} \nabla_{\boldsymbol{\xi}} \right) \cdot \mathbf{f}(u) \, \mathrm{d}A_{\boldsymbol{\xi}} = -2 \int_{\chi} \nabla_{\boldsymbol{\xi}} \cdot \left(\boldsymbol{A}^{-1} \mathbf{f}(u) \right) \mathrm{d}A_{\boldsymbol{\xi}}, \tag{6.6}$$

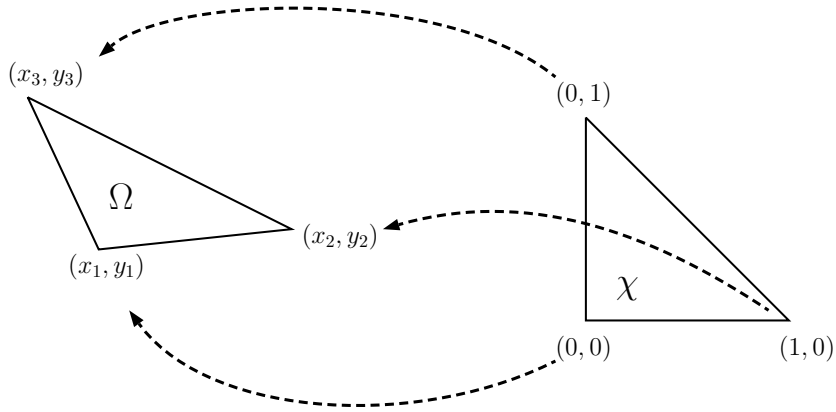


Figure 6.1: Mapping from the standard to a general triangle.

the last equality follows from the fact that $\boldsymbol{A}$ is a constant matrix. This allows us to define

$$\tilde{\mathbf{f}}(u) = \boldsymbol{A}^{-1}\mathbf{f}(u). \tag{6.7}$$

Applying the divergence theorem on the reference domain, the finite volume formulation yields

$$\frac{\mathrm{d}\bar{u}}{\mathrm{d}t} = -2 \oint_{\partial\chi} \tilde{\mathbf{f}}(u) \cdot \mathbf{n} \, \mathrm{d}\sigma, \tag{6.8}$$

which is simply the volume-averaged total boundary flux in the standard domain since $|\chi| = \frac{1}{2}$. Again, $\mathbf{n}$ is the outward pointing normal and the orientation of the closed curve $\partial\chi$ is counter-clockwise in accordance with the right-hand rule.

6.1.2 Approximation of the fluxes

To proceed, we need to approximate the right-hand side of (6.8). First, we can split the boundary integral into the separate line integrals over each edge. Define f_k as the flux over edge ℓ_k, i.e.,

$$f_k(u) = 2 \int_{\ell_k} \tilde{\mathbf{f}}(u) \cdot \mathbf{n}_k \, \mathrm{d}\sigma, \tag{6.9}$$

where ℓ_k, $k = 1, 2, 3$ are the edges of the reference triangle and $\mathbf{n}_k$ are their outward unit normals. The total boundary flux is then the sum of the edge-wise

contributions. Each line integral can now be approximated by a one-dimensional quadrature rule. Here, we choose Simpson's rule to approximate the integrals, resulting in the addition of the midpoints of each side to the collection of nodes, see Figure 6.2.

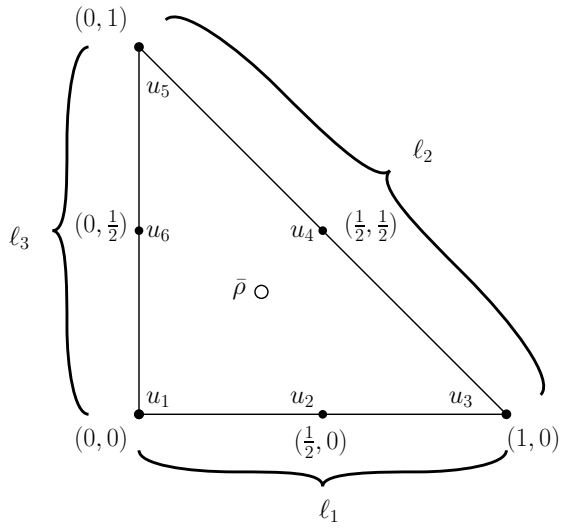


Figure 6.2: Gauß-Lobatto nodes around the reference triangle χ with values labelled that should be tracked. The edges are also indicated.

Simpson's rule is the three-node case of the Gauß-Lobatto[3] quadrature rules, being exact for polynomials up to degree three [40]. Therefore, if a higher-order quadrature is desirable, that's the natural extension. The Gauß-Lobatto rules have a slight advantage over the Gauß rules since each vertex, which is already needed to determine the mesh, can be used twice. Otherwise, the order of convergence is equal, which is the content of the following lemma[4].

Lemma 6.1. *Given a polygon, Gauß and Gauß-Lobatto rules need exactly the same number of nodes to provide exact integration over the boundary for a poly-*

[3]I always mention in my presentations that Rehuel Lobatto was a Dutch mathematician born from Portuguese parents. Most people aren't aware of it.

[4]I don't know if this lemma is known in the literature. I've never seen it mentioned in textbooks on spectral methods, which is where I'd expect to find it, if it were already known. I think it's rather nice though.

nomial of degree p. Specifically, $s(p+1)/2$ points in total are needed to provide exact integration on a polygon of s sides for all polynomials of degree p.

Proof. 1. The degree of exactness for Gauß quadrature is $p = 2n_G - 1$, where n_G is the number of nodes per side. Note that for the Gauß nodes, there are no nodes on the vertices of the polygon. The total number of nodes needed to achieve degree of exactness p is therefore $s(p+1)/2$.

2. For Gauß-Lobatto nodes, the degree is $p = 2n_{GL} - 3$, with n_{GL} the number of nodes per side including the boundary points. The number of nodes per side needed to achieve degree of exactness p is therefore $(p+3)/2$. However, the nodes on the vertices can be used twice, once for either edge connecting to it. Hence, s nodes in total can be used twice, so that the total number of nodes needed becomes $s(p+3)/2 - s = s(p+1)/2$. □

According to the lemma, in order to integrate cubic polynomials exactly, we need six nodes in total distributed around the boundary of the reference triangle. Another advantage of the Gauß-Lobatto nodes is that they allow a slightly larger time step, but more on that later.

We label the nodes as in Figure 6.2 and define the unit direction vectors $\mathbf{e}_1 = (1,0)^T$ and $\mathbf{e}_2 = (0,1)^T$, so that we may approximate the flux over each edge as follows,

$$f_1(\boldsymbol{u}) \approx -\tfrac{1}{3}\left(\tilde{\mathbf{f}}(u_1) + 4\tilde{\mathbf{f}}(u_2) + \tilde{\mathbf{f}}(u_3)\right) \cdot \mathbf{e}_2, \tag{6.10a}$$

$$f_2(\boldsymbol{u}) \approx \tfrac{1}{3}\left(\tilde{\mathbf{f}}(u_3) + 4\tilde{\mathbf{f}}(u_4) + \tilde{\mathbf{f}}(u_5)\right) \cdot (\mathbf{e}_1 + \mathbf{e}_2), \tag{6.10b}$$

$$f_3(\boldsymbol{u}) \approx -\tfrac{1}{3}\left(\tilde{\mathbf{f}}(u_5) + 4\tilde{\mathbf{f}}(u_6) + \tilde{\mathbf{f}}(u_1)\right) \cdot \mathbf{e}_1, \tag{6.10c}$$

where $\boldsymbol{u} = (u_1, u_2, u_3, u_4, u_5, u_6)^T$ is the vector of point values. Note that the normalisation of $\mathbf{n}_2$ cancels against the length of the hypotenuse. The total boundary flux is approximated by

$$F(\boldsymbol{u}) = \sum_{k=1}^{3} f_k(\boldsymbol{u}). \tag{6.11}$$

The spatial discretisation lead to the following semi-discrete scheme,

$$\frac{\mathrm{d}\bar{u}}{\mathrm{d}t} = -F(\boldsymbol{u}). \tag{6.12}$$

To emphasise: the vector $\boldsymbol{u}$ denotes the collection of six point values on the boundary of the element, while $\bar{u}$ represents the average value over the element.

Recall that we've assumed, for the moment, that we're given a method to find the point values at any given time. We can therefore treat the right-hand side of (6.12) as a function of time only. Thus, computing the averages reduces to evaluating a *definite* integral.

To conclude, we introduce the time discretisation $t^n = n\Delta t$ with $n = 0, 1, \ldots, M$ with M the number of steps and $\Delta t = \frac{T}{M}$ for a given integration time T. In order to find the average value at the next time level, we again employ Simpson's rule, as time integration then proceeds with the same order of accuracy as the spatial discretisation. This leads to

$$\bar{u}^{n+1} = \bar{u}^n - \frac{\Delta t}{6}\left(F^n + 4F^{n+\frac{1}{2}} + F^{n+1}\right), \tag{6.13}$$

where F^n is the approximation of the flux integral, $F^n \approx F(\boldsymbol{u}(t^n))$, etc. Note that with this choice of time integration, we need to know the total flux over the boundary at three time levels: t^n, $t^{n+\frac{1}{2}}$ and t^{n+1}. Hence, we need to compute point values at these time levels, which is the topic of the next section.

The above discussion is valid per element, so that for every element the fluxes over the edges in the physical domain have to be transformed to fluxes in the standard domain. The time step update for the averages is done by simply looping over all elements, which has potential for parallelisation.

6.2 Finding point values

For each time step, we need to find the point values at the time levels t^n, $t^{n+\frac{1}{2}}$ and t^{n+1}. The scheme is completed by using a semi-Lagrangian step [116]. The idea stems from the method of characteristics (MOC) and is exactly the same as the time stepping in the scheme of Courant, Isaacson and Rees, see Chapter 5.

The conservation law (6.1) is first rewritten to the advective form, yielding

$$\frac{\partial u}{\partial t} + \mathbf{f}'(u) \cdot \nabla u = 0. \tag{6.14}$$

Suppose now that we follow the solution along curves $\mathbf{x}(t)$, so that we can define $u^\star(t) = u\big(t, \mathbf{x}(t)\big)$. Differentiating with respect to time yields

$$\frac{\mathrm{d}}{\mathrm{d}t}u^\star(t) = \frac{\partial u}{\partial t} + \mathbf{x}'(t) \cdot \nabla u, \tag{6.15}$$

where u is to be evaluated at $\big(t, \mathbf{x}(t)\big)$. For the special choice of $\mathbf{x}(t)$ being characteristics, i.e.,

$$\mathbf{x}'(t) = \mathbf{f}'\big(u^\star(t)\big), \tag{6.16}$$

$u^\star$ becomes a constant, since (6.15) becomes equal to (6.14) and therefore vanishes. Consequently, the right-hand side of (6.16) becomes constant as well, so that the characteristics are in fact straight lines.

We solve (6.16) backward in time using the terminal condition that $\mathbf{x}(t^{n+1}) = \mathbf{x}^{n+1}$ is located at a node. This is the essence of a semi-Lagrangian method. If $u^\star$ were known, the solution for $\mathbf{x}$ would be

$$\mathbf{x}(t) = \mathbf{x}^{n+1} + (t - t^{n+1})\mathbf{f}'(u^\star). \tag{6.17}$$

Employing this relation in a time-stepping fashion, this leads to an implicit relation for $u^\star$, i.e.,

$$u(t^n, \mathbf{x}(t^n)) = u\big(t^n, \mathbf{x}^{n+1} - \Delta t \mathbf{f}'(u^\star)\big) = u^\star. \tag{6.18}$$

This equation for $u^\star$ provides an exact solution for u. However, at the previous time level, t^n, we only know an approximation to u. Therefore, the exact solution is simply replaced by the approximation at t^n, which is denoted u_{int}. How the approximation u_{int} is constructed in explained in the next section. Note that the approximation holds at time level t^n, so that it has no time argument. We find that $u^\star$ satisfies

$$u^\star = u_{\text{int}}\big(\mathbf{x}^{n+1} - \Delta t \mathbf{f}'(u^\star)\big), \tag{6.19}$$

which is solved using, e.g., Newton iteration. Note that $\mathbf{x}(t^{n+\frac{1}{2}})$, which is also required by (6.13), can be found cheaply, since the characteristic on which it lies is already found. In particular

$$\mathbf{x}(t^{n+\frac{1}{2}}) = \mathbf{x}^{n+1} - \tfrac{1}{2}\Delta t \mathbf{f}'(u^\star), \tag{6.20}$$

where at this point $u^\star$ is already determined. As a final note, we mention that the positions $\mathbf{x}(t^n)$ and $\mathbf{x}(t^{n+\frac{1}{2}})$ only have to be determined to second-order accuracy in Δt. This is due to the quadrature over the boundary, giving the total flux a mixed third-order error, more details are given in Section 6.4.

6.2.1 Reconstruction

The approximation in each element consists of a polynomial reconstruction, made up of a quadratic interpolation of the point values together with a bubble function that compensates for the average value. The bubble function, as we'll see, has zero value on the boundary, so that interpolation of the point values

isn't disturbed. In the following discussion, we again omit any element markers for brevity.

The quadratic interpolation on the reference domain is given by

$$\tilde{u}(\boldsymbol{\xi}) = \tfrac{1}{2}\boldsymbol{\xi}^T \boldsymbol{R}\boldsymbol{\xi} + \mathbf{r}^T\boldsymbol{\xi} + u_1, \tag{6.21}$$

with $\boldsymbol{\xi} = (\xi, \eta)^T$. The symmetric matrix $\boldsymbol{R} \in \mathbb{R}^{2\times 2}$ approximates the Hessian while the vector $\mathbf{r} \in \mathbb{R}^2$ approximates the gradient, both at $\boldsymbol{\xi} = 0$. To determine them, we simply substitute the node locations and demand that the value of $\tilde{u}$ equals the given point values. This leads to

$$\boldsymbol{R} = 4\begin{pmatrix} u_1 - 2u_2 + u_3 & u_1 - u_2 + u_4 - u_6 \\ u_1 - u_2 + u_4 - u_6 & u_1 - 2u_6 + u_5 \end{pmatrix} \tag{6.22a}$$

and

$$\mathbf{r} = \begin{pmatrix} -3u_1 + 4u_2 - u_3 \\ -3u_1 + 4u_6 - u_5 \end{pmatrix}, \tag{6.22b}$$

where one can recognise various undivided differences over the ξ- and η-directions, with the off-diagonal elements of $\boldsymbol{R}$ representing the mixed spatial derivative. For instance, the vector $\mathbf{r}$ is composed of one-sided second-order differences.

The eagle-eyed observer will notice that this interpolation does not necessarily accommodate the average value $\bar{u}$. Indeed, the average value of $\tilde{u}$, given by

$$2\iint_\chi \tilde{u}(\xi,\eta)\,\mathrm{d}\xi\,\mathrm{d}\eta = \tfrac{1}{3}\left(u_2 + u_4 + u_6\right), \tag{6.23}$$

may certainly be different from $\bar{u}$. To ensure that the average value of the reconstruction is equal to $\bar{u}$, and to thus ensure that the scheme is conservative, we must add a third-order function that compensates for the deficit. This function must furthermore not interfere with the quadratic interpolation already established, hence it must be zero on the triangle edges. Such a function is called a bubble function, which can be interpreted as the product of all three barycentric coordinates, i.e.,

$$\varphi(\xi,\eta) = 60\xi\eta(1-\xi-\eta), \tag{6.24}$$

where the normalisation constant is chosen such that the bubble function has unit average value, i.e., $\iint_\chi \varphi\,\mathrm{d}\xi\,\mathrm{d}\eta = |\chi| = \frac{1}{2}$. As a consequence, the total local reconstruction of the solution on the reference domain is given by

$$u_{\mathrm{int}}(\xi,\eta) = \tilde{u}(\xi,\eta) + \left(\bar{u} - \tfrac{1}{3}(u_2+u_4+u_6)\right)\varphi(\xi,\eta). \tag{6.25}$$

6.3 Summary of the algorithm

The algorithm resulting from the synthesis of the previous discussion. As an input, it requires a triangular mesh and initial conditions supplied in terms of point values on the nodes and average values over the triangular elements. Note that there's a one-way coupling between the point and average values. In particular, the point values are influenced by the point and average values of the previous time level, while the average values are influenced only by the point values of the current time level. The output is the set of point and average values at the final time T. The integration time is divided into an integer number, M, of time step, so that $\Delta t = \frac{T}{M}$. The time step Δt needs to be sufficiently small to ensure stability, more on that in the next section. Let's denote the current time level as t^n, then one time step of the algorithm is given by Algorithm 10. The time stepping algorithm is itself of course performed M times.

Algorithm 10 One time step of Active Flux

for each *node* in the mesh **do**
1) Solve (6.19) for $u^\star$ to find the point value at t^n.
2) Compute $\mathbf{x}^{n+\frac{1}{2}}$ by plugging the result into (6.20).
Keep track of which neighbouring element
the characteristic originates from.
3) Transform $\mathbf{x}^{n+\frac{1}{2}}$ to the standard coordinates, i.e.,
apply the inverse transformation to (6.3a).
4) Find the point value at $t^{n+\frac{1}{2}}$ using the reconstruction,
detailed in (6.21), (6.22), (6.24) and (6.25).
end for
for each *element* in the mesh **do**
1) Compute the flux function at t^n and $t^{n+\frac{1}{2}}$ using (6.10) and (6.11).
2) Update the average value using (6.13).
end for

Let's now investigate the time scaling of the algorithm. The first for-loop runs over all nodes in the mesh, where each substep requires a constant amount of work. The second for-loop runs over all elements, where again each substep requires a constant amount of work. Thus, the total amount of work in Algorithm 10 scales with $\mathcal{O}(N_{\text{nodes}} + N_{\text{elements}})$. Interpreting the mesh as a planar graph, the Euler characteristic[5] tells us that the number of nodes scales linearly

[5]Euler invented graph theory to solve the famous Seven Bridges of Königsberg problem.

with the number of elements. The algorithm is repeated M times, whence we can conclude that the work needed by the algorithm scales as $\mathcal{O}(MN_{\text{elements}})$.

6.4 CFL condition and error behaviour

Like many hyperbolic discretisation methods, the active flux scheme is also subject to a CFL condition, i.e.,

$$\Delta t \leq \frac{h_{\min}}{b_{\max}}, \tag{6.26}$$

where $h_{\min}$ is the smallest spatial size in the mesh and $b_{\max}$ is the maximum value of the norm of the velocity field. For the active flux scheme, the CFL condition arises by the assumption that the point values are retrieved from neighbouring elements. Note that this assumption is not necessary. We may, for instance, perform a search over all elements every time, which would result in a very slow algorithm with unconditional stability. Hence, restricting the search set is mainly done for speed.

Thus, for each node, we restrict the search set to all elements for which the node in question lies on the boundary. For stability, the analytical domain of dependence must lie entirely within the numerical domain of dependence. The numerical domain of dependence for a node is the set of neighbouring elements. The analytical domain of dependence can be bounded by a circle of radius $b_{\max}\Delta t$ centred on the node, which we call the bounding circle. Note that this is a rather pessimistic, though safe, estimate. A better estimate can be obtained by using, for instance, an ellipse to bound the analytical domain of dependence. The time step Δt must be chosen such for all nodes, the bounding circle lies entirely within the search set, see Figure 6.3.

The shortest distance from each node to the boundary of its numerical domain of dependence can be found by dropping an altitude line. Clearly, the shortest altitude line in the mesh originates from a midpoint. Calling the edge length a, basic trigonometry tells us that $\Delta y = \frac{1}{2}a\sin\alpha$, which must be bigger than the radius of the bounding circle. Thus, the minimum angle in the mesh $\alpha_{\min}$ has some influence on the overall stability of the scheme. Moreover, the minimum angle bounds the maximum number of neighbouring elements. Nodes on the midpoints of triangle edges will always have two elements as their direct neighbours. However, the maximum number of neighbouring elements of vertex

One remarkable result is now known as the Euler characteristic, see Richeson's excellent book for an in-depth discussion [117].

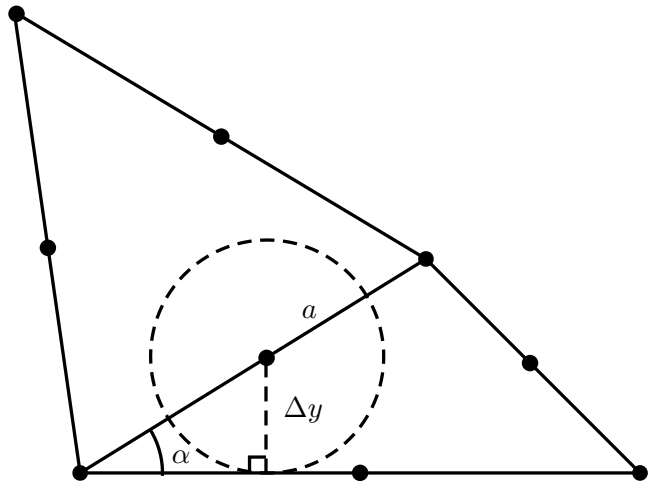


Figure 6.3: Sketch of the numerical domain of dependence of a node, its neighbouring elements, and a bound on the analytical domain of dependence, the circle. The radius of the circle is $b_{\max}\Delta t$.

nodes can be bounded by $\frac{2\pi}{\alpha_{\min}}$. For these two reasons, the minimum angle often serves as a measure for mesh quality [115].

Most mesh generators have a minimum angle that can be set, we've used the Triangle software package created by Shewchuck [118], where a minimum angle of 32° usually works well. This allows us to estimate the minimal distance as

$$h_{\min} = \tfrac{1}{2} \min_i (a_i \sin \alpha_i) \geq \tfrac{1}{2} \sin \alpha_{\min} \min_i a_i, \tag{6.27}$$

where α_i and a_i are the collection of angles and edges in the mesh for $i = 1, 2, \ldots$. To find the maximum velocity simply requires a loop over all nodes in the mesh, evaluating the local velocity field.

With the CFL condition, the scaling behaviour between spatial and temporal grid sizes is straightforward: $\Delta t = \mathcal{O}(h_{\min})$. Earlier, we asserted that the scheme only needs to resolve the point values to second order accuracy in Δt. The flux is integrated around the boundary so that the fluxes carry a mixed error of $\mathcal{O}(h_{\min}\Delta t^2)$. Thus, the scaling of the CFL condition then implies that the total error scales in the fluxes scales as $\mathcal{O}(h_{\min}^3)$.

6.5 Moving mesh

In some cases, it's convenient or necessary to align the mesh with moving physical boundaries. An example in aerospace engineering might be the deployment of a flap, while an example in optics simply involves curved interfaces. In some cases, the changes will be so sudden or great that a completely new mesh has to be generated. This involves running the meshing software and interpolating all values from the old to the new mesh. On the other hand, when the change of the physical boundaries is sufficiently slow or smooth, such a complete remeshing can be avoided by interpreting the changes as motion of several or all nodes.

On a moving mesh, the active flux formulation has to be slightly adjusted. Ding et al. have developed an active flux scheme on a moving mesh by considering space-time elements as a whole [119]. We take a different, and as as we're aware novel, approach and leave time continuous, fitting in with the method of lines paradigm. As usual, we can afterwards apply any numerical integrator.

Let's assume that the motion of the mesh is given and we're able to compute time derivatives exactly. This happens in some cases, as we'll see in Chapter 9. Other times, the motion of the mesh is only specified in terms of a dynamical system, e.g. a spring-dashpot system. For now, we'll take mesh node positions and velocities as given. Consider Reynolds' Transport Theorem on the, now moving, test domain $\Omega(t)$, i.e.,

$$\frac{\mathrm{d}}{\mathrm{d}t} \int\limits_{\Omega(t)} u \,\mathrm{d}A = \int\limits_{\Omega(t)} \frac{\partial u}{\partial t} + \nabla \cdot (u\mathbf{v}) \,\mathrm{d}A, \tag{6.28}$$

where $\mathbf{v}$ is the velocity field of the moving element. The vector field $\mathbf{v}$ can be found by differentiating (6.3a), i.e.,

$$\mathbf{v}(t) = \boldsymbol{A}'(t) \begin{pmatrix} \xi \\ \eta \end{pmatrix} + \mathbf{x}_1'(t), \tag{6.29}$$

with the primes denoting differentiation with respect to t. The matrix $\boldsymbol{A}(t)$ is still given by (6.3b), but now the vertex positions are allowed to depend on time. Applying the hyperbolic conservation law (6.1), we find

$$\frac{\mathrm{d}}{\mathrm{d}t} \int\limits_{\Omega(t)} u \,\mathrm{d}V = - \int\limits_{\Omega(t)} \nabla \cdot \big(\mathbf{f}(u) - u\mathbf{v}\big) \,\mathrm{d}V. \tag{6.30}$$

The area integral of u is by definition equal to the average value times the area

size. Furthermore, transforming the integral to the standard domain yields

$$\frac{\mathrm{d}}{\mathrm{d}t}\Big(\bar{u}|\Omega(t)|\Big) = -\int_{\chi} \mathcal{J}(t)\big(\boldsymbol{A}^{-T}(t)\nabla_{\boldsymbol{\xi}}\big)\cdot\big(\mathbf{f}(u)-u\mathbf{v}\big)\,\mathrm{d}V_{\boldsymbol{\xi}}. \tag{6.31}$$

Compare this ODE for $\bar{u}$ to (6.6). Recall that we've labelled the vertices such that $\mathcal{J} > 0$ and $\mathcal{J}(t) = 2|\Omega(t)|$. This also shows that the determinant is constant in the spatial coordinates, so that it may be taken outside the integral. Note that some grid movements may cause the determinant to become negative or the minimum angle to fall below a certain threshold. In these cases, to ensure stability, a complete remeshing is unavoidable.

The matrix $\boldsymbol{A}$, though now time dependent, is still constant with respect to the spatial coordinates. Therefore, the flux on the reference domain χ can be defined as

$$\tilde{\mathbf{f}}(t,u) = \boldsymbol{A}^{-1}(t)\left(\mathbf{f}(u)-u\mathbf{v}\right). \tag{6.32}$$

Applying the divergence theorem on the reference domain, we obtain

$$\frac{\mathrm{d}}{\mathrm{d}t}\big(\tfrac{1}{2}\bar{u}\mathcal{J}\big) = -\mathcal{J}\oint_{\partial\chi}\tilde{\mathbf{f}}(t,u)\cdot\mathbf{n}\,\mathrm{d}\sigma, \tag{6.33}$$

where we've replaced the volume on the left-hand-side by the determinant. Once again, $\mathbf{n}$ is the outward pointing normal and the orientation of the closed curve integral is counter-clockwise, conforming to the right-hand rule. This ODE for $\bar{u}$ should be compared to (6.8). Clearly, if the mesh is static, (6.33) reduces to (6.8) since $\mathcal{J}$ is constant in that case.

In some situations, there may be some ambient free-stream flux, corresponding to $u = \text{const.}$ since then the flux is constant too. An example from aerospace would be the ambient free stream far from an airfoil. Close to the surface of the airfoil, all sorts of complicated boundary layers are forming while far away the air is just whizzing by at a constant speed. Given a constant flux, we'd like the numerical solution to be constant as well, a property we might call 'free-stream preserving'. This is possible by considering the evolution of the volume $|\Omega|$, obtained from Reynold's transport theorem, leading to

$$\frac{\mathrm{d}\mathcal{J}}{\mathrm{d}t} = 2\mathcal{J}\oint_{\partial\chi}\widetilde{\mathbf{v}}\cdot\mathbf{n}\,\mathrm{d}\sigma, \tag{6.34}$$

where $\widetilde{\mathbf{v}} = \boldsymbol{A}^{-1}(t)\mathbf{v}$. Alternatively, one can set $u = 1$ in (6.33) to derive (6.34). If the numerical solution is to be free-stream preserving, we need (6.34) and (6.33) to be satisfied simultaneously in a discrete sense.

Let's approximate the flux integral of (6.33) as outlined in Section 6.2. The semi-Lagrangian step allows us to find the flux integral as a function of time only. This provides no serious complications, as the semi-Lagrangian approach only requires the mesh position at time t^{n+1}. In other words, the semi-Lagrange step doesn't even see the mesh motion. Furthermore, since the velocity field is given, the flux integral in (6.34) can also be computed as a function of time, so that after approximation the integrals, we have a coupled system of equations, i.e.,

$$\frac{\mathrm{d}}{\mathrm{d}t}\big(\bar{u}\mathcal{J}\big) = -\mathcal{J}(t)F(t), \tag{6.35a}$$

$$\frac{\mathrm{d}\mathcal{J}}{\mathrm{d}t} = \mathcal{J}(t)G(t), \tag{6.35b}$$

where $F(t)$ approximates the total volume-averaged flux across the boundary and $G(t)$ approximates of the right-hand side of (6.34). Note that if u is a constant, then also $F(t) = -uG(t)$. This follows from the fact that the boundary flux integration is exact if the flux is any cubic polynomial. In particular, it's exact for a constant, so that the divergence theorem implies that the integral of the flux is zero. The only contribution is therefore from the movement of the boundary.

Similar to the results of Acosta Minoli et al. for spectral element methods on moving meshes [120], we find that these equations must be integrated simultaneously using the same time integrator. Perhaps counter-intuitively, a free-stream preserving scheme needs to use the values obtained from numerical integration of (6.35b) rather than the exact value of the Jacobian, even if it is available. This is demonstrated in the following theorem.

Theorem 6.1. *Consider the system* (6.35)*, where we define* $\tilde{u} = \bar{u}\mathcal{J}$*, so that*

$$\frac{\mathrm{d}\tilde{u}}{\mathrm{d}t} = -\mathcal{J}(t)F(t), \tag{6.36a}$$

$$\frac{\mathrm{d}\mathcal{J}}{\mathrm{d}t} = \mathcal{J}(t)G(t). \tag{6.36b}$$

Then the time integration is free-stream preserving if this system is integrated with any general linear method and the average value is defined as

$$\bar{u}^n := \frac{\tilde{u}^n}{\mathcal{J}^n}. \tag{6.36c}$$

Proof. Let's write one time step of the integrator as ψ, i.e., $\tilde{u}^{n+1} = \tilde{u}^n + \psi(-\mathcal{J}^n F^n)$ where $\mathcal{J}^n$ is the approximation to $\mathcal{J}(t^n)$ etc. Any general linear method satisfies $\psi(cx) = c\psi(x)$ for any $c \in \mathbb{R}$. Next, assume that u is constant, so that $F(t) = -uG(t)$ and $\tilde{u}^n = u\mathcal{J}^n$, then we have

$$\begin{aligned}\tilde{u}^{n+1} = \tilde{u}^n + \psi(-\mathcal{J}^n F^n) = \tilde{u}^n + \psi(u\mathcal{J}^n G^n) = \tilde{u}^n + u\psi(\mathcal{J}^n G^n) \\ = u\big(\mathcal{J}^n + \Psi(\mathcal{J}^n G^n)\big) = u\mathcal{J}^{n+1}.\end{aligned}$$

The average value is recovered as $\bar{u}^{n+1} = \frac{u\mathcal{J}^{n+1}}{\mathcal{J}^{n+1}} = u$, so that the constant state is preserved. □

Remark. *General linear methods are a broad class of numerical integrators [35]. The class contains all Runge–Kutta methods, linear multistep methods and all predictor-corrector methods composed thereof.*

The theorem ensures that any Runge–Kutta method will provide a free-stream preserving scheme in combination with the space discretisation as introduced earlier. Moreover, for one-step methods, the exact value of the Jacobian $\mathcal{J}$ can be inserted at the beginning of each time step. We'll use the standard fourth-order RK4 method, as this reduces to Simpson's rule whenever $\mathcal{J}$ is constant. Note that the system (6.35) is a one-way coupled system, where $\bar{u}$ depends on $\mathcal{J}$, but $\mathcal{J}$ can be computed independently under the assumption that the mesh motion is doesn't depend on u. Let's therefore denote the RK4 stages of (6.35b) as

$$\mathcal{J}^{(1)} = \mathcal{J}(t^n), \tag{6.37a}$$
$$\mathcal{J}^{(2)} = \mathcal{J}(t^n) + \tfrac{1}{2}\Delta t\mathcal{J}^{(1)}G^n, \tag{6.37b}$$
$$\mathcal{J}^{(3)} = \mathcal{J}(t^n) + \tfrac{1}{2}\Delta t\mathcal{J}^{(2)}G^{n+1/2}, \tag{6.37c}$$
$$\mathcal{J}^{(4)} = \mathcal{J}(t^n) + \Delta t\mathcal{J}^{(3)}G^{n+1/2}, \tag{6.37d}$$

where we've used the shorthand notation $G^n = G(t^n)$ etc. We denote the exact value of the Jacobian at time level t^n as $\mathcal{J}(t^n)$. The updates for the average value are found using Theorem 6.1, in particular (6.36c). The resulting time integration scheme is therefore given by

$$\mathcal{J}^{n+1} = \mathcal{J}(t^n) + \frac{\Delta t}{6}\left[\mathcal{J}^{(1)}G^n + 2\left(\mathcal{J}^{(2)} + \mathcal{J}^{(3)}\right)G^{n+1/2} + \mathcal{J}^{(4)}G^{n+1}\right], \tag{6.38a}$$

$$\bar{u}^{n+1} = \frac{\mathcal{J}(t^n)}{\mathcal{J}^{n+1}}\bar{u}^n - \frac{\Delta t}{6\mathcal{J}^{n+1}}\left[\mathcal{J}^{(1)}F^n + 2\left(\mathcal{J}^{(2)} + \mathcal{J}^{(3)}\right)F^{n+1/2} + \mathcal{J}^{(4)}F^{n+1}\right], \tag{6.38b}$$

which is simply the RK4 method applied to (6.35) together with (6.36c). By definition of G, we see that if the grid is fixed, we have $G = 0$. From (6.37) - (6.38), one can clearly see that the Jacobian is constant and the scheme (6.38b) reduces to Simpson's rule in t, i.e., all the way to (6.13). Moreover, whenever u is constant, Theorem 6.1 ensures us that the constant state is preserved. This is also easy to check as then $F^n = -uG^n$, etc., so that $\bar{u}^{n+1} = u$.

6.6 Application to Liouville's equation

The active flux scheme is an excellent method for Liouville's equation for several reasons. Firstly, it's conservative. As we've explained in Chapter 4, symplectic methods in the Lagrangian picture translate roughly to conservative methods in the Eulerian picture. Secondly, it uses the method of characteristics to update the point values on the mesh nodes. In the context of optics, this is better known as ray tracing. This means the point values can be updated using symplectic integrators if so desired. Thirdly, it's locally defined on elements. In the previous chapter, we've briefly mentioned the complications that arise when stencils are crossed by the interface. These complications won't occur as long as we make sure the mesh is properly aligned with any interfaces. We'll explore the application of the active flux scheme to Liouville's equation fully in Chapter 9.

6.7 Interpretation as a spectral method

During our investigation into the active flux scheme, we became more and more aware of the similarities with spectral element methods. However, the active flux scheme is a funny sort of spectral element method, as it uses a mix between modal and nodal representations. Nodal means that point values are directly used, while modal means a representation in terms of basis functions, for instance Fourier modes.

The active flux scheme uses a nodal approximation on the boundary while it uses a modal approximation in the interior. The scheme we discussed here has one interior mode, the bubble function, that corresponds to the average value. This interpretation as a spectral element method also suggests a possible way of extending the active flux scheme to higher-order accuracy. Instead of simply adding nodes to the boundary, more interior modes should also be included. This avenue of research was not explored. Instead, we chose to explore spectral element methods themselves as solvers for Liouville's equation. The next chapter

therefore gives a short overview of the discontinuous Galerkin spectral element method.

Chapter 7

The discontinuous Galerkin spectral element method

> Things just seem to work out nicer on quads.
>
> David Kopriva on DG
> ICOSAHOM 2016

The discontinuous Galerkin (DG) spectral element method (SEM) is a state-of-the-art method that provides exponential convergence for piecewise smooth solutions[1]. This incredible performance is obtained by allowing a variable order of interpolation. We give here a short overview of the nodal DG-SEM, although for a complete discussion we urge the reader to take a look at some of the excellent textbooks on the subject. Most of this discussion is taken from Kopriva [121]; Hesthaven and Warburton [122]; and Canuto et al. [123].

Let's take as an example once again the scalar hyperbolic conservation law in two dimensions,

$$\frac{\partial u}{\partial t} + \nabla \cdot \mathbf{f} = 0, \tag{7.1}$$

where we don't restrict $\mathbf{f} = (f, g)$ to depend only on u, it can depend on space and time as well. Any Galerkin method is derived from the weak form, which

[1] When I first learnt about spectral methods I thought exponential convergence sounded too good to be true.

is obtained by integrating over an element Ω with a smooth test function φ, so that

$$\int_{\Omega} \frac{\partial u}{\partial t}\varphi - \nabla\varphi \cdot \mathbf{f}\,\mathrm{d}A + \oint_{\partial\Omega} \varphi \mathbf{f}\cdot\mathbf{n}\,\mathrm{d}s = 0, \tag{7.2}$$

where $\mathbf{n}$ is the outward unit normal and as per usual the closed curve integral is traversed counter-clockwise, observing the right-hand rule. The next step is to choose a basis in which we expand the test function and the solution. An adequate choice is a set of orthogonal polynomials, for instance the Legendre polynomials. The approximation lies in the fact that a finite number of them are used, yielding a finite-dimensional space of polynomials. This formulation is more general than the finite volume formulation, which may be obtained by the special choice of $\varphi = 1$. DG methods are conservative by virtue of the fact that constants are zeroth-order polynomials.

Let's assume that we're interested in solving the PDE in two dimensions and let's restrict the shape of Ω to be a quadrilateral. There are several advantages to using quadrilaterals instead of triangles, for some comparisons, see for instance Pasquetti et al. [124], or Wirasaet et al. [125]. Much like triangular domains, however, any quadrilateral can be converted to a reference square $\chi = [-1, 1]^2$. Of course, any reference square will do, but this one is particularly convenient since quadrature rules are by convention defined on the interval $[-1, 1]$.

One major advantage of quadrilaterals is that higher-dimensional basis functions can be constructed by direct products. If we denote the one-dimensional basis functions by φ_i, $i = 0, \ldots, N$, then u is approximated by

$$u(t, x, y) = \sum_{i,j=0}^{N} u_{ij}(t)\varphi_i(x)\varphi_j(y), \tag{7.3}$$

where the u_{ij} are the expansion coefficients for the particular choice of basis. In principle, any orthonormal basis can be used and it's even possible to use a different basis for each dimension. For instance, using the Fourier basis in one direction and Legendre in another is useful for problems that have periodic boundary conditions in one direction. Here, we'll use Lagrange polynomials on a suitable set of interpolation points as our basis, resulting in what's known as nodal DG.

7.1 Bilinear mapping

As we've seen in the previous chapter, it's possible to map any triangle to a reference domain by an affine mapping. For quadrilaterals, however, an extra degree of freedom is needed, resulting in a bilinear mapping. We assume a straight-edged quadrilateral with vertices $\mathbf{x}_i$, $i = 1, 2, 3, 4$, where the vertices are numbered counter-clockwise. The starting vertex is arbitrary. The mapping is then given by

$$\begin{aligned}\mathbf{x}(\xi,\eta) =& \frac{1}{4}\big[\mathbf{x}_1(1-\xi)(1-\eta) + \mathbf{x}_2(1+\xi)(1-\eta) \\ &+ \mathbf{x}_3(1+\xi)(1+\eta) + \mathbf{x}_4(1-\xi)(1+\eta)\big].\end{aligned} \tag{7.4}$$

The coordinates $(\xi, \eta) \in [-1, 1]^2$ are simply Cartesian coordinates on the reference domain, which is a square. Like with the active flux scheme, we map one set of Cartesian coordinates to another. To be able to distinguish the sets, we'll call (x, y) the physical coordinates and (ξ, η) the standard coordinates. A sketch of this mapping is shown in Figure 7.1.

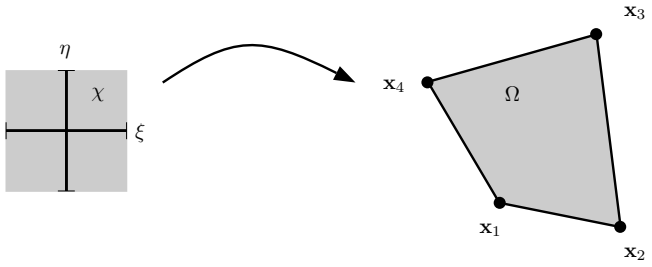


Figure 7.1: Sketch of the bilinear mapping.

The Jacobian of the transformation is easily found, let's write $\frac{\partial(x,y)}{\partial(\xi,\eta)} = (\frac{\partial \mathbf{x}}{\partial \xi}, \frac{\partial \mathbf{x}}{\partial \eta})$, where the columns are given by

$$\frac{\partial \mathbf{x}}{\partial \xi} = \tfrac{1}{4}\left[(1-\eta)\left(\mathbf{x}_2 - \mathbf{x}_1\right) + (1+\eta)\left(\mathbf{x}_3 - \mathbf{x}_4\right)\right], \tag{7.5}$$

$$\frac{\partial \mathbf{x}}{\partial \eta} = \tfrac{1}{4}\left[(1-\xi)\left(\mathbf{x}_4 - \mathbf{x}_1\right) + (1+\xi)\left(\mathbf{x}_3 - \mathbf{x}_2\right)\right]. \tag{7.6}$$

Like the linear transformation in the active flux scheme, we can find a transfor-

mation rule for the gradient by

$$\begin{pmatrix} \partial_x \\ \partial_y \end{pmatrix} = \begin{pmatrix} \xi_x & \eta_x \\ \xi_y & \eta_y \end{pmatrix} \begin{pmatrix} \partial_\xi \\ \partial_\eta \end{pmatrix}, \tag{7.7}$$

where subscripts denote partial differentiation. This is exactly the same transformation rule as (6.5). However, since the Jacobi matrix of the transformation is now not necessarily constant, the transformation of the divergence involves a bit more work. Special cases of course exist where the Jacobi matrix and its determinant are in fact constant. Using (7.7), the divergence term can be rewritten as

$$\nabla \cdot \mathbf{f} = \xi_x f_\xi + \eta_x f_\eta + \xi_y g_\xi + \eta_y g_\eta. \tag{7.8}$$

Unfortunately, the derivatives of the standard coordinates to the physical coordinates aren't easy to find. Hence, we represent the Jacobi matrix, or rather its transpose, in terms of its inverse, i.e.,

$$\begin{pmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{pmatrix} = \begin{pmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{pmatrix}^{-1} = \frac{1}{\mathcal{J}} \begin{pmatrix} y_\eta & -x_\eta \\ -y_\xi & x_\xi \end{pmatrix}, \tag{7.9}$$

where $\mathcal{J} = x_\xi y_\eta - y_\xi x_\eta$ is the Jacobian determinant. The determinant $\mathcal{J}$ will also, in general, vary throughout the element. Using now (7.9), we find that

$$\nabla \cdot \mathbf{f} = \frac{1}{\mathcal{J}} \left[y_\eta f_\xi - y_\xi f_\eta - x_\eta g_\xi + x_\xi g_\eta \right]. \tag{7.10}$$

We can now rewrite this as follows,

$$\nabla \cdot \mathbf{f} = \frac{1}{\mathcal{J}} \left[\frac{\partial}{\partial \xi} \left(y_\eta f - x_\eta g \right) + \frac{\partial}{\partial \eta} \left(x_\xi g - y_\xi f \right) \right], \tag{7.11}$$

since the cross-terms involving the second derivatives of physical coordinates cancel out. The flux function on the standard domain can be read off as

$$\tilde{\mathbf{f}} = \begin{pmatrix} y_\eta & -x_\eta \\ -y_\xi & x_\xi \end{pmatrix} \mathbf{f}, \tag{7.12}$$

with $\tilde{\mathbf{f}} = (\tilde{f}, \tilde{g})^T$. The matrix involved is known as the adjugate of the Jacobian. The divergence therefore transforms as

$$\nabla \cdot \mathbf{f} = \frac{1}{\mathcal{J}} \left(\frac{\partial \tilde{f}}{\partial \xi} + \frac{\partial \tilde{g}}{\partial \eta} \right). \tag{7.13}$$

Let's write the gradient on the reference coordinates as $\nabla_{\boldsymbol{\xi}} = (\partial_\xi, \partial_\eta)^T$, then the conservation law in the reference domain becomes

$$\frac{\partial u}{\partial t} + \frac{1}{\mathcal{J}} \nabla_{\boldsymbol{\xi}} \cdot \tilde{\mathbf{f}} = 0. \tag{7.14}$$

It should be noted that it's also possible to map to curved quadrilateral domains, where each edge of the element is allowed to be an arbitrary smooth curve. Such mappings are called transfinite [126]. We'll see an example of this in Chapter 13.

7.2 Quadrature rules

To progress, we need to convert (7.2) to something that we can actually compute[2]. We therefore have to use a quadrature rule to approximate the integrals. A quadrature rule on $[-1, 1]$ for some sufficiently smooth function g is given by

$$\int_{-1}^{1} g(\xi) \, \mathrm{d}\xi \approx \sum_{i=0}^{N} g(\xi_i) w_i, \tag{7.15}$$

with $-1 \leq \xi_i \leq 1$ being the nodes and $w_i > 0$ being the weights. Sometimes, the nodes and weights are defined starting at index $i = 1$. However, by defining the nodes starting at index $i = 0$, interpolation polynomials on the nodes have degree N. Quadrature rules on the reference square have a dyadic structure, i.e., the nodes are simply the points (ξ_i, η_j) for all $i, j = 0, \ldots, N$, while the weight of node (ξ_i, η_j) is the product $w_i w_j$. The two-dimensional analogue of (7.15) therefore becomes

$$\int_{-1}^{1} \int_{-1}^{1} g(\xi, \eta) \, \mathrm{d}\xi \, \mathrm{d}\eta = \sum_{j=0}^{N} \sum_{i=0}^{N} g(\xi_i, \eta_j) w_i w_j. \tag{7.16}$$

One way to interpret this rule is to first define an auxiliary function $G(\xi) = \sum_{j=0}^{N} g(\xi, \eta_j) w_j$, which is simply the quadrature rule applied in the η-direction for any ξ. Consequently, the quadrature rule is then applied to G in the ξ-direction, leading to the product structure.

There are two especially interesting quadrature rules that we may consider: the Gauß and the Gauß-Lobatto rules. The computation of the nodes and

[2]To quote Alan Turing [127]: *the "computable" numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means.*

weights is a standard problem in numerical theory. They're related to the roots of orthogonal polynomials and as such there are efficient off-the-shelf algorithms to compute them. Such algorithms can be found in any number of textbooks on numerical analysis or spectral methods; see e.g. Gautschi [40] or any of the aforementioned textbooks on spectral methods.

Here, we choose Gauß quadrature, since it doesn't have nodes on the edges, something that turns out to be an advantage in optics. The essential difference is that with Gauß-Lobatto quadrature, the endpoints of the interval are also included. In one dimension Gauß quadrature gives exact integration for all polynomials of degree at least $2N + 1$. This remains true in a square domain for bivariate polynomials of degree $2N + 1$. As a reminder: the degree of the bivariate polynomial $x^n y^m$ is $n + m$ and the degree of any bivariate polynomial is the highest degree of its terms [128].

7.3 Interpolation

Polynomial interpolation has the reputation of being unstable when increasing the number of nodes, though this is only justified on uniformly distributed nodes. The quality of interpolation on a set of nodes is typically measured by the Lebesgue constant Λ_N, which is related to the best possible polynomial fit, i.e.,

$$\|I_N g - g\|_\infty \leq (1 + \Lambda_N) \inf_{p \in \mathrm{poly}(N)} \|p - g\|_\infty, \tag{7.17}$$

where I_N is the interpolation operator and $\mathrm{poly}(N)$ is the space of polynomials of maximal degree N. For a uniform set of interpolation points, the Lebesgue constant blows up exponentially in N, which is illustrated by Runge's phenomenon. For Gauß nodes, the Lebesgue constant scales as $\mathcal{O}\left(\sqrt{N}\right)$ [129]. Unlike a set of uniform interpolation points, the Gauß nodes provide stable interpolation, see Figure 7.2.

Polynomial interpolation is particularly easy in Lagrange form. The Lagrange polynomials for a point set $\{\xi_i\}_{i=0}^N$ are defined by

$$\ell_i(\xi) = \prod_{\substack{j=0 \\ j \neq i}}^{N} \frac{\xi - \xi_j}{\xi_i - \xi_j}, \tag{7.18}$$

which is indeed a degree N polynomial. Lagrange polynomials satisfy the Kronecker property, i.e.,

$$\ell_i(\xi_j) = \delta_{ij}, \tag{7.19}$$

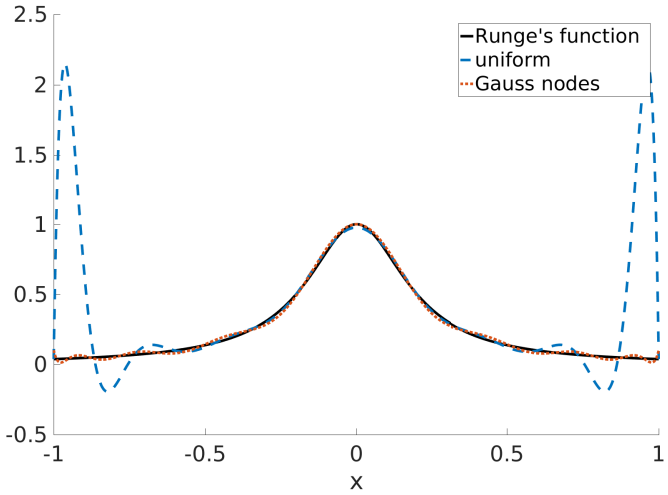


Figure 7.2: Runge's phenomenon: the error blows up for polynomial interpolation on a uniform grid of the function $\frac{1}{1+25x^2}$. For Gauß nodes, the interpolation converges for $N \to \infty$. Both interpolations were computed using 16 nodes.

where δ_{ij} is the Kronecker delta, so that $\delta_{ij} = 0$ if $i \neq j$ and $\delta_{ii} = 1$. Moreover, the Lagrange polynomials defined on Gauß nodes are orthogonal with respect to the standard L^2-inner product. This follows directly from the fact that the quadrature is exact for any polynomial up to degree $2N + 1$. In particular, the product of any two Lagrange polynomials is a polynomial of degree $2N$, yielding exact integration and therefore

$$\int_{-1}^{1} \ell_i(\xi)\ell_j(\xi)\,\mathrm{d}\xi = \sum_{n=0}^{N} \ell_i(\xi_n)\ell_j(\xi_n)w_n = w_i\delta_{ij}. \tag{7.20}$$

Thanks to the Kronecker property (7.19), Lagrange polynomials make interpolation rather easy, indeed any function $p(\xi)$ can be interpolated as

$$p_N(\xi) = (I_N p)(\xi) = \sum_{j=0}^{N} \ell_j(\xi)p(\xi_j). \tag{7.21}$$

The approximation of the derivative is simply defined as the derivative of the

polynomial interpolant, i.e.,

$$p'_N(\xi) = \sum_{j=0}^{N} \ell'_j(\xi) p(\xi_j). \tag{7.22}$$

Note that if p happens to be a polynomial of degree N or less, this differentiation is exact. In general, of course, interpolation and differentiation do not commute, resulting in an aliasing error. Aliasing, just as in Fourier theory, means the correct basis function cannot be identified uniquely. Functions that vary too rapidly may be misidentified as slowly varying instead. It will be useful to know the approximated derivative on the nodes themselves, i.e.,

$$p'_N(\xi_i) = \sum_{j=0}^{N} \ell'_j(\xi_i) p(\xi_j) = \sum_{j=0}^{N} D_{ij} p(\xi_j), \tag{7.23}$$

with $D_{ij} = \ell'_j(\xi_i)$. Differentiation of (7.18) gives

$$D_{ij} = \ell'_j(\xi_i) = \sum_{\substack{l=0 \\ l \neq j}}^{N} \frac{1}{\xi_j - \xi_l} \prod_{\substack{k=0 \\ k \neq l, k \neq j}}^{N} \frac{\xi_i - \xi_k}{\xi_j - \xi_k}, \tag{7.24}$$

for $i \neq j$. To find the diagonal elements of the differentiation matrix, we observe that any constant function must have a vanishing derivative, which implies

$$D_{ii} = -\sum_{\substack{j=0 \\ j \neq i}}^{N} D_{ij}. \tag{7.25}$$

This is sometimes referred to as the negative sum trick [121]. Efficient algorithms to compute the D_{ij} and analogous higher derivatives can be found in any of the aforementioned textbooks.

7.4 Putting it all together

The approximation consists of expanding the solution in terms of basis functions while switching to quadrature rules in (7.2). Using the Lagrange basis, this type of DG is called nodal, referring to the coefficients u_{ij} directly representing values of the numerical solution on the nodes. If we were to use, for instance, the

Legendre polynomials as basis functions for the numerical solution, the resulting scheme would be called modal DG. The coefficients u_{ij} wouldn't represent any point value of the approximate solution, but rather coordinates of the orthogonal decomposition. By choosing the test function to be equal to each of the basis functions, we can obtain an ODE for every expansion coefficient $u_{ij}(t)$. We start with the weak form of the PDE transformed to the reference domain, i.e.,

$$\int_\chi \frac{\partial u}{\partial t}\varphi \mathcal{J}\,\mathrm{d}A_{\boldsymbol{\xi}} = \int_\chi \nabla_{\boldsymbol{\xi}}\varphi \cdot \tilde{\mathbf{f}}\,\mathrm{d}A_{\boldsymbol{\xi}} - \oint_{\partial\chi} \varphi \tilde{\mathbf{f}}\cdot\mathbf{n}\,\mathrm{d}\sigma, \tag{7.26}$$

where σ is the arc length in the reference domain. As is customary, $\mathbf{n}$ is the outward pointing normal and the orientation of the closed curve is counter-clockwise, fulfilling yet again the right-hand rule. Next, we set $\varphi_{ij}(\xi,\eta) = \ell_i(\xi)\ell_j(\eta)$, so that the first term in (7.26) yields

$$\int_\chi \frac{\partial u}{\partial t}\varphi_{ij}\mathcal{J}\,\mathrm{d}V_{\boldsymbol{\xi}} \approx \sum_{n,m=0}^{N}\left(\sum_{k,l=0}^{N} u'_{kl}(t)\ell_k(\xi_n)\ell_l(\eta_m)\right)\ell_i(\xi_n)\ell_j(\eta_m)\mathcal{J}_{nm}w_n w_m, \tag{7.27}$$

with $\mathcal{J}_{nm} = \mathcal{J}(\xi_n,\eta_m)$. The term in parenthesis is the approximation of $\frac{\partial u}{\partial t}$. Using the Kronecker property (7.19) of the Lagrange polynomials four times, the sums simplify all the way down to

$$\int_\chi \frac{\partial u}{\partial t}\varphi_{ij}\mathcal{J}\,\mathrm{d}V_{\boldsymbol{\xi}} \approx u'_{ij}(t)\mathcal{J}_{ij}w_i w_j. \tag{7.28}$$

The next term to be approximated is the area term on the right-hand side of (7.26). Let's write $\tilde{\mathbf{f}} = (\tilde{f},\tilde{g})^T$, the area integral is then approximated by

$$\int_\chi \nabla_{\boldsymbol{\xi}}\varphi_{ij}\cdot\tilde{\mathbf{f}}\,\mathrm{d}V_{\boldsymbol{\xi}} \approx \sum_{n,m=0}^{N} w_n w_m\left(\ell'_i(\xi_n)\ell_j(\eta_m)\tilde{f}_{nm} + \ell_i(\xi_n)\ell'_j(\eta_m)\tilde{g}_{nm}\right), \tag{7.29}$$

where we've already used the fact that due to the choice of test function, we find $\nabla_{\boldsymbol{\xi}}\varphi_{ij}(\xi,\eta) = \left(\ell'_i(\xi)\ell_j(\eta), \ell_i(\xi)\ell'_j(\eta)\right)^T$. We can identify the differentiation matrices inside the sums, i.e., $D^{(\xi)}_{ni} = \ell'_i(\xi_n)$ and $D^{(\eta)}_{mj} = \ell'_j(\eta_m)$. We can yet again employ the Kronecker property (7.19) to simplify (7.29), yielding

$$\int_\chi \nabla_{\boldsymbol{\xi}}\varphi\cdot\tilde{\mathbf{f}}\,\mathrm{d}\xi \approx w_j\sum_{n=0}^{N} w_n D^{(\xi)}_{ni}\tilde{f}_{nj} + w_i\sum_{m=0}^{N} w_m D^{(\eta)}_{mj}\tilde{g}_{im}. \tag{7.30}$$

Finally, the boundary terms are approximated in a similar vein with one difference. Whereas inside an element, the numerical solution is continuous, on element boundaries it's allowed to be discontinuous. Thus, the limit toward the boundary of an element can have two values, one for each element the boundary touches. As a consequence, the flux on the boundary must be replaced with a numerical flux, e.g. the Godunov flux. It's therefore through the numerical flux that neighbouring elements communicate. We'll denote $\tilde{\mathbf{F}} = (\tilde{F}, \tilde{G})^T$ the numerical flux. The numerical flux is often written as depending on the left and right limits towards element boundary, u_L and u_R respectively, so that $\tilde{\mathbf{F}} = \tilde{\mathbf{F}}(u_L, u_R)$.

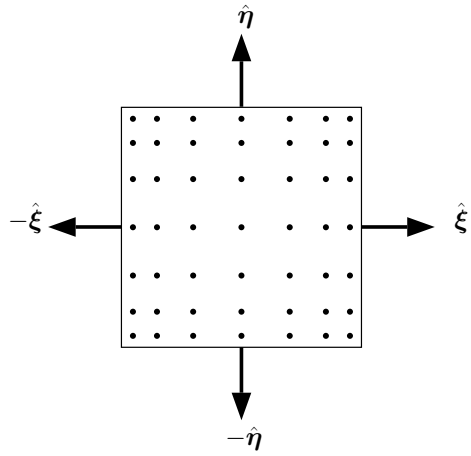


Figure 7.3: Sketch of the reference domain with $N = 6$. Normals are indicated with arrows.

A sketch of the reference domain is shown in Figure 7.3. Each boundary integral can be approximated by a one-dimensional quadrature rule. The result is added up to approximate the total boundary integral term. Using the picture as a guideline, the bottom boundary integral becomes

$$\int_{-1}^{1} \varphi_{ij} \tilde{\mathbf{f}} \cdot (-\hat{\boldsymbol{\eta}}) \, \mathrm{d}\xi \approx -\sum_{n=0}^{N} w_n \ell_i(\xi_n) \ell_j(-1) \tilde{G}(\xi_n, -1), \tag{7.31}$$

where $\hat{\boldsymbol{\eta}}$ is the unit vector pointing in the η-direction. The Kronecker property

(7.19) once again helps us out, so that we find

$$\int_{-1}^{1} \varphi_{ij} \tilde{\mathbf{f}} \cdot (-\hat{\boldsymbol{\eta}}) \, \mathrm{d}\xi \approx -w_i \ell_j(-1) \tilde{G}(\xi_i, -1). \tag{7.32}$$

The other three line integrals of the boundary are treated similarly, so that the entire boundary integral becomes

$$\begin{aligned} \oint_{\partial\chi} \varphi_{ij} \tilde{\mathbf{f}} \cdot \mathbf{n} \, \mathrm{d}\sigma \approx{}& w_j \ell_i(1) \tilde{F}(1, \eta_j) - w_j \ell_i(-1) \tilde{F}(-1, \eta_j) \\ &+ w_i \ell_j(1) \tilde{G}(\xi_i, 1) - w_i \ell_j(-1) \tilde{G}(\xi_i, -1). \end{aligned} \tag{7.33}$$

To finish up, the approximation to the weak form (7.26) with $\varphi = \varphi_{ij}$ is given by the sum of its constituent terms, that is (7.28), (7.30) and (7.33). We divide by $w_i w_j \mathcal{J}_{ij}$ to obtain an ODE for u_{ij}, which results in a large set of coupled ODEs:

$$\begin{aligned} \frac{\mathrm{d}u_{ij}}{\mathrm{d}t} + \frac{1}{\mathcal{J}_{ij}} \Big\{ & \tilde{F}(1, \eta_j) \frac{\ell_i(1)}{w_i} - \tilde{F}(-1, \eta_j) \frac{\ell_i(-1)}{w_i} + \sum_{n=0}^{N} \hat{D}^{(\xi)}_{in} \tilde{f}_{nj} \\ & + \tilde{G}(\xi_i, 1) \frac{\ell_j(1)}{w_j} - \tilde{G}(\xi_i, -1) \frac{\ell_j(-1)}{w_j} + \sum_{m=0}^{N} \hat{D}^{(\eta)}_{jm} \tilde{g}_{im} \Big\} \\ & = 0 \end{aligned} \tag{7.34a}$$

where we've defined the modified differentiation matrices

$$\hat{D}^{(\xi)}_{in} = -D^{(\xi)}_{ni} \frac{w_n}{w_i}, \tag{7.34b}$$

$$\hat{D}^{(\eta)}_{jm} = -D^{(\eta)}_{mj} \frac{w_m}{w_j}. \tag{7.34c}$$

7.5 Error behaviour and CFL condition

In the above derivation, we've showed the DG discretisation method for a single element. Of course, the DG spectral element method (SEM) is defined on an unstructured mesh. The individual elements communicate with each other by means of the numerical flux. Let's assume that we're working on a mesh with typical dimension h. With the DG-SEM, the exact solution is approximated by

an Nth-order polynomial, so the spatial error is of order $N+1$ with respect to the typical mesh size h, i.e.,

$$e = \mathcal{O}(h^{N+1}). \tag{7.35}$$

However, this error formula also implies exponential convergence with respect to N. Discontinuous Galerkin methods are an example of hp-methods, where h refers to mesh size and p to polynomial order. Contrast this to the previous solvers we've discussed: the upwind and WENO methods work on a uniform grid, precluding the possibility of using local refinement; the active flux scheme allows local refinement of the mesh, but the order of approximation is fixed. The DG-SEM allows both local refinement of the mesh, as well as local enrichment of the polynomial degree. It's also possible to do this on the fly. Adaptive hp-methods look for the most efficient way to decrease error by either refining the mesh or enriching the polynomial order [130–137].

The discontinuous Galerkin spectral element method as presented provides a semi-discretisation that again fits perfectly into the method of lines paradigm. However, to work as a numerical method, it's necessary to use some numerical integrator on (7.34a). Popular choices, as always, are Runge–Kutta methods [138]. Other common choices are linear multistep methods such as Adams-Bashforth methods [120, 139]. Unless stated otherwise, we'll be using the RK4 method. For a system of ODEs represented by $\mathbf{y}' = \mathbf{f}(t, \mathbf{y})$, one step of the RK4 method is given by

$$\begin{aligned}
\mathbf{k}_1 &= \mathbf{f}(t_n, \mathbf{y}_n) \\
\mathbf{k}_2 &= \mathbf{f}\left(t_n + \frac{\Delta t}{2}, \mathbf{y}_n + \frac{\Delta t}{2}\mathbf{k}_1\right) \\
\mathbf{k}_3 &= \mathbf{f}\left(t_n + \frac{\Delta t}{2}, \mathbf{y}_n + \frac{\Delta t}{2}\mathbf{k}_2\right) \\
\mathbf{k}_2 &= \mathbf{f}(t_n + \Delta t, \mathbf{y}_n + \Delta t \mathbf{k}_3) \\
\mathbf{y}_{n+1} &= \mathbf{y}_n + \frac{\Delta t}{6}\left(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4\right).
\end{aligned} \tag{7.36}$$

The RK4 method is a good all-purpose time integrator that provides fourth-order accuracy, i.e., the global error is $\mathcal{O}(\Delta t^4)$.

As with all previous discretisation methods we've seen, the DG-SEM coupled to a numerical time integrator also has to satisfy a CFL condition. As usual, we define the CFL number as $c = \frac{b_{\max}\Delta t}{\Delta x}$, where $b_{\max}$ is the maximum velocity occurring in the problem while Δx is the smallest distance in the mesh. Contrary

to the other solvers we've discussed, no tight CFL condition is known for this class of methods. For instance, the upwind scheme has a tight CFL condition, meaning it becomes unstable when $c = 1 + \varepsilon$ for any $\varepsilon > 0$, while it is stable for $c = 1$. In practice the recommended condition for DG is given by

$$|c| \leq \frac{1}{2N+1}. \tag{7.37}$$

This condition is tight for $N = 0$ and $N = 1$, while for higher polynomial orders it's typically within 5% accurate [140, 141].

7.6 Arbitrary Lagrangian-Eulerian description

Spectral element methods may be considered on a moving mesh as well, where the vertices of each element move with respect to time. The same reasons as mentioned in the previous chapter apply. For instance, a curved optic can be represented by a moving physical boundary in phase space. When changes are large and sudden, a complete remeshing may be unavoidable, where we need to rerun the meshing software and interpolate from the old mesh to the new. When the changes are relatively small and slow, such as with the smooth motion of a physical boundary, we can interpret any changes to the mesh as motion of the nodes. To find the correct numerical method, we first transform the strong form of the PDE. We consider a transformation from the static reference domain $\chi = [-1, 1]^2$ to a moving element given by $(\tau, \boldsymbol{\xi}) \to (t, \mathbf{x})$ with $t = \tau$. We'd like to transform the PDE to back the reference domain, so that we may compute the numerical solution there. We denote $u^*(\tau, \boldsymbol{\xi}) = u(t, \mathbf{x})$, so that the τ-derivative of u^* is given by

$$\frac{\partial u^*}{\partial \tau} = \frac{\partial u}{\partial t} + \mathbf{v} \cdot \nabla u, \tag{7.38}$$

where $\mathbf{v} = \frac{\partial \mathbf{x}}{\partial \tau} = \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t}$ and ∇ is the gradient with respect to $\mathbf{x}$. The transformation in time therefore reads

$$\frac{\partial u^*}{\partial \tau} + \nabla \cdot \mathbf{f} - \mathbf{v} \cdot \nabla u = 0. \tag{7.39}$$

We can draw the advection term into the divergence operator provided we add a compensating term, i.e.,

$$\frac{\partial u^*}{\partial \tau} + \nabla \cdot (\mathbf{f} - u\mathbf{v}) + \frac{u}{\mathcal{J}} \frac{\partial \mathcal{J}}{\partial \tau} = 0, \tag{7.40}$$

where we've used the identity $\frac{\partial \mathcal{J}}{\partial \tau} = \mathcal{J} \nabla \cdot \mathbf{v}$, which also appears as Lemma 1.1 in Chapter 1. Next, we transform in space, yielding

$$\nabla \cdot (\mathbf{f} - u\mathbf{v}) = \frac{1}{\mathcal{J}} \nabla_{\boldsymbol{\xi}} \cdot \tilde{\mathbf{f}}, \tag{7.41}$$

where we've used $\nabla_{\boldsymbol{\xi}} = (\partial_\xi, \partial_\eta)^T$ just as before in (7.14). The transformed flux, which can be compared to (7.12), is now defined as

$$\tilde{\mathbf{f}} = \begin{pmatrix} y_\eta & -x_\eta \\ -y_\xi & x_\xi \end{pmatrix} (\mathbf{f} - u\mathbf{v}), \tag{7.42}$$

where once again the adjugate of the spatial Jacobian of the transformation appears. The PDE in the reference coordinates therefore satisfies

$$\frac{\partial u^*}{\partial \tau} + \frac{1}{\mathcal{J}} \left(\nabla_{\boldsymbol{\xi}} \cdot \tilde{\mathbf{f}} + u^* \frac{\partial \mathcal{J}}{\partial \tau} \right) = 0. \tag{7.43}$$

Furthermore, we define the transformed velocity field

$$\tilde{\mathbf{v}} = \begin{pmatrix} y_\eta & -x_\eta \\ -y_\xi & x_\xi \end{pmatrix} \mathbf{v}, \tag{7.44}$$

so that multiplying (7.43) by the $\mathcal{J}$ allows us to find a one-way coupled system of PDEs, i.e.,

$$\frac{\partial}{\partial \tau}(u\mathcal{J}) + \nabla_{\boldsymbol{\xi}} \cdot \tilde{\mathbf{f}} = 0. \tag{7.45a}$$

$$\frac{\partial \mathcal{J}}{\partial \tau} - \nabla_{\boldsymbol{\xi}} \cdot \tilde{\mathbf{v}} = 0, \tag{7.45b}$$

where we've dropped the asterisk for brevity. The second PDE is the identity from Lemma 1.1 transformed to the reference domain. The spatial part of this system is discretised with the DG method as outlined above. Acosta Minoli and Kopriva proved that integrating the semi-discrete system with a general linear method, e.g. an RK method or LMM, is free-stream preserving [120]. The result can be interpreted similarly to Theorem 6.1 for the active flux scheme: rather than the exact determinant, we need to use the numerical determinant to recover the solution u.

7.7 Liouville and DG-SEM

Like the active flux scheme, the DG-SEM is an excellent method for Liouville's equation. The reasons are more or less the same: it's a conservative method that's locally defined on elements. As we've also argued in Chapter 4, a symplectic method in the Lagrangian frame is more or less comparable to conservative methods in the Eulerian frame. The conservative nature of the DG method is therefore a requirement for its usefulness to Liouville's equation. Furthermore, higher orders are achieved by increasing the number of internal degrees of freedom in each element, i.e., by using more interpolation nodes. Thus, achieving high-order accuracy in optics problems is again a matter of aligning the mesh properly with optical interfaces.

The DG-SEM has some advantages over the active flux scheme as well, mainly its extraordinary flexibility. As we've briefly touched upon in Section 7.5, the DG-SEM is an *hp*-method, which means both mesh size and polynomial degree can be adjusted to achieve some desired accuracy. Moreover, the discontinuous Gelerkin method can in principle be defined on elements of any shape, e.g. triangular elements [122]. Another advantage is that it is, at least in theory, quite easy to extend to higher dimensions. Every extra dimension just requires another multiplication with a basis function in that direction. We won't be stretching the flexibility of the DG-SEM too far, but with an eye on future application and development, these advantages could turn out to be crucial.

Part III

Numerical methods for Liouville's equation

This part discusses the application of the schemes presented in Part 2 to Liouville's equation. We restrict our attention to two-dimensional optics, hence Liouville's equation in conservative form is given by

$$\frac{\partial \rho}{\partial z} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{7.46}$$

with $\mathbf{u} = \left(\frac{\partial h}{\partial p}, -\frac{\partial h}{\partial q}\right)^T$. The schemes are presented in the chronological order in which they were developed, which also happens to be in order of increasing complexity of the original schemes.

Chapter 8

An upwind Liouville solver

> Ma démonstration, quoique très simple, serait sans doute appréciée par les géomètres qui ... savent combien en général elles offrent de difficultés.
>
> Joseph Liouville

Recall that in Chapter 4, we concluded using scaling arguments that solving Liouville's equation should lead to faster algorithms for illumination optics problems. Moreover, we argued that the accuracy should be at least as good or better. We didn't discuss any of the difficulties that might arise in attempting to solve Liouville's equation when optical interfaces are present.

In terms of Liouville's equation, an optical interface represents a sudden transition to a different advection speed. Interfaces are therefore in some sense comparable to shocks in, for instance, Euler's equations. However, different from a shock is that the position of the interface is given to us and doesn't depend dynamically on the solution. The hard part is that the effect of the interface is nonlocal. For finite difference methods, only the closest neighbours of a grid point contribute to its evolution, even when shocks are present. When dealing with optical interfaces, completely different parts of phase space are in contact with each other. This means we need to be very careful with how we apply the finite difference across the interface. Our main theoretical result is to derive a jump condition on the solution to Liouville's equation in Theorem 8.1.

This will be used to construct the upwind Liouville solver, which is detailed in Theorem 8.2.

We'll now apply the upwind scheme to Liouville's equation in advection form[1]. We restrict ourselves to the two-dimensional case, so that we aim to numerically solve

$$\frac{\partial \rho}{\partial z} + \frac{\partial h}{\partial p}\frac{\partial \rho}{\partial q} - \frac{\partial h}{\partial q}\frac{\partial \rho}{\partial p} = 0. \tag{8.1}$$

Whenever h is sufficiently smooth, we can apply the off-the-shelf upwind scheme. Therefore, only close to an interface do we need to alter the scheme. To find the derivatives at a grid point, we need to relate the gradients on both sides of the interface. The basic strategy is to perform a Taylor expansion from the interface on both sides.

When traversing an interface where the refractive index changes discontinuously from n_1 to n_2, rays obey Snell's law, given in two dimensions by

$$\mathcal{S}\left(p; n_1, n_2, \nu\right) := \begin{cases} p - \left(\psi + \operatorname{sgn}\left(n_2\right)\sqrt{\delta}\right)\nu & \text{if } \delta \geq 0, \\ p - 2\psi\nu & \text{if } \delta < 0, \end{cases} \tag{8.2a}$$

where

$$\delta := n_2^2 - n_1^2 + \psi^2 \quad \text{and} \quad \psi := p\nu \pm \sqrt{n_1^2 - p^2}\sqrt{1-\nu^2}, \tag{8.2b}$$

where the sign is to be taken such that $\psi \leq 0$, which follows from the angle convention of Snell's law, see Section 1.3. This is the two-dimensional form, so that one component of the normal provides sufficient information, hence $\nu \in \mathbb{R}$ such that $|\nu| \leq 1$.

8.1 Jump condition on the gradient

Near interfaces, h is discontinuous and therefore a classical solution to (8.1) doesn't exist. In Chapter 3 we argued that physical solutions are characterised by being constant along rays, so that at an interface given by $q = q^\star$, we should use

$$\rho(z^+, q^+, p^+) = \rho(z^-, q^-, p^-), \tag{8.3}$$

where a plus or minus denotes a one-sided limit towards the interface, so that $z^+ = z^-$, $q^+ = q^- = q^\star$ and $p^+ = \mathcal{S}(p^-)$ with $\mathcal{S}$ being the explicit form of

[1] Ironically, I found out later that this way of getting the upwind scheme to work for optics problems is much harder than for active flux schemes or spectral element methods.

Snell's law as defined by (8.2). If there are several characteristics with each a different brightness, then any linear combination of those values of ρ will also be a constant. Hence, we can advance as follows to derive the jump condition we need.

First, consider several points in phase space close to each other on the z^--side of the interface. In a two-dimensional setting, we can do with three suitably chosen points to approximate the derivatives in the q- and p-direction by finite differences, see the arrangement in Figure 8.1. Next, move along characteristics that go through these points by a small amount Δz such that all the points cross the interface. By applying (8.3), the corresponding finite differences will also be constant in z. Taking limits allows us to relate gradients on both sides of the interface. This reasoning is simplified greatly by using a flat surface. Moreover, since Snell's law only depends on the local gradient, the treatment of a curved surface can be derived from that of a flat surface through a suitable coordinate transform.

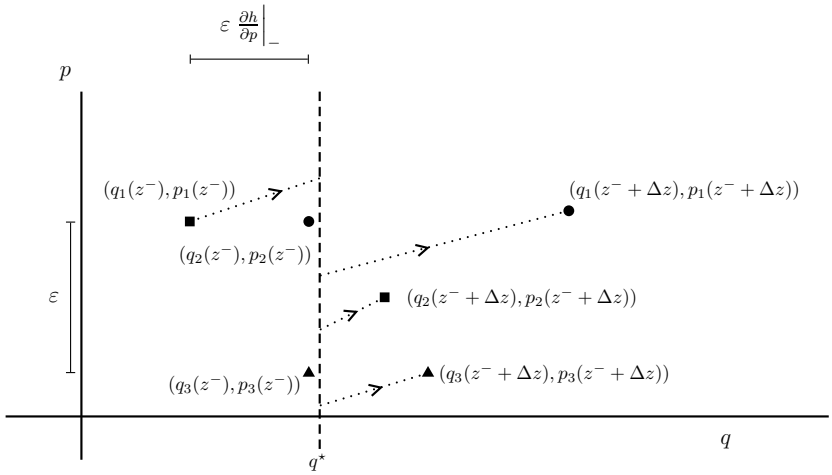


Figure 8.1: Basic idea for finding the jump condition on the gradient of ρ. The interface is represented by the dashed line, the rays projected onto phase space are represented by dotted lines.

Theorem 8.1. *Let $h : \mathcal{P} \to \mathbb{R}$ be piecewise smooth, let the interface be flat and*

the initial condition differentiable, i.e., $\rho_0 \in C^1(\mathcal{P})$. *Then* (8.3) *implies*

$$\left(\frac{\partial h}{\partial p}\frac{\partial \rho}{\partial q} - \frac{\partial h}{\partial q}\frac{\partial \rho}{\partial p}\right)\Big|_- = \left(\frac{\partial h}{\partial p}\frac{\partial \rho}{\partial q} - \frac{\partial h}{\partial q}\frac{\partial \rho}{\partial p}\right)\Big|_+, \tag{8.4}$$

where $\cdot|_\pm$ *is shorthand for evaluation at* $(z^\pm, q(z^\pm), p(z^\pm))$. *Furthermore, Snell's law implies*

$$\frac{\partial \rho}{\partial p}\Big|_- = \frac{\partial \mathcal{S}}{\partial p}\Big|_- \frac{\partial \rho}{\partial p}\Big|_+. \tag{8.5}$$

Proof. 1. Let $z \mapsto (q_1(z), p_1(z))$ be the parametrisation of a base characteristic that intersects the interface at $z^\star$, i.e. $q(z^\star) = q^\star$. Let's denote one-sided limits towards $z^\star$ with a superscript plus or minus. Thus,

$$p^- := \lim_{z \uparrow z^\star} p_1(z), \quad p^+ := \lim_{z \downarrow z^\star} p_1(z),$$

and similarly for $q^\pm$ and $z^\pm$, see Figure 8.1. We use as an initial condition for this characteristic

$$q_1(z^-) = q^-,$$
$$p_1(z^-) = p^-.$$

Let $z \mapsto (q_2(z), p_2(z))$ be the parametrisation of the second base characteristic. Both base characteristics should intersect the surface at the same point $q^\star$, but with slightly different momenta and at slightly different z, i.e., it should satisfy $q_2(z^- + \varepsilon) = q^-$ to second order. Therefore, let $\varepsilon > 0$ be some small number, then the second characteristic has initial conditions

$$q_2(z^-) = q^- - \varepsilon \frac{\partial h}{\partial p}\Big|_- + \tfrac{1}{2}\varepsilon^2 \left(\frac{1}{h}\frac{\partial h}{\partial q} + \frac{p^-}{h}\frac{\partial h}{\partial z}\right)\Big|_-,$$
$$p_2(z^-) = p^-,$$

where we use $\cdot|_-$ as shorthand for evaluation at (z^-, q^-, p^-). In Figure 8.1, (q_1, p_1) is marked by a bullet, while (q_2, p_2) is marked by a square. Next, let's define the values of ρ for these two characteristics as

$$\rho_1^\star(z) := \rho\left(z, q_1(z), p_1(z)\right),$$
$$\rho_2^\star(z) := \rho\left(z, q_2(z), p_2(z)\right),$$

which are both in fact constants. Note that due to these definitions, after a Taylor series expansion of $\rho_1^\star$ and $\rho_2^\star$ around z^- in ε, we have

$$\frac{\rho_1^\star(z^-) - \rho_2^\star(z^-)}{\varepsilon} = \frac{\partial h}{\partial p}\frac{\partial \rho}{\partial q}\Big|_- + \mathcal{O}(\varepsilon), \tag{$*$}$$

which is also a constant.

2. The evolution of the second characteristic is sketched in Figure 8.2. Roughly speaking, during the first ε of propagation, it reaches the interface and changes discontinuously according to Snell's law. During the second ε of propagation, it travels through the second medium.

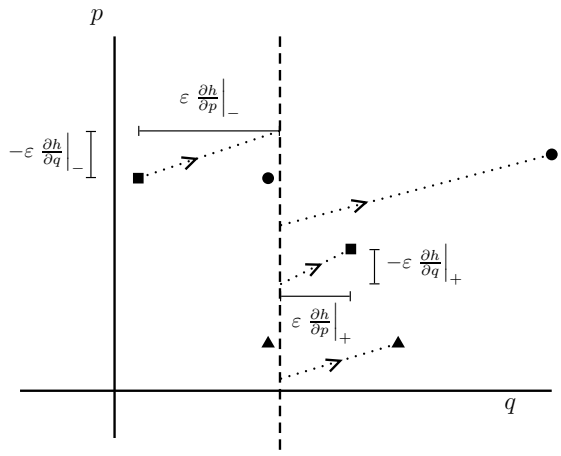


Figure 8.2: Evolution of the second characteristic with distances indicated. The base characteristic (q_1, p_1) is indicated with a bullet, (q_2, p_2) is indicated with a square and (q_3, p_3) is indicated with a triangle.

According to (8.3), $\rho_1^\star$ and $\rho_2^\star$ remain constant when encountering an interface. The characteristics change discontinuously, but the value of $\rho_1^\star$ and $\rho_2^\star$ will not change. Thus, we see that $(*)$ is also true across an interface. Therefore, we can advance z by some small amount Δz such that both characteristics have crossed the interface. Choosing $\Delta z = 2\varepsilon$, we end up with

$$q_1(z^+ + \Delta z) = q^+ + 2\varepsilon \left.\frac{\partial h}{\partial p}\right|_+ + \mathcal{O}(\varepsilon^2),$$

for the first base characteristic. For the second base characteristic, we first propagate by an amount ε in z, so that $q_2(z^- + \varepsilon) = q^- + \mathcal{O}(\varepsilon^2)$, then we apply Snell's law and propagate another ε, resulting in

$$q_2(z^+ + \Delta z) = q^+ + \varepsilon \left.\frac{\partial h}{\partial p}\right|_+ + \mathcal{O}(\varepsilon^2).$$

In a geometrical optics setting, q is continuous and therefore $q^- = q^+$. Denoting $p^+ = \mathcal{S}(p^-; n(q^-), n(q^+), \nu)$, we obtain for the momenta

$$p_1(z^+ + \Delta z) = p^+ - 2\varepsilon \left.\frac{\partial h}{\partial q}\right|_+ + \mathcal{O}(\varepsilon^2),$$
$$p_2(z^+ + \Delta z) = \mathcal{S}\left(p^- - \varepsilon \left.\frac{\partial h}{\partial q}\right|_- ; n(q^-), n(q^+), \nu\right) - \varepsilon \left.\frac{\partial h}{\partial q}\right|_+ + \mathcal{O}(\varepsilon^2).$$

A Taylor expansion of Snell's function yields

$$p_2(z^+ + \Delta z) = p^+ - \varepsilon \left.\frac{\partial \mathcal{S}}{\partial p}\right|_- \left.\frac{\partial h}{\partial q}\right|_- - \varepsilon \left.\frac{\partial h}{\partial q}\right|_+ + \mathcal{O}(\varepsilon^2),$$

where $\mathcal{S}(p^-) = p^+$ and $\left.\frac{\partial \mathcal{S}}{\partial p}\right|_-$ is the derivative of Snell's law. We take again the finite difference similar to $(*)$, but now evaluated at $z + \Delta z$, giving

$$\frac{\rho_1^\star(z^+ + \Delta z) - \rho_2^\star(z^+ + \Delta z)}{\varepsilon} =$$
$$\left.\frac{\partial h}{\partial p}\right|_+ \cdot \left.\frac{\partial \rho}{\partial q}\right|_+ + \left(\left.\frac{\partial \mathcal{S}}{\partial p}\right|_- \left.\frac{\partial h}{\partial q}\right|_- - \left.\frac{\partial h}{\partial q}\right|_+\right) \cdot \left.\frac{\partial \rho}{\partial p}\right|_+ + \mathcal{O}(\varepsilon).$$

However, since $\rho_1^\star$ and $\rho_2^\star$ are both constant, it follows that this must be equal to $(*)$, yielding

$$\left.\frac{\partial h}{\partial p} \cdot \frac{\partial \rho}{\partial q}\right|_- = \left.\frac{\partial h}{\partial p} \cdot \frac{\partial \rho}{\partial q}\right|_+ + \left(\left.\frac{\partial \mathcal{S}}{\partial p}\right|_- \left.\frac{\partial h}{\partial q}\right|_- - \left.\frac{\partial h}{\partial q}\right|_+\right) \cdot \left.\frac{\partial \rho}{\partial p}\right|_+ + \mathcal{O}(\varepsilon), \qquad (**)$$

where taking the limit of $\varepsilon \to 0$ removes the $\mathcal{O}(\varepsilon)$ terms.

3. Next, we consider a third characteristic that passes through the same intersection point q^- and at the same z-coordinate z^-, but with a slightly different momentum. This third base characteristic is marked with a triangle in Figure 8.2. Hence, the third characteristic should have as an initial condition,

$$q_3(z^-) = q^-,$$
$$p_3(z^-) = p^- - \varepsilon,$$

with $|p^- - \varepsilon| \leq n(q^-)$, which ensures that the momentum p_3 is physical. We also define the value of ρ along the third characteristic,

$$\rho_3^\star(z) := \rho\left(z, q_3(z), p_3(z)\right),$$

where we once again point out that $\rho_3^\star$ is in fact a constant. Furthermore, analogous to $(*)$, we have

$$\frac{\rho_1^\star(z^-) - \rho_3^\star(z^-)}{\varepsilon} = \left.\frac{\partial \rho}{\partial p}\right|_- + \mathcal{O}(\varepsilon). \tag{\#}$$

On the other hand, we can advance z by an arbitrarily small amount $\Delta z > 0$. For simplicity, we'll again choose $\Delta z = 2\varepsilon$ and, after similar operations as earlier, we obtain

$$\frac{\rho_1^\star(z+\Delta z) - \rho_3^\star(z+\Delta z)}{\varepsilon} = \left.\frac{\partial \mathcal{S}}{\partial p}\right|_- \left.\frac{\partial \rho}{\partial p}\right|_+ + \mathcal{O}(\varepsilon),$$

However, since $\rho_1^\star$ and $\rho_3^\star$ are constant, we find that this expression must be equal to (#), yielding

$$\left(\left.\frac{\partial \rho}{\partial p}\right|_- - \left.\frac{\partial \mathcal{S}}{\partial p}\right|_- \left.\frac{\partial \rho}{\partial p}\right|_+ \right) = \mathcal{O}(\varepsilon).$$

Furthermore, this equality must hold for all admissible ε, and letting $\varepsilon \to 0$ yields (8.5).

4. We can rewrite $(**)$ into the form

$$\left.\frac{\partial h}{\partial p}\frac{\partial \rho}{\partial q}\right|_- - \left.\frac{\partial h}{\partial q}\right|_- \left(\left.\frac{\partial \mathcal{S}}{\partial p}\right|_- \left.\frac{\partial \rho}{\partial p}\right|_+ \right) = \left.\frac{\partial h}{\partial p}\frac{\partial \rho}{\partial q}\right|_+ - \left.\frac{\partial h}{\partial q}\frac{\partial \rho}{\partial p}\right|_+,$$

where applying (8.5) gives (8.4). □

Remark. *Using the definition of the* Poisson bracket*,*

$$\{f, g\} := \frac{\partial f}{\partial q}\frac{\partial g}{\partial p} - \frac{\partial f}{\partial p}\frac{\partial g}{\partial q}, \tag{8.6}$$

we can rewrite the jump condition (8.4) *into the concise form,*

$$\{\rho, h\}_- = \{\rho, h\}_+ . \tag{8.7}$$

Corollary 8.1. *When the interface is curved, we must adjust the result of Theorem 8.1 as follows,*

$$\left[\frac{\partial \rho}{\partial q} \cdot \left(\frac{\partial h}{\partial p} - Q'(z^\star)\right) - \frac{\partial h}{\partial q} \cdot \frac{\partial \rho}{\partial p}\right]\Bigg|_- = \left[\frac{\partial \rho}{\partial q} \cdot \left(\frac{\partial h}{\partial p} - Q'(z^\star)\right) - \frac{\partial h}{\partial q} \cdot \frac{\partial \rho}{\partial p}\right]\Bigg|_+, \tag{8.8}$$

where $Q : \mathbb{R}^+ \to Q$ is differentiable and $q = Q(z^\star)$ gives the location of the intersection point of a characteristic with the surface. Furthermore, (8.5) *is also valid.*

Proof. A curved interface is represented by a plane in phase space moving with velocity $Q'(z)$. Hence, we perform a coordinate transform affecting the q-direction only, defined as

$$\tilde{q} = q - Q(z),$$

which results in a coordinate system where the surface is standing still. This transformation may be absorbed into the Hamiltonian by defining a new one, given by

$$\tilde{h}(z, q, p) = h(z, q, p) - Q'(z)p. \tag{8.9}$$

Applying Theorem 8.1 to this new system yields (8.8). □

It's important to note that Theorem 8.1 remains true in a very general setting. It can easily be generalised to 3D optics, while its formulation is independent of the particular form of the Hamiltonian, coordinate system or evolution coordinate. We can, for example, find a Hamiltonian system describing geometrical optics in terms of real time or arc length.

The jump condition (8.4) is a consequence of the fact that the underlying ray transformation from z^- to z^+ is symplectic. Snell's law, properly interpreted, constitutes a symplectic transformation [16, 36]. The influence of an interface is reflected in the gradient of ρ according to Theorem 8.1. When the action of an interface is such that rays are moved closer together in the q-direction, they must get further apart in the p-direction. If, for instance, the distance in the q-coordinates between two reference points is squeezed by a factor of two over the interface, the gradient in the q-direction will also become steeper by the same factor. This follows directly from the the fact that ρ is constant along rays. However, these rays will also have to be a moved apart in the p-direction due to conservation of étendue, so that at the same time the gradient in the p-direction is shallower by a factor of two. This is, in effect, exactly what the theorem asserts.

8.2 Derivation of the scheme

Let's introduce some shorthand notation for the advection speeds on the screen, given by

$$a(z,q,p) := \frac{\partial h}{\partial p} = \frac{p}{\sqrt{n(z,q)^2 - p^2}}, \tag{8.10a}$$

$$b(z,q,p) := -\frac{\partial h}{\partial q} = \frac{n(z,q)}{\sqrt{n(z,q)^2 - p^2}} \frac{\partial n}{\partial q}. \tag{8.10b}$$

We look for approximate solutions to

$$\frac{\partial \rho}{\partial z} + a \frac{\partial \rho}{\partial q} + b \frac{\partial \rho}{\partial p} = 0, \tag{8.11}$$

under the additional condition that whenever n changes discontinuously, we use (8.3) with p^- and p^+ related through Snell's law (8.2) and $q^- = q^+$, $z^- = z^+$.

We apply a grid on phase space such that we have $\{q_i\}_{i=1}^N$ for the positions and $\{p_j\}_{j=1}^M$ for the momenta. We rescale the position space such that $q \in [0,1]$, giving

$$q_i := (i-1)\Delta q, \quad \Delta q := \frac{1}{N-1}, \tag{8.12}$$

where $i \in \{1,\dots,N\}$. Note that the advection speeds, a and b, go to infinity for p close to $n(q,z)$. Therefore, in many cases it's practical to choose a maximum allowed momentum in the system, $p_{\max}$. This corresponds roughly to setting a maximum angular aperture. The discretisation of p is thus defined as

$$p_j := \tfrac{1}{2}(2j - M - 1)\Delta p, \quad \Delta p := 2\frac{p_{\max}}{M-1}, \tag{8.13}$$

for $j = \{1,\dots,M\}$. Finally, we discretise z as

$$z^t := (t-1)\Delta z, \quad \Delta z = \frac{z_{\max}}{T-1}, \tag{8.14}$$

for $t = \{1,\dots,T\}$ and $z_{\max}$ is the total length of the optical axis along which we integrate Liouville's equation. The numerical approximation of the solution is then denoted by $\rho_{ij}^t \approx \rho(z^t, q_i, p_j)$.

Whenever the refractive index is differentiable, (8.11) has a classical solution and the upwind scheme is straightforwardly found as

$$\begin{aligned}\frac{\rho_{ij}^{t+1}-\rho_{ij}^{t}}{\Delta z}+\max(a_{ij}^{t},0)\,\frac{\rho_{ij}^{t}-\rho_{i-1,j}^{t}}{\Delta q}+\min(a_{ij}^{t},0)\,\frac{\rho_{i+1,j}^{t}-\rho_{ij}^{t}}{\Delta q}\\ +\max(b_{ij}^{t},0)\,\frac{\rho_{ij}^{t}-\rho_{i,j-1}^{t}}{\Delta p}+\min(b_{ij}^{t},0)\,\frac{\rho_{i,j+1}^{t}-\rho_{ij}^{t}}{\Delta p}=0,\end{aligned} \tag{8.15a}$$

where

$$a_{ij}^{t}:=a(z^{t},q_i,p_j),\quad b_{ij}^{t}:=b(z^{t},q_i,p_j). \tag{8.15b}$$

As one can see, the expression in a two-dimensional optical system is already quite cumbersome, though not complicated. A three-dimensional optical system will just add four more upwind difference terms, two for each added dimension.

We now wish to find a scheme that gives us the correct physical solution whenever we allow n to have discontinuities. The correction to the upwind scheme applies only locally around the interface, away from the interface the scheme will be given by (8.15). Thus, to illustrate our method, we use the simplest case of a piecewise constant refractive index. We choose a system that has $0 \le q \le 1$. Fix $1 < k < N$, and let's place the interface at $q_{k+\frac{1}{2}} = q_k + \frac{1}{2}\Delta q$, i.e.,

$$n(q)=\begin{cases}n_1 & \text{if } q<q_{k+\frac{1}{2}},\\ n_2 & \text{if } q\ge q_{k+\frac{1}{2}}.\end{cases} \tag{8.16}$$

It's clear that $\frac{\partial n}{\partial q} = 0$ almost everywhere, thus $b(z,q,p) = 0$ at all the grid points. Away from the interface, $i \ge k+2$ or $i \le k-1$, the refractive index is smooth, resulting in the following scheme,

$$\frac{\rho_{ij}^{t+1}-\rho_{ij}^{t}}{\Delta z}+\max(a_{ij},0)\frac{\rho_{ij}^{t}-\rho_{i-1,j}^{t}}{\Delta q}+\min(a_{ij},0)\frac{\rho_{i+1,j}^{t}-\rho_{ij}^{t}}{\Delta q}=0, \tag{8.17}$$

where now a does not depend on z so that $a_{ij} = a(q_i, p_j)$. Even close to the interface, this scheme works as long as the upwind grid point isn't on the other side of the interface. Hence,

$$\frac{\rho_{kj}^{t+1}-\rho_{kj}^{t}}{\Delta z}+a_{kj}\frac{\rho_{kj}^{t}-\rho_{k-1,j}^{t}}{\Delta q}=0,\qquad p_j\ge 0, \tag{8.18a}$$

$$\frac{\rho_{k+1,j}^{t+1}-\rho_{k+1,j}^{t}}{\Delta z}+a_{k+1,j}\frac{\rho_{k+2,j}^{t}-\rho_{k+1,j}^{t}}{\Delta q}=0,\qquad p_j<0, \tag{8.18b}$$

since the sign of a_{ij} is equal to the sign of p_j. Let's now consider the collection of grid points (q_k, p_j) with $p_j < 0$ and (q_{k+1}, p_j) with $p_j > 0$. These are grid points that have their upwind grid point on the other side of the interface. Our scheme has to be different here since the characteristics have a jump in momentum when crossing the interface. We wish to approximate $\frac{\partial \rho}{\partial q}$ at the grid point (q_k, p_j), which we can do by utilizing Theorem 8.1.

The idea is once again straightforward, we perform Taylor expansions towards the interface from all grid points involved. In general, there will be one grid point on the downwind side, (q_k, p_j), and two on the upwind side. At the interface, we apply Snell's law, see Figure 8.3. Furthermore, we use Theorem 8.1 to relate the gradients on both sides of the interface. This allows us to approximate the gradient at (q_k, p_j) using information from the other side of the interface. The resulting scheme is summarised in the following theorem.

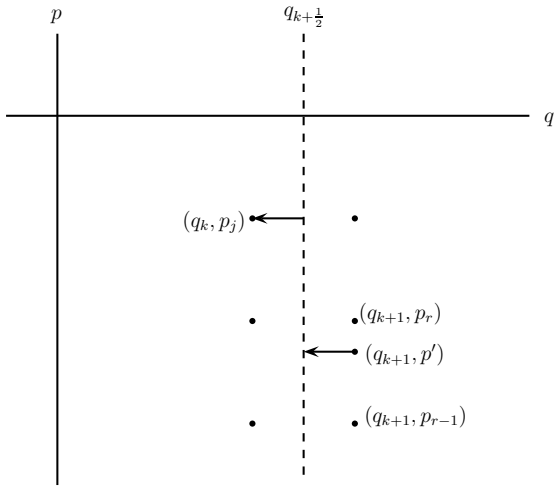


Figure 8.3: Sketch of the scheme close to the interface, the upwind grid point is on the other side of the interface. The ray propagation direction is indicated with arrows.

Theorem 8.2. *Consider the collection of grid points such that* $\{q_k, p_j < 0\}$. *Let's denote* $p' = -\mathcal{S}(-p_j; n_2, n_1, -\nu)$, *and* $\delta \geq 0$ *in* (8.2), *where* δ *is given by*

(8.2b). *The scheme is then given by*

$$\frac{\rho_{kj}^{t+1} - \rho_{kj}^{t}}{\Delta z} + \tilde{a}\frac{\rho' - \rho_{kj}^{t}}{\Delta q} = 0, \tag{8.19a}$$

where

$$\tilde{a} := 2\left(\frac{1}{a_{kj}} + \frac{1}{a'}\right)^{-1} \quad \textit{and} \quad a' = \frac{p'}{\sqrt{n_2^2 - p'^2}}, \tag{8.19b}$$

$$\rho' = \theta\rho_{k+1,r}^{t} + (1-\theta)\rho_{k+1,r-1}^{t}, \tag{8.19c}$$

where $\theta = (p' - p_{r-1})/\Delta p$, *with* r *such that* $p_{r-1} < p' \leq p_r$.

In the case that $\delta < 0$, *reflection occurs and we have to use* (8.19a), *now with*

$$\tilde{a} := 2\left(\frac{1}{a_{kj}} - \frac{1}{a'}\right)^{-1} \quad \textit{and} \quad a' = \frac{p'}{\sqrt{n_1^2 - p'^2}}, \tag{8.20a}$$

$$\rho' = \theta\rho_{k,r}^{t} + (1-\theta)\rho_{k,r-1}^{t}. \tag{8.20b}$$

Remark. *The case for* $\{q_{k+1}, p_j > 0\}$ *is similar.*

Proof. 1. We start by using the method of lines (MOL) approach, which is to say, we discretise space, but leave z continuous. Let's assume that j is such that $\delta \geq 0$, thus we have refraction. Let's further assume that the initial conditions are smooth. Furthermore, define $p' = -\mathcal{S}(-p_j; n_2, n_1, -\nu)$, in shorthand $p' = -\mathcal{S}(-p_j)$, such that p' is the momentum that becomes p_j if the characteristic traverses the interface. We define r as the unique index such that $p_{r-1} < p' \leq p_r$.

2. Performing a Taylor expansion to first order about the relevant grid points close to the interface on the left side reveals,

$$\rho(z^-, q_k, p_j) \approx \rho(z^-, q_{k+\frac{1}{2}}, p_j) - \frac{\Delta q}{2}\left.\frac{\partial\rho}{\partial q}\right|_{(z^-, q_k, p_j)}, \tag{$*$}$$

and similarly on the right side,

$$\begin{aligned}\rho(z^+, q_{k+1}, p_r) \approx \rho(z^+, q_{k+\frac{1}{2}}, p') &+ \frac{\Delta q}{2} \left.\frac{\partial \rho}{\partial q}\right|_{(z^+, q_{k+\frac{1}{2}}, p')} \\ &+ (p_r - p') \left.\frac{\partial \rho}{\partial p}\right|_{(z^+, q_{k+\frac{1}{2}}, p')} \\ \rho(z^+, q_{k+1}, p_{r-1}) \approx \rho(z^+, q_{k+\frac{1}{2}}, p') &+ \frac{\Delta q}{2} \left.\frac{\partial \rho}{\partial q}\right|_{(z^+, q_{k+\frac{1}{2}}, p')} \\ &+ (p_{r-1} - p') \left.\frac{\partial \rho}{\partial p}\right|_{(z^+, q_{k+\frac{1}{2}}, p')},\end{aligned}$$

where the error in both approximations is second-order. Next, we again Taylor expand to first order, this time for the q-derivative, i.e.,

$$\left.\frac{\partial \rho}{\partial q}\right|_{(z^-, q_{k+\frac{1}{2}}, p_j)} \approx \left.\frac{\partial \rho}{\partial q}\right|_{(z^-, q_k, p_j)} + \frac{\Delta q}{2} \left.\frac{\partial^2 \rho}{\partial q^2}\right|_{(z^-, q_k, p_j)}. \qquad (\#)$$

Now we apply (8.4) to find

$$a' \left.\frac{\partial \rho}{\partial q}\right|_{(z^+, q_{k+\frac{1}{2}}, p')} = a_{k+\frac{1}{2}, j} \left.\frac{\partial \rho}{\partial q}\right|_{(z^-, q_{k+\frac{1}{2}}, p_j)},$$

where a' is given by (8.19b). However, since n is assumed piecewise constant, we have $a_{k+\frac{1}{2}, j} = a_{kj}$. Hence, combining this with (#) gives us a first-order approximation, i.e.,

$$\left.\frac{\partial \rho}{\partial q}\right|_{(z^+, q_{k+\frac{1}{2}}, p')} \approx \frac{a_{kj}}{a'} \left.\frac{\partial \rho}{\partial q}\right|_{(z^-, q_k, p_j)} + \mathcal{O}(\Delta q),$$

and consequently

$$\begin{aligned}\rho(z^+, q_{k+1}, p_r) &\approx \rho(z^+, q_{k+\frac{1}{2}}, p') + \frac{\Delta q}{2}\frac{a_{kj}}{a'}\left.\frac{\partial \rho}{\partial q}\right|_{(z^-, q_k, p_j)} \\ &\quad + (p_r - p')\left.\frac{\partial \rho}{\partial p}\right|_{(z^+, q_{k+\frac{1}{2}}, p')} \\ \rho(z^+, q_{k+1}, p_{r-1}) &\approx \rho(z^+, q_{k+\frac{1}{2}}, p') + \frac{\Delta q}{2}\frac{a_{kj}}{a'}\left.\frac{\partial \rho}{\partial q}\right|_{(z^-, q_k, p_j)} \\ &\quad + (p_{r-1} - p')\left.\frac{\partial \rho}{\partial p}\right|_{(z^+, q_{k+\frac{1}{2}}, p')}.\end{aligned} \tag{$**$}$$

3. We'll now take a linear combination of the values of ρ at the three grid points (q_k, p_j), (q_{k+1}, p_r) and (q_{k+1}, p_{r+1}). We wish to find a linear combination which approximates the q-derivative of ρ at the grid point (q_k, p_j) using $(*)$ and $(**)$. Assume that $\lambda_l \in \mathbb{R}$, for $l = 1, 2, 3$, then the upwind finite difference should satisfy

$$\left.\frac{\partial \rho}{\partial q}\right|_{(z^-, q_k, p_j)} \approx \lambda_1 \rho(z^-, q_k, p_j) + \lambda_2 \rho(z^+, q_{k+1}, p_r) + \lambda_3 \rho(z^+, q_{k+1}, p_{r-1}),$$

where the approximation should be correct up to first-order. Using continuity of ρ along rays (8.3), we find the following system of equations,

$$\begin{aligned}\lambda_1 + \lambda_2 + \lambda_3 &= 0, \\ \frac{\Delta q}{2}\left(\frac{a_{kj}}{a'}\lambda_2 + \frac{a_{kj}}{a'}\lambda_3 - \lambda_1\right) &= 1, \\ \lambda_2(p_r - p') + \lambda_3(p_{r-1} - p') &= 0.\end{aligned}$$

Defining $\theta = (p' - p_{r-1})/\Delta p$, with $\theta \in [0, 1]$ by definition of r, the determinant of this system is given by $-\frac{\Delta q \Delta p}{2}\left(1 + \theta\frac{a_{kj}}{a'}\right)$. Since a_{kj} has the same sign as a', the determinant of the linear system is nonzero. Defining $\tilde{a}$ as the harmonic mean of a_{kj} and a', i.e., (8.19b), we find the solution of this system as

$$\lambda_1 = -\frac{\tilde{a}}{a_{kj}}\frac{1}{\Delta q}, \quad \lambda_2 = \frac{\tilde{a}}{a_{kj}}\frac{\theta}{\Delta q}, \quad \lambda_3 = \frac{\tilde{a}}{a_{kj}}\frac{1-\theta}{\Delta q}.$$

The resulting approximation for the q-derivative at (q_k, p_j) is given by

$$\left.\frac{\partial \rho}{\partial q}\right|_{(z^-, q_k, p_j)} \approx \frac{1}{\Delta q}\frac{\tilde{a}}{a_{kj}}\Big(\theta\rho(z^+, q_{k+1}, p_r) + (1-\theta)\rho(z^+, q_{k+1}, p_{r-1}) - \rho(z^-, q_k, p_j)\Big).$$

We now approximate the z-derivative by the Forward Euler method, whence we find the scheme

$$\frac{\rho_{kj}^{t+1} - \rho_{kj}^t}{\Delta z} + \tilde{a}\frac{\theta\rho_{k+1,r}^t + (1-\theta)\rho_{k+1,r-1}^t - \rho_{kj}^t}{\Delta q} = 0.$$

Recognizing that $\rho' := \theta\rho_{k+1,r}^t + (1-\theta)\rho_{k+1,r-1}^t$ is nothing but a linear interpolation approximating $\rho(z^t, q_{k+1}, p')$, we find (8.19a).

4. When j is such that $\delta < 0$, we have reflection and the derivation is largely the same. The only difference being that the upwind grid points are now given by (q_k, p_r) and (q_k, p_{r-1}) instead of (q_{k+1}, p_r) and (q_{k+1}, p_{r-1}). After doing completely similar operations, we find (8.19a) but now with ρ' given by (8.20b) and a' given by (8.20a). □

Remark. *When the surface is curved we find that we must furthermore replace* a_{kj} *with* $a_{kj} - Q'(z^\star)$ *and* a' *with* $a' - Q'(z^\star)$.

From Theorem 8.2, we see that the scheme close to an interface is still an upwind scheme since (8.19a) has completely the same form as (8.17). However, we must replace both the advection speed and the value of ρ at the upwind grid point. Fortunately, the replacement values can be explicitly computed by invoking Snell's law. The scheme is in essence a first order accurate upwind scheme, with the correction only occurring near the interface. We thus expect the scheme to be globally first order accurate.

As for the CFL condition, we should note that the harmonic mean of two numbers always lies in between them. Thus, the adjusted advection speed near the interface can never be higher than the advection speed occurring away from the interface. Furthermore, the adjusted value ρ' is a linear interpolation in p, which is a stable operation. Linear interpolation also exhibits the property that the computed value will lie in between the two interpolated values. As a result, the interface doesn't generate any instabilities. The CFL condition will therefore also be unaltered compared to the standard upwind scheme, i.e.,

$$|c| \leq 1, \tag{8.21}$$

with $c = \frac{a_{\max}\Delta z}{\Delta q}$ with $a_{\max}$ the maximum velocity in absolute value in the system.

8.3 Results

For the sake of simplicity, we present two cases which will exhibit all the features of the scheme that are different from a standard first-order upwind scheme. The first is the bucket of water problem, introduced in Chapter 4, which can be solved analytically fairly easily. The second test case is the two-dimensional compound parabolic concentrator (CPC). In the second test case, the CPC, we'll compare our method with Monte Carlo ray tracing, see Section 4.3.1 and for instance [23].

8.3.1 Bucket of water

We take a simple jump in refractive index as in (8.16) and we pick $n_1 = 1.4$ and $n_2 = 1$. The refractive index field is given by

$$n(q) := \begin{cases} n_1 & \text{if } q \leq q_{k+\frac{1}{2}}, \\ n_2 & \text{if } q > q_{k+\frac{1}{2}}, \end{cases} \tag{8.22}$$

where we pick k such that $q_{k+\frac{1}{2}} = \frac{1}{2}$. This case corresponds roughly to a water-air transition, and we'll at times refer to this problem as the bucket of water problem. One important thing to note is that the optical axis is parallel to the interface[2], resulting in a refractive index field which does not depend on z. This is, as we've also mentioned in Section 4.2.2, because otherwise there'd be nothing special to see on, the resulting distribution would simply look like it was subject to straightforward propagation in a constant medium.

As discussed in Chapter 4, the reason we choose this particular problem is because it exhibits both refraction and total internal reflection. At the same time, the problem is exactly solvable by the method of characteristics, see Chapter 4 for a complete description. Using an initial condition ρ_0 with support restricted

[2]Beware optical engineers and scientists: due to the definition of the optical axis, the momentum will *decrease* when rays transmit into the second medium!

to the upper left quadrant $\{q < \frac{1}{2}, p > 0\}$, the exact solution is given by

$$\rho(z,q,p) = \begin{cases} \rho_0\left(q - z\frac{p}{\sqrt{n_1^2-p^2}}, p\right), & \text{if } q < \frac{1}{2}, p \geq 0, \\ \rho_0\left(1 - q + z\frac{p}{\sqrt{n_1^2-p^2}}, -p\right), & \text{if } q < \frac{1}{2}, -p_c < p < 0, \\ \rho_0\left(\frac{1}{2} - (z - \Delta z(q,p))\frac{p'}{\sqrt{n_2^2-(p')^2}}, p'\right), & \text{if } q \geq \frac{1}{2}, p \geq 0, \\ 0 & \text{otherwise,} \end{cases} \tag{8.23}$$

where $p' = -\mathcal{S}(-p; n_2, n_1, -\nu)$ with $\mathcal{S}$ defined in (8.2) and

$$\Delta z(q,p) := \frac{\frac{1}{2} - q}{p}\sqrt{n_2^2 - p^2}. \tag{8.24}$$

The interval Δz is such that a ray which passes through the point (q,p), where $q > \frac{1}{2}$, at z will hit the interface at $z - \Delta z(q,p)$. In (8.23), the first statement is free propagation, the second is refraction, the third comes from total internal reflection and the fourth statement comes from the compact support of ρ_0. We introduce the critical momentum $p_c = \sqrt{n_1^2 - n_2^2} = 0.9798$, so that when $p < p_c$, total internal reflection occurs. This condition is perhaps counter-intuitive for some audiences, however this is due to the choice of optical axis[3]. We see that, although in principle not too difficult to solve, the expressions become large and unwieldy. Going to a slightly more complicated geometry already precludes analytical solutions.

We take as initial condition the distribution

$$\rho_0(q,p) = \begin{cases} 1 & \text{if } 0.3 \leq q \leq 0.35 \text{ and } 0 \leq p \leq 1.1, \\ 0 & \text{otherwise,} \end{cases} \tag{8.25}$$

see Figure 8.4. These initial conditions contain the critical momentum p_c, such that both refraction and total internal reflection will occur.

We use an integration length of $Z = 0.4$ with 1000 steps on an 800×800 grid. For the discretisation of phase space we use 800 grid points for both position and momentum, see Figure 8.5. We see from the figure that the numerical solution is very close to the exact solution. The only difference that is noticeable by eye is the numerical diffusion at the edges. However, note that the edge at the interface is sharp. The numerical diffusion is of course an effect of the first order

[3]See footnote 2.

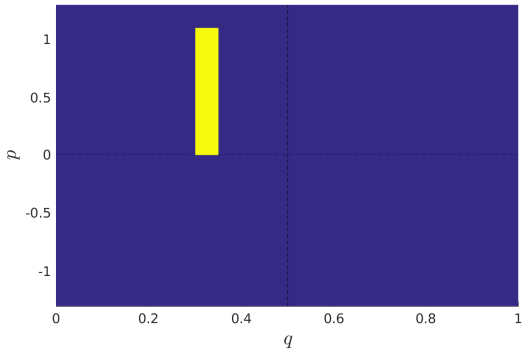


Figure 8.4: Initial condition of the bucket of water problem, black has value 1 and white has value 0. The transition is at $q = \frac{1}{2}$ and the zero momentum line $p = 0$ are indicated with a dashed line.

accurate upwind scheme. The sharp edges are an effect of the instantaneous transformation that occurs at the interface.

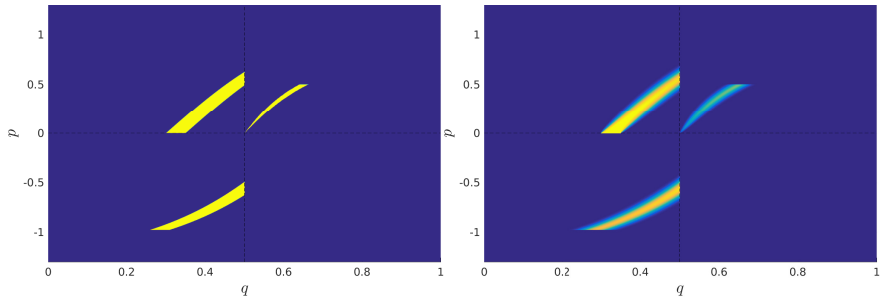


Figure 8.5: Exact (left) and numerical (right) solutions to the bucket of water problem at $z = 0.4$.

In this case, the bucket of water problem, we can implement the scheme in a matrix-vector formulation, where the evolution matrix is fixed. Hence, we only have to compute this evolution matrix once, and we can then use it for any initial condition. Every step in z then becomes a matrix-vector product, where the matrix is sparse. This won't be the case in our next example, where the evolution matrix has to be adjusted every time step. However, we'll first

investigate numerically the convergence properties of the scheme.

Convergence results

The convergence of our scheme is tested numerically by performing several runs of the bucket of water problem with different grid sizes. The refractive index is once again given by (8.22). However, as our analysis of the error is only valid for functions that are smooth away from the interface, we'll use a smooth initial condition, $\rho_0 \in C_0^\infty(\mathcal{P})$. This initial condition is constructed by using the bump function [71], given by

$$\psi(x) := \begin{cases} e^{-\frac{1}{1-x^2}}, & \text{for } |x| < 1, \\ 0, & \text{otherwise.} \end{cases} \tag{8.26}$$

One can check that the bump function is infinitely smooth and has compact support. The initial condition is constructed as follows,

$$\rho_0(q,p) := \psi\left(\frac{q-q_0}{\lambda_q}\right)\psi\left(\frac{p-p_0}{\lambda_p}\right), \tag{8.27}$$

where

$$q_0 = \tfrac{1}{4}, \quad \lambda_q = \tfrac{3}{20}, \quad p_0 = \tfrac{3}{5}, \quad \lambda_p = \tfrac{1}{2}, \tag{8.28}$$

which results in the support of ρ_0 being $\left\{\frac{1}{10} \le q \le \frac{2}{5}, \frac{1}{10} \le p \le \frac{11}{10}\right\}$. The integration time is chosen as $Z = \frac{2}{5}$, which results in a large part of the initial condition to be refracted and reflected by the interface.

We compute the error by using the exact solution and taking the L^1-norm of the difference, i.e.,

$$\mathrm{e}_L = \Delta q \Delta p \sum_{i=1}^{N}\sum_{j=1}^{M} \left|\rho\left(Z, q_i, p_j\right) - \rho_{ij}^T\right|, \tag{8.29}$$

where $Z = z^T = \frac{2}{5}$ and $\rho\left(Z, q_i, p_j\right)$ is the exact solution given by (8.23). For simplicity, we'll choose $\Delta q = \Delta p$, the results are displayed in Table 8.1. The order of convergence, denoted γ, is computed using

$$e_L \approx \frac{C_1}{E^{\gamma/2}} = \frac{C_2}{N_q^\gamma}, \tag{8.30}$$

for some constants $C_1, C_2 > 0$ and E the total number of grid points. Doubling the grid size allows for easy estimation of the order of convergence. We'll use

Table 8.1: Error and convergence rates using the bucket of water problem as a benchmark.

N_q	$e_L[10^{-4}]$	order
200	11.1	
400	6.19	0.84
800	3.34	0.89
1600	1.75	0.94
3200	0.85	0.98

a large number of z-steps, 15000, to make sure that the z-integration error is much smaller than the spatial errors.

As the analytical solution for this problem is known, we can use it as a benchmark test and determine the error of our solver exactly. The results clearly show that our scheme is first order accurate in the number of grid points in the q-direction. This is as expected since an ordinary upwind solver has an error of $\mathcal{O}(\Delta q) + \mathcal{O}(\Delta p)$. The adjustments to the upwind scheme have virtually no impact on the error behaviour.

8.3.2 Compound parabolic concentrator

The compound parabolic concentrator is a set of parabolic mirrors that perfectly concentrate light, see Chapter 4. It accepts light incoming within an acceptance angle θ from the optical axis. The width of the bottom aperture is $2a$, while the length of the optic, Z, is given by (4.23), i.e.,

$$Z = a\frac{(1+\sin\theta)\cos\theta}{\sin^2\theta}.$$

The initial condition is chosen as Lambertian light at $z = 0$, i.e.,

$$\rho_0(q,p) = \begin{cases} 1, & \text{for } -a \le q \le a, -1 \le p \le 1, \\ 0, & \text{otherwise.} \end{cases} \tag{8.31}$$

Since the CPC is a perfect concentrator, the solution at $z = Z$ is given by

$$\rho(Z,q,p) = \begin{cases} 1, & \text{for } -a \le q\sin\theta \le a, -\sin\theta \le p \le \sin\theta, \\ 0, & \text{otherwise.} \end{cases} \tag{8.32}$$

The results are plotted in Figure 8.6. We compute the numerical solution using a 100×100 grid and 800 time steps. We have used $\theta = \frac{\pi}{6}$ and $a = \frac{1}{2}$ and integrated Liouville's equation on the CPC to $z = Z$.

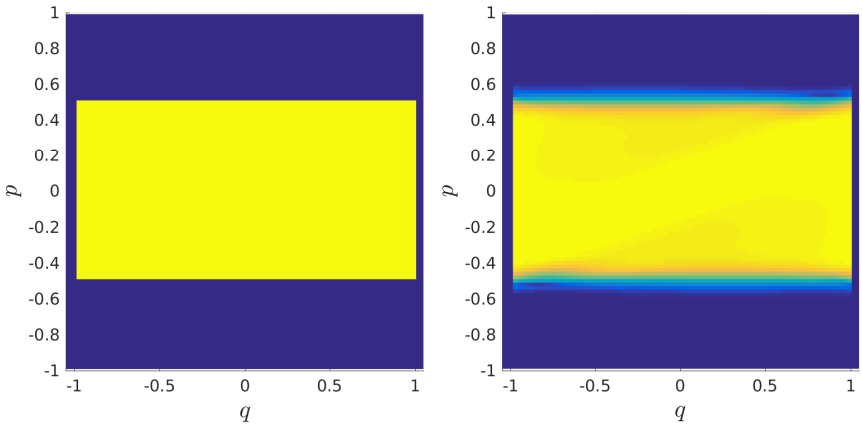


Figure 8.6: Exact solution for the CPC. Numerical solution for the CPC.

The figure shows a great resemblance between the numerical and exact solutions. Apart from some numerical diffusion at the top and bottom edges, the numerical solution is equal to the exact solution. An intermediate result at $z = Z/3$ is shown in Figure 8.7. It shows that the numerical diffusion comes from the upwind scheme, whereas the sharp edges come from the symplectic transformation that is the reflection.

The Liouville solver is again implemented using a matrix-vector formulation. We first construct a basic evolution matrix, so that every time step only $2N_p$ entries need to be adjusted. These entries correspond to the grid points that are downwind from the reflecting walls. The exact solution is only known at $z = Z$, hence that is where we compare both methods against the exact solution. We choose $N_p = N_q$ and we fix the CFL number for the q-direction to be 0.96.

We compare the upwind method to first-order ray tracing as explained in Section 4.3.1. This test problem is extremely favourable for ray tracing. First off, intersection points with the CPC wall can be determined analytically. Ordinarily this would require a root-finder to find the intersection points between rays and the curved surface, hence some computation time is saved. Furthermore, due to the uniformity of the grid, the correct bin can be computed explicitly in

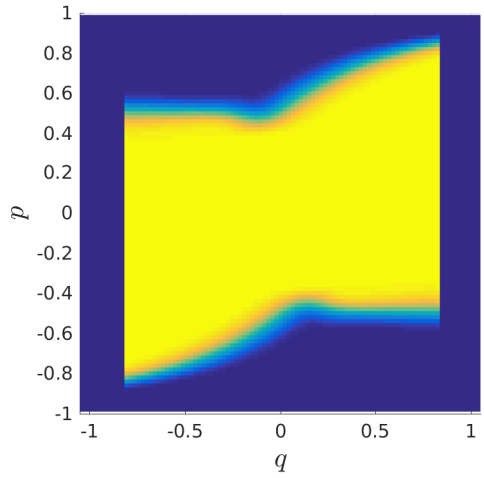


Figure 8.7: Intermediate result at $z = Z/3$.

constant time for any ray, i.e., no search over the bins is needed. Finally, there's some ambiguity in the correspondence between elements and bins, as the numerical solution is only defined on grid points for the upwind method. Here, we associate with each grid point a control volume $[q_i - \frac{\Delta q}{2}, q_i + \frac{\Delta q}{2}] \times [p_j - \frac{\Delta p}{2}, p_j + \frac{\Delta p}{2}]$ and use these as bins.

The definition of the bins means that (q_i, p_j) is the midpoint for its bin. Using the error (8.29) for the solution obtained by ray tracing should therefore be second-order convergent according to Section 4.3.1. As we're aiming to use a first-order convergent ray tracer, we therefore only need to use $N_{\mathrm{RT}} = \mathcal{O}(E) = \mathcal{O}(N_q^2)$. Hence, we can therefore use much fewer rays than we'd otherwise need. Constant-time search for the correct bin then means that $t_{\mathrm{RT}} = \mathcal{O}(N_{\mathrm{RT}}) = \mathcal{O}(N_q^2)$. Compare this to the ordinary case, where we'd need $N_{\mathrm{RT}} = \mathcal{O}(E^2) = \mathcal{O}(N_q^4)$.

The test consists of varying the number of grid points, and therefore the bins, while choosing the number rays to scale linearly with the number of elements. In particular, we use $N_{\mathrm{RT}} = 600 N_q^2$. The constant 600 was chosen such that on the coarsest grid both errors are roughly equal. We point out that typical choices for the grid size in applications are 300×300, hence we've chosen our comparison to include this particular value. The results are displayed in Table

Table 8.2: Computation times and error of the CPC test case. The subscript L stand for Liouville while RT stands for Ray Tracing.

N_q	$e_L[10^{-2}]$	t_L	$N_{RT}[10^6]$	$e_{RT}[10^{-2}]$	t_{RT}
100	8.31	2.37 s	6	8.44	47.7 s
200	6.48	25.77 s	24	6.47	3 min 7 s
300	5.72	2 min 12 s	54	5.86	7 min 24 s
400	5.16	6 min 52 s	96	4.71	12 min 50 s

8.2.

For all grid sizes, our method is faster, in terms of computation time, than Monte Carlo ray tracing. The scaling behaviour for the upwind solver is as predicted, scaling quadratically with the number of bins, thus $t_L = \mathcal{O}(N_q^4)$. In practice, however the upwind solver is faster for all grid sizes that we've looked at. Hence, even with a comparison that's extremely biased towards ray tracing, the Liouville solver is still faster in practice.

8.4 WENO?

In theory, it's fairly straightforward to extend the upwind Liouville solver to larger stencils and therefore higher orders. For instance, it's reasonably clear how we should apply a WENO scheme to Liouville's equation. In practice, however, the details around an optical interface become extremely convoluted frighteningly quickly. The simple upwind scheme is already somewhat complicated when having to deal with interfaces.

For any stencil-based method, far from the interface we should simply apply the original scheme directly to Liouville's equation. Only close to the interface do we actually need to alter the scheme. As an example, whereas we have two cases of where the interface can be on the stencil for a three-point scheme, a five-point scheme would provide us with four different cases. Crossing the interface, we need to compute the backward ray momentum by Snell's law, which will in general not be exactly on a grid point. Therefore, interpolation is needed in the p-direction as well. Of course, the interpolation in the p-direction needs to be of the same accuracy as the q-direction.

The complications arise here. First, we need to decide how to put a stencil around the backward ray momentum. For instance, do we somehow apply an upwind bias or not? And if so what does this upwind bias mean? Next, we

need to determine the interpolation. In the case of a WENO scheme, this will in general involve recomputing new interpolation weights every single time, as the backward momentum can be anywhere between two grid points. This also implies that we need to determine the WENO weights every single time. Ordinarily, WENO schemes explicitly use the fact that the cell edges are exactly in between two grid points. It may be the case, for instance, that a convex combination resulting in higher-order accuracy is not even possible.

For these reasons, we've decided not to pursue higher-order upwind or WENO schemes. Instead, we've searched for other higher-order methods. One desired property is that the methods are locally defined, e.g., on elements, instead of using information from an extended stencil. Examples of such methods are, of course, the active flux scheme and spectral element methods, which we'll visit in the next two chapters.

Chapter 9

An active flux Liouville solver

Fast is fine, but accuracy is final.

Wyatt Earp

As our previous discussion on WENO schemes shows, any higher-order scheme for Liouville's equation cannot rely on large stencils. The complications in possible cases with the stencil across an interface compound quickly as the stencil size increases. We therefore need to use a method that works on elements instead of a stencil. In such methods, higher orders are achieved on the elements by internal degrees of freedom. One such method is the active flux scheme, where the elements are triangular. We'll see an example of the active flux scheme on both a fixed mesh and on a moving mesh by the novel method we introduced in Section 6.5. Third-order accuracy is achieved by using point values around the boundary of each element together with the average value on the element. Communication between elements occurs through the flux, which allows us to incorporate Snell's law.

9.1 Details of the scheme

We'll now construct a Liouville solver based on the active flux scheme presented in Chapter 6. To this end, we consider the conservative form of Liouville's

equation in two-dimensions, i.e.

$$\frac{\partial \rho}{\partial z} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{9.1}$$

where $\nabla = \left(\frac{\partial}{\partial q}, \frac{\partial}{\partial p}\right)^T$ and $\mathbf{u} = \boldsymbol{S}\nabla h$, with

$$\boldsymbol{S} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}. \tag{9.2}$$

Compared to the standard active flux scheme, the flux is now linear in ρ but depends on the phase space coordinates, q and p, and possibly on z. As such, the characteristics can no longer be found by solving an algebraic equation. The basic working, however, remains unaltered: employ a semi-Lagrangian step to find the point values as a function of z.

As we've seen, solutions to Liouville's equation are constant along rays, hence the semi-Lagrangian step is equivalent to local ray tracing. This entails integrating Hamilton's equations using a node in the mesh as a terminal condition and applying

$$\rho(z + \Delta z, q(z + \Delta z), p(z + \Delta z)) = \rho(z, q(z), p(z)). \tag{9.3}$$

Using the shorthand notation for the velocity field, Hamilton's equations can be written as

$$\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}z} = \mathbf{u}(z, \mathbf{y}). \tag{9.4}$$

It's prudent to solve this ODE using a symplectic, or geometric, integrator [31, 35]. The time discretisation is denoted $z^t = t\Delta z$, with $t = 0, 1, \ldots, M$. The ODE (9.4) is integrated backwards in z to find $\mathbf{y}^t$ with the condition that $\mathbf{y}^{t+1}$ is on a node.

Also in this case, the position $\mathbf{y}^t$ only has to be determined to second-order accuracy in z, i.e. $\mathbf{y}^t = \mathbf{y}(z^t) + \mathcal{O}(\Delta z^2)$. This is due to the integration over the boundary, making the error in the total flux $\mathcal{O}(\Delta y \Delta z^2)$, with Δy some representative distance in the mesh. Active flux operates under the assumption that $\mathbf{y}^t$ lies in a direct neighbour of the origin element, implying the CFL-type condition $\Delta z \leq \frac{\Delta y}{b_m ax}$ with $b_{\max}$ the maximum magnitude of the velocity field. Thus, when using a second-order accurate integrator to find $\mathbf{y}^t$, the mixed error in the flux becomes $\mathcal{O}(\Delta y^3)$.

Following through on this reasoning, the implicit midpoint rule provides an excellent integrator for our purposes, which is a second-order accurate symplectic integrator that is time reversible, given by

$$\mathbf{y}^t = \mathbf{y}^{t+1} - \Delta z \mathbf{u}\left(z^t + \tfrac{1}{2}\Delta z, \frac{\mathbf{y}^t + \mathbf{y}^{t+1}}{2}\right). \tag{9.5}$$

This is an implicit equation for $\mathbf{y}^t$ which can be solved using, for instance, Newton's method.

As an added benefit, the midpoint method comes with a built-in second-order accurate approximation to $\mathbf{y}(z^{t+\frac{1}{2}})$, which we should recall, is also required by the scheme. Indeed, we have the Taylor expansions

$$\mathbf{y}(z^{t+1}) = \mathbf{y}\left(z^{t+\frac{1}{2}}\right) + \tfrac{1}{2}\Delta z \mathbf{u}\left(z^{t+\frac{1}{2}}, \mathbf{y}\left(z^{t+\frac{1}{2}}\right)\right) + \mathcal{O}(\Delta z^2), \tag{9.6a}$$

$$\mathbf{y}(z^{t}) = \mathbf{y}\left(z^{t+\frac{1}{2}}\right) - \tfrac{1}{2}\Delta z \mathbf{u}\left(z^{t+\frac{1}{2}}, \mathbf{y}\left(z^{t+\frac{1}{2}}\right)\right) + \mathcal{O}(\Delta z^2). \tag{9.6b}$$

Taking the average of the two expansions leads to

$$\frac{\mathbf{y}(z^t) + \mathbf{y}(z^{t+1})}{2} = \mathbf{y}\left(z^{t+\frac{1}{2}}\right) + \mathcal{O}(\Delta z^2). \tag{9.7}$$

Once the point $\mathbf{y}^t$ has been computed, we perform a local search over the neighbouring elements. Moreover, since $\mathbf{y}^{t+\frac{1}{2}}$ is simply the midpoint of the line segment connecting $\mathbf{y}^t$ and $\mathbf{y}^{t+1}$, all three points will always lie in the same element. By the CFL condition, the origin node $\mathbf{y}^{t+1}$ will be in the same element as $\mathbf{y}^t$, while (9.7) is simply the midpoint of the line segment connecting the endpoints. By convexity, the midpoint is in the same element as the endpoints, see Figure 9.1

After identifying the correct element, both points are transformed to the reference coordinates in χ and the point value is reconstructed as detailed in Section 6.2.1.

9.2 Mesh alignment

In optical systems, ray momentum changes discontinuously when encountering an optical interface. A ray that encounters an interface will be refracted according to Snell's law or reflected according to the law of specular reflection. Rays that are so affected will therefore carry their energy from one part of phase

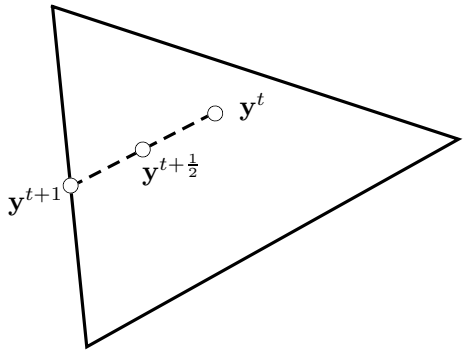


Figure 9.1: The CFL condition ensures that $\mathbf{y}^t$ is within an element, while $\mathbf{y}^{t+1}$ is by construction on the edge of the same element. Triangles are convex, hence the midpoint $\mathbf{y}^{t+\frac{1}{2}}$ is also in the same element.

space to a different part. Clearly, this behaviour must be incorporated into the solver to find physically relevant solutions, much as was done in the previous chapter with the upwind method. Since ρ is transported along rays and rays are discontinuous across interfaces, the solution itself will be discontinuous across interfaces, even for smooth initial conditions.

As a consequence, it's necessary to align elements with the interface, see Figure 9.2. For refractive surfaces, it's also necessary to have two values for all the nodes that are on the interface, one for each one-sided limit. The point values for elements on the interface have to be treated slightly differently. If a backward ray from a node crosses the interface, Snell's law or the law of specular reflection has to be applied. As the point values on element boundaries uniquely specify the flux, this is sufficient to capture the physical solution[1].

> The mesh must be aligned with the interface in phase space, i.e., the mesh must be such that the interface is equal to the union of a set of edges. Elements that have these edges as part of their boundary are called interface elements.

Note that this central idea of mesh alignment implies a moving mesh in some cases. As example, consider the compound parabolic concentrator where one

[1] As I alluded to earlier, it's surprisingly easy to accommodate optics problems with the active flux scheme. When I started this work, I thought the upwind method would be easiest to adapt.

wall is described by $q = Q(z)$; the curved mirror manifests itself as a moving boundary in phase space. For the mesh to be aligned for all z, it has to move along with the mirror. To be able to do so, we'll use the theory described in Section 6.5. The CPC will be treated as a numerical example in Section 9.3.2.

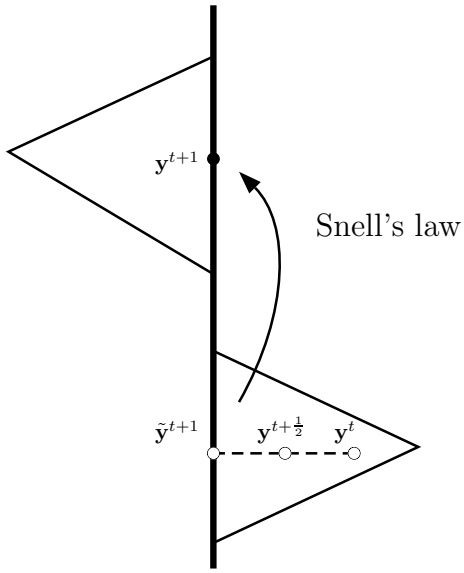


Figure 9.2: Two interface elements and a backward ray (dashed line) originating from a node on the interface.

Due to the CFL condition as discussed in Section 6.4, the only nodes where Snell's law actually has to be applied are located on the interface itself. In particular, consider a node that is part of an interface element, but doesn't lie on the interface itself. The CFL condition ensures that the time step is chosen such that even for the closest such nodes, the backward ray will never cross the interface, see Figure 9.3. Therefore, Snell's law will only have to be applied to nodes that lie on the interface themselves.

Another consequence of the mesh alignment is that we can consider all three points of the backward ray as lying in the same interface element. The solution should be continuous along characteristics, hence the left- and right-sided limits along the ray towards the interface should have the same value. In Figure 9.2, this is indicated by the points $\tilde{\mathbf{y}}^{t+1}$ and $\mathbf{y}^{t+1}$, where the solution has the same

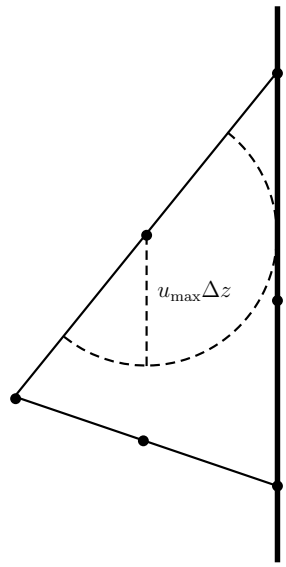


Figure 9.3: When the mesh is aligned with the interface, the CFL condition guarantees that nodes that don't lie on the interface can't have rays that cross the interface.

value. Therefore, it doesn't matter whether we use the originating node, $\mathbf{y}^{t+1}$, or its image under Snell's law, $\tilde{\mathbf{y}}^{t+1}$. Thus, computationally we can work with three points that are all in the same interface element.

Finally, optical interfaces may have critical momenta at which the behaviour of Snell's law changes from refractive to reflective. The velocity field at these points changes sign discontinuously. For stability, therefore, nodes must be placed at these critical momenta as well. The meshing software we've employed, Shewchuck's Triangle package, allows forced nodes in the triangular mesh to be set [118].

9.3 Results

As our test cases, we use again the bucket of water problem and the CPC. For both problems, exact solutions are given for special cases in Chapter 4. The bucket of water problem is used for a convergence test, demonstrating that the

method is third-order accurate. The CPC is used to demonstrate the active flux method on a moving grid.

9.3.1 Bucket of water

The simplest nontrivial optics problem is a single flat interface between two media of constant refractive index. This problem was previously dubbed the bucket of water problem, as the problem roughly represents a beam of light shone at a water surface. The refractive index is given by

$$n(q) = \begin{cases} n_1 & \text{if } q < 0, \\ n_2 & \text{if } q \geq 0, \end{cases} \tag{9.8}$$

where we choose $n_1 = 1.4$ and $n_2 = 1$. Using an initial condition that has support with $q < 0$ and $p > 0$, the solution features both refraction and total internal reflection. A typical result is shown in Figure 9.4, which was generated with an initial condition

$$\rho_0(q,p) = \begin{cases} 1 & \text{if } -\frac{1}{5} < q < -\frac{1}{2} \text{ and } 0 < p < \frac{13}{10}, \\ 0 & \text{otherwise.} \end{cases} \tag{9.9}$$

The result clearly shows the four regions as discussed in Section 4.2.2, where we derived the analytical solution. The four regions are named after their behaviour: the source, refraction, reflection and dark regions. There is a minor amount of numerical diffusion, much less compared to the upwind method from the previous chapter, see Figure 9.4. Note also that the solution is perfectly discontinuous along the interface at $q = 0$, as expected.

To verify the assertions made in Chapter 4 about the computation times and convergence rates of both active flux and ray tracing, we perform some numerical experiments. For the convergence test, we need to use an initial condition that results in a solution that's piecewise smooth to the right and left of the interface. To do so, recall the bump function

$$\psi(x) := \begin{cases} e^{-\frac{1}{1-x^2}}, & \text{for } |x| < 1, \\ 0, & \text{otherwise,} \end{cases} \tag{9.10}$$

which is a C_0^∞ function, meaning it's infinitely differentiable and has compact support. We use it to construct a smooth initial condition defined as

$$\rho_0(q,p) = \psi\left(\frac{q-q_0}{\lambda_q}\right)\left[\psi\left(\frac{p-p_0}{\lambda_p}\right) + \psi\left(\frac{p-p_1}{\sigma_p}\right)\right], \tag{9.11}$$

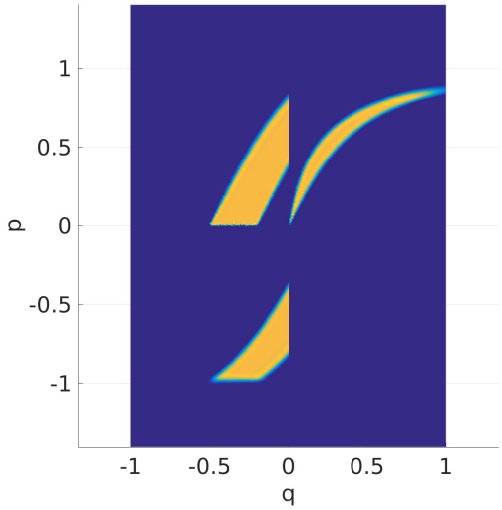


Figure 9.4: Numerical solution to the bucket of water problem using 31,293 elements and 1,216 time steps.

with the constants defined as $q_0 = -\frac{1}{2}$, $\lambda_q = \frac{1}{4}$, $p_0 = \frac{9}{20}$, $\lambda_p = \frac{9}{20}$, $p_1 = \frac{23}{20}$ and $\sigma_p = \frac{3}{20}$. This initial condition is chosen such that the light that's reflected is separate from the light that's transmitted, i.e., the critical momentum is avoided. As such, this ensures that the solution stays piecewise smooth, where the only discontinuity in the solution occurs at the interface.

For the number of rays, we used $N_{\text{RT}} = \frac{1}{25}E^2$ with E the number of elements. As outlined in Chapter 4, this is the optimal scaling for the number of rays compared to the number of elements. The constant $\frac{1}{25}$ is chosen to give reasonable results. Too few rays means not every bin receives at least one ray, resulting in an error of $\mathcal{O}(1)$ and thus a nonsensical numerical solution. We've empirically determined the constant $\frac{1}{25}$ such that on the coarsest grid, the ray tracing solution at least makes sense. The error is computed using the L^2-norm on phase space, hence

$$e = \left(\int_{\mathcal{P}} \left(\rho_{\text{AF}}(Z, q, p) - \rho(Z, q, p) \right)^2 \mathrm{d}y \right)^{\frac{1}{2}}, \tag{9.12}$$

with ρ_{AF} the numerical solution and ρ the exact solution. This integral is numerically evaluated by first interpolating both ρ_{AF} and ρ to a fine Cartesian grid and then using a quadrature rule. The fine grid is chosen such that the error in the quadrature rule is much smaller than the error of the numerical solution. In case of the bucket of water problem, the exact solution and both reconstructions are known on each element, hence the error can be computed to sufficient accuracy. Furthermore, the error scaling as a function of the number of elements is assumed to satisfy

$$e = C_e E^{-\frac{\gamma}{2}}, \tag{9.13}$$

with γ the order of convergence and $C_e > 0$ some constant. This error scaling is valid in the limit of large numbers of elements. It can be estimated using two numerical solutions of the same problem for a different number of elements. Both algorithms were implemented in Fortran and run on a single core of a Dell Optiplex 7010. In practice, we must remark, commercial ray tracing software would use a multithread implementation, leading to much shorter computation times. However, the active flux scheme also allows for copious amounts of parallel processing. It's for this reason that we implemented both in a single-thread fashion.

Table 9.1 shows convergence data for the two methods. The meshing software allows for setting a maximum element size, which scales inversely with the

Table 9.1: Errors and estimated orders versus the maximal element size for the convergence test. Active flux is indicated with subscript AF while ray tracing is indicated with subscript RT.

$\max_i \lvert\Omega_i\rvert$	e_{AF}	γ_{AF}	N_{RT}	e_{RT}	γ_{RT}
$5.1 \cdot 10^{-2}$	$2.1 \cdot 10^{-2}$		795	$3.5 \cdot 10^{-2}$	
$1.3 \cdot 10^{-2}$	$9.1 \cdot 10^{-3}$	1.2	14,933	$1.6 \cdot 10^{-2}$	1.2
$3.2 \cdot 10^{-3}$	$2.9 \cdot 10^{-3}$	1.7	235,225	$8.0 \cdot 10^{-3}$	1.0
$8.0 \cdot 10^{-4}$	$5.5 \cdot 10^{-4}$	2.4	3,785,359	$3.9 \cdot 10^{-3}$	1.0
$2.0 \cdot 10^{-4}$	$8.4 \cdot 10^{-5}$	2.7	31,330,826	$1.9 \cdot 10^{-3}$	1.0
$5.0 \cdot 10^{-5}$	$1.2 \cdot 10^{-5}$	2.8	980,917,344		

number of elements. Because the problem is two-dimensional, we've chosen to multiply the element size by $\frac{1}{4}$ each time, so that the error of the active flux scheme should reduce by a factor of eight. Note that the table doesn't show ray tracing data for the smallest element size, as the computation time grew prohibitively large. Both schemes hold up to the theoretical arguments, the predicted error scaling is verified by the data. The active flux scheme is indeed third-order accurate while the ray tracing method provides first-order accuracy. It's interesting to note that the lowest error achieved by ray tracing is roughly comparable to the error from the active flux scheme with $\max_i |\Omega_i| = 3.2 \cdot 10^{-3}$.

The computation times of the previous test are shown in Table 9.2. Like the error scaling, the time scaling is determined empirically using a power law assumption, i.e.,

$$t = C_t E^s, \tag{9.14}$$

with some constant $C_t > 0$, which is valid as the number of elements grows large. The time scaling for ray tracing indeed comes out as predicted in Chapter 4, i.e., cubic in the number of elements. The time scaling of the active flux scheme is close to the predicted quadratic, although it takes even more elements to find the limit numerically. For ray tracing on the other hand, the limit is indeed reached. Now that the theoretical time scaling is confirmed for both methods, we can estimate the computation time for the last entry of the table for ray tracing: it would take over 21 days using roughly 980 million rays. We can now compare also the error-versus-time behaviour. For an error around $3 \cdot 10^{-3}$, ray tracing takes almost eight hours, while the active flux scheme is done in 4.6 seconds. It seems an understatement to say that the speed-up is significant.

Table 9.2: Number of elements E, computation times t and their scaling s of the previous test. Active flux is indicated with subscript AF while ray tracing is indicated with subscript RT.

E	t_{AF}	s_{AF}	t_{RT}	s_{RT}
141	0.6 s		0.051 s	
611	1.4 s	0.58	0.12 s	0.54
2,425	4.6 s	0.86	6.8 s	2.91
9,728	24.3 s	1.20	7 min 0 s	2.97
39,157	4 min 19 s	1.70	7 h 37 min 45 s	3.00
156,598	55 min 54 s	1.85		

9.3.2 Compound parabolic concentrator

The compound parabolic concentrator (CPC), see Section 4.2.3, provides a rather difficult test case for the active flux scheme. The CPC features rays that are reflected any number of times on one side. That is, for any integer r, there are rays that are reflected r times on one wall. The limiting case are the so-called whispering modes that move continuously along one parabola, reflecting an infinite number of times. Furthermore, rays that travel close to perpendicular to the optical axis carry some energy.

The advection speed for perpendicular rays, i.e., rays with momentum $p = \pm 1$, is infinite. To cope with this, we have to set some maximum momentum $p_{\max} = 1 - \varepsilon$ for some $\varepsilon > 0$. We choose $\varepsilon = \frac{1}{2} l_{\min}$, where $l_{\min}$ is the minimum edge length in the initial grid. Furthermore, it's somewhat easier to implement the CPC in the reverse direction, as an ideal diluter. This is due to the fact that momenta tend to decrease in this direction, while they would otherwise increase when the CPC is employed as an ideal concentrator. The initial conditions therefore consist of

$$\rho_0(q,p) = \begin{cases} 1 & \text{if } -a \le q \le a \text{ and } -p_{\max} \le p \le p_{\max}, \\ 0 & \text{otherwise.} \end{cases} \tag{9.15}$$

We use the parameters $a = \frac{1}{2}$ and $\theta = \frac{\pi}{6}$ for the CPC, where a is the exit aperture width and θ the acceptance angle. The optic length is given by (4.23), which gives $Z \approx 2.6$. We fix the left and right sides of the grid to the position of the mirrors, while the rest of the grid is uniformly stretched. We set a maximum element size of 10^{-2}, resulting in 3,345 triangles and 10,201 total degrees of freedom. The CFL condition results in 2765 time steps. The outcomes are

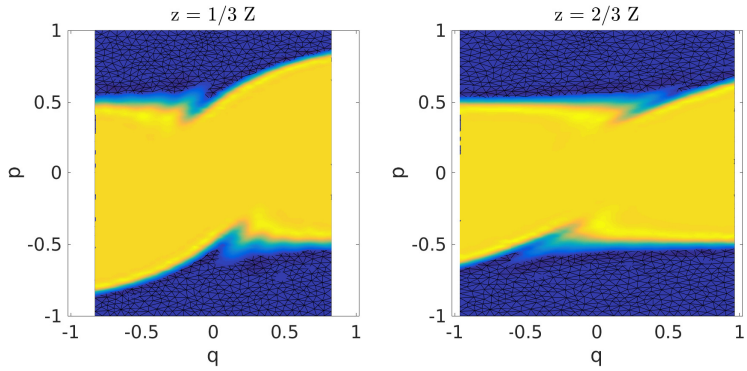


Figure 9.5: Numerical solution of the CPC problem at various z levels.

shown at several z-levels. In Figure 9.5, the solution is shown at $z = \frac{1}{3}Z$ and at $z = \frac{2}{3}Z$. In Figure 9.6 the final solution at $z = Z$ is shown.

The analytical solution at $z = Z$ is simply a rectangular area, $-1 \leq q \leq 1$ and $-\frac{1}{2} \leq p \leq \frac{1}{2}$, where $\rho = 1$ and elsewhere $\rho = 0$. The numerical solution has a slightly different shape, gaps are visible near the top-right and bottom-left. This is due to the restriction that we must have a finite maximum velocity in our scheme. Other than that, the solution is rather good. The error in the inner region is within machine precision of the exact solution, which is caused by the constant-state preserving property of the scheme, see Section 6.5

9.4 Concluding remarks

The active flux scheme is the first numerical method we've examined that truly outperforms Monte Carlo ray tracing for our two test problems. It's both faster and more accurate, which results in similar performance in seconds compared to what ray tracing does in hours. There can be no doubt about the superiority of solving Liouville's equation over ray tracing when global information is required. Naturally, the type of information is different, as no optical path is generated or stored.

Next, we'll examine a spectral element method applied to Liouville's equation. The main difference with the active flux scheme is that spectral element methods are *hp*-methods, where h stands for element size and p stands for the order of approximation [130, 131]. Clearly, the active flux scheme is designed to

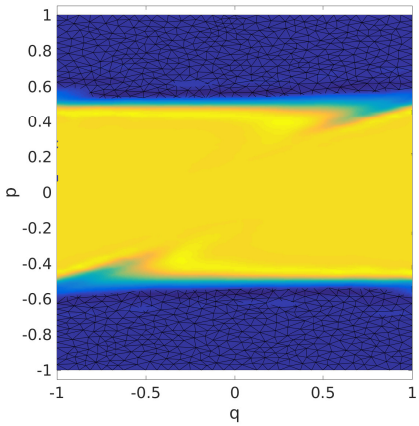


Figure 9.6: Numerical solution of the CPC problem at $z = Z$.

accommodate h-refinement. However, changing the order of approximation currently requires designing a completely new scheme, although this may of course change in the future. Spectral element methods have the ability to refine the mesh or enrich the order of approximation on the fly.

Chapter 10

A spectral element Liouville solver

> SPECTRE always delivers what it promises.
>
> Ernst Stavro Blofeld
> *From Russia with love*

In the previous chapter, we've discussed the active flux scheme, a numerical discretisation technique that computes a numerical solution on triangular elements. Before that, we've seen how to construct a upwind Liouville solver in Chapter 8. Remarkably, it turned out to be relatively easy to convert the active flux scheme to a solver for geometric optics problems. In particular, for the upwind method we needed to derive an extra jump condition on the solution and apply it across the interface in the numerical solution. For the active flux scheme, it was really just a matter of properly aligning the mesh with any interfaces present. We demonstrated this important concept with the compound parabolic concentrator, a pair of curved mirrors. For our next solver, which is based on the discontinuous Galerkin method, we again find that a proper alignment of the mesh is all it takes. We'll extend this idea to encompass lenses and other freeform refractive surfaces.

As our last Liouville solver, we'll discuss how to implement the discontinuous

Galerkin spectral element method. Recall that Liouville's equation is given by

$$\frac{\partial \rho}{\partial z} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{10.1}$$

where $\nabla = \left(\frac{\partial}{\partial q}, \frac{\partial}{\partial p}\right)^T$ and $\mathbf{u} = \boldsymbol{S}\nabla h$, with

$$\boldsymbol{S} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}. \tag{10.2}$$

As with the previous two methods, upwind and active flux, whenever the refractive index is smooth, the scheme can be applied off-the-shelf without modifications. However, when optical interfaces are present, the scheme needs to be altered locally.

10.1 Incorporating Snell's law

The DG spectral element method works purely in the Eulerian picture, no references to rays or characteristics are made in the derivation presented in Chapter 7. This as opposed to the active flux scheme, which uses local ray tracing to advance the point values in time. At first glance, therefore, one might wonder how to incorporate Snell's law into the solver. A closer look reveals that numerical solutions produced by DG are smooth in the element interiors and the only communication between elements is facilitated through a numerical flux. This suggests that the effects of optical interfaces have to be incorporated into the numerical flux.

To see how this is done, we think of the two regions of phase space separated by the interface as two separate problems. In both regions, the brightness ρ satisfies a linear hyperbolic equation. Moreover, we know that the brightness distribution should be continuous along rays. Therefore, on the parts of the interface where the velocity field points out of the interface, we have a Dirichlet boundary condition. The boundary value of one region prescribed by the brightness in the other region. Furthermore, Snell's law tells us from where to obtain the boundary value. Whenever the velocity field points into the interface, the brightness is left free. We can conclude that whenever the upwind direction leads across an interface, we need to apply Snell's law first, otherwise, we apply free outflow.

As alluded to earlier, like with the active flux scheme, we need to align the mesh with any optical interfaces, see Figure 10.1. For completeness, we repeat the idea of mesh alignment from the previous chapter.

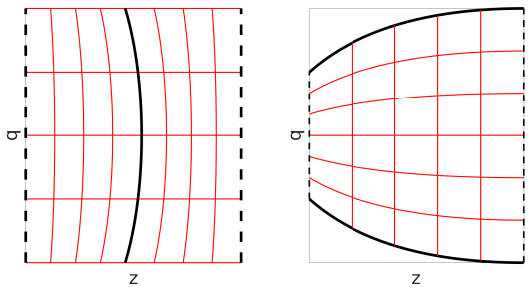


Figure 10.1: Sketch of aligning elements with the optic in the case of a lens (left) and a mirror (right). The optics are indicated with black lines, the screens with dashed lines and the elements with red.

> The mesh must be aligned with the interface in phase space, i.e., the mesh must be such that the interface is equal to the union of a set of edges. Elements that have these edges as part of their boundary are called interface elements.

As observed in Section 9.2, when dealing with mirrors, or any surface where the q-coordinate is described as a function of z, this amounts to a moving mesh. We add here that for lenses, which is to say any surface where the z-coordinate is a function of q, this amounts to integrating on curved screens, which we'll discuss in Section 10.3. Naturally, in either case we'll keep track of the interface elements, so we know exactly which elements are affected by Snell's law.

10.2 Dealing with discontinuous sources

If DG has one flaw, it's that it really only works well for piecewise smooth solutions. Discontinuities can occur at element boundaries, but inside each element the solution should be smooth. If the solution contains shocks or sharp gradients that don't coincide with element boundaries, a lot of things can go wrong. For linear equations, however, exponential convergence can be recovered with perfect shock capturing in post-processing [122]. Fortunately, Liouville's equation is linear, so this line of attack is always open for future research.

Perfect Lambertian sources are discontinuous in phase space. In particular, such sources have a region in phase space where the brightness is constant and

zero everywhere else. There are three approaches one might consider when it comes to such sources. First, use the perfect source, i.e., a discontinuous initial condition, and apply postprocessing to recover the discontinuous solution. Second, move the mesh along with the edge of the distribution, e.g. front tracking. Lastly, approximate a perfect source with a smooth distribution.

In dealing with discontinuous solutions, DG methods produce oscillatory solutions, with the oscillations focussed around where the discontinuity should be. The first option involves expanding the numerical solution in terms of a different basis and finding the location of the discontinuity to sufficient accuracy [142]. The second option involves front tracking, see e.g. [104–108], where the front is defined by the edge of the distribution. Parts of the mesh are then simply attached to rays, in a manner of speaking. It is, however, unclear how to proceed with this technique in the presence of interfaces. The third option is by far the simplest, as we'll demonstrate below.

Any discontinuous function can be approximated arbitrarily close with smooth functions by using mollifiers [71]. A mollifier is a positive infinitely smooth function with compact support and unit integral. Given a mollifier φ_ε, where the parameter ε measures the amount of smoothing, the mollified function g_ε is defined by

$$g_\varepsilon = \varphi_\varepsilon * g, \tag{10.3}$$

with $*$ denoting convolution. As a consequence, the function g_ε is therefore also infinitely smooth for any $\varepsilon > 0$. If we let $\varepsilon \downarrow 0$, g_ε converges to g in a suitable function space such as L^2. One possible mollifier can be constructed with the help of a suitably normalised bump function, i.e.,

$$\psi(x) = \begin{cases} C \exp\left(-\frac{1}{1-x^2}\right) & \text{if } |x| < 1, \\ 0 & \text{otherwise,} \end{cases} \tag{10.4}$$

where $C > 0$ is such that $\int_{-1}^{1} \psi(x)\,\mathrm{d}x = 1$. Note that the bump function is indeed infinitely smooth with compact support. In one dimension, we can define the mollifier as $\varphi_\varepsilon(x) = \frac{1}{\varepsilon}\psi(\frac{x}{\varepsilon})$. Higher-dimensional analogues are easily constructed, for instance by using the radial coordinate $r = |\mathbf{x}|$, i.e., $\varphi_\varepsilon(\mathbf{x}) = \frac{1}{\varepsilon^d}\psi(\frac{r}{\varepsilon})$, with d the dimension. The support of this mollifier is a ball centred on the origin with radius ε.

Thus, mollifying the discontinuous initial condition gives a smooth solution that remains smooth in the interior of all elements, provided the elements are well aligned with any optical interfaces. In practice, a near-perfect source is also simply an arbitrarily close smooth distribution. Therefore, whenever a

Lambertian light source is specified as an initial condition, we'll mollify it first, resulting into a smooth approximation to the light source.

10.3 Integrating to lenses

Suppose we're tasked to solve the light distribution on a given freeform surface. We only require that the freeform lens can be described with some sufficiently smooth function of position q. Since z acts as a time coordinate, this is equivalent to finding the solution to Liouville's equation on a curved space-time slice.

We manage this by making a linear blending transformation in z. For generality, consider two freeform surfaces, $z = \zeta_1(q)$ and $z = \zeta_2(q)$, which may also be constant. Suppose we know the light distribution on $z = \zeta_1(q)$ and we'd like to know the distribution on $z = \zeta_2(q)$. The linear blending is given by

$$z(\lambda, q) = \lambda\zeta_2(q) + (1 - \lambda)\zeta_1(q), \tag{10.5}$$

where $\lambda = 0$ gives surface $z = \zeta_1(q)$ while $\lambda = 1$ gives surface $z = \zeta_2(q)$, see Figure 10.2.

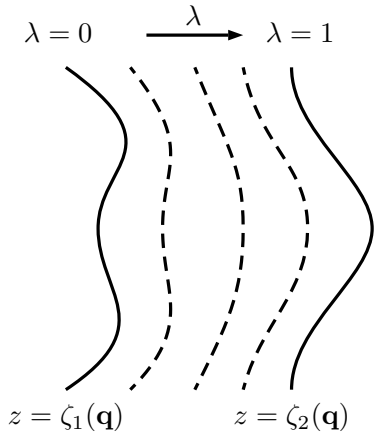


Figure 10.2: Sketch of the linear blending approach, time steps in z are on surfaces that blend from ζ_1 to ζ_2.

Our intent is to find a PDE for ρ in the coordinates (λ, q, p) which we'll then integrate from $\lambda = 0$ to $\lambda = 1$, thereby passing from the one surface to the

other. Let's define $\tilde{\rho}$ as

$$\tilde{\rho}(\lambda, q, p) = \rho\big(z(\lambda, q), q, p\big). \tag{10.6}$$

The transformation rules can be found as

$$\frac{\partial \tilde{\rho}}{\partial \lambda} = \frac{\partial \rho}{\partial z}\frac{\partial z}{\partial \lambda}, \tag{10.7a}$$

$$\frac{\partial \tilde{\rho}}{\partial q} = \frac{\partial \rho}{\partial z}\frac{\partial z}{\partial q} + \frac{\partial \rho}{\partial q}, \tag{10.7b}$$

$$\frac{\partial \tilde{\rho}}{\partial p} = \frac{\partial \rho}{\partial p}. \tag{10.7c}$$

Using these transformation rules, we find that Liouville's equation transforms as

$$\frac{1}{z_\lambda}\frac{\partial \tilde{\rho}}{\partial \lambda} + \frac{\partial h}{\partial p}\left(\frac{\partial \tilde{\rho}}{\partial q} - \frac{1}{z_\lambda}\frac{\partial \tilde{\rho}}{\partial \lambda}\frac{\partial z}{\partial q}\right) - \frac{\partial h}{\partial q}\frac{\partial \tilde{\rho}}{\partial p} = 0, \tag{10.8}$$

where $z_\lambda = \frac{\partial z}{\partial \lambda} = \zeta_2(q) - \zeta_1(q)$. Dropping the tildes in notation, we conclude that ρ satisfies the following PDE in the new coordinates,

$$\alpha(\lambda, q, p)\frac{\partial \rho}{\partial \lambda} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{10.9a}$$

where

$$\alpha(\lambda, q, p) = \frac{1 - \frac{\partial h}{\partial p}\big(\lambda \zeta_2'(q) - (1-\lambda)\zeta_1'(q)\big)}{\zeta_2(q) - \zeta_1(q)}. \tag{10.9b}$$

We can now semi-discretise with DG as outlined in Chapter 7 and integrate with the usual suspects, such as RK methods. This leads to the observation that each point in phase space should be integrated with a different step size. The step size can be read off as

$$\Delta z(\lambda, q, p) = \frac{\Delta \lambda}{\alpha(\lambda, q, p)}, \tag{10.10}$$

where $\Delta\lambda$ is the step size in λ. The CFL condition can now be stated in terms of $\Delta\lambda$, so that for a DG method using polynomial degree N, we have

$$\Delta\lambda \leq \frac{\alpha_{\min}\Delta y_{\min}}{b_{\max}(2N+1)}, \tag{10.11}$$

where $\Delta y_{\min}$ is the minimum size in the mesh, $b_{\max}$ is the maximum velocity in the problem and $\alpha_{\min}$ is the minimum value of the correction factor.

The resulting method for refractive optics is as follows: between any pair of surfaces, we solve (10.8) numerically and if the surface is refractive, we apply Snell's law globally to the solution. This way, we can integrate from surface to surface until the target is reached.

10.4 Results

To test the DG method for Liouville's equation, we once again use a test problem that should by now be quite familiar: the bucket of water problem. We use the bucket of water problem to verify that the convergence properties of the DG spectral element method still apply for Liouville's equation. Apart from this, a spherical lens is also used to test the integration method discussed in the previous section. As no exact solution is known for that problem, a reference solution is found by Monte Carlo ray tracing.

10.4.1 Bucket of water

We again use the bucket of water problem as a test case, the results on both structured and unstructured grids are shown in Figure 10.3. The unstructured grid is rather unrealistic, but it demonstrates the point that DG works well on both types of grids.

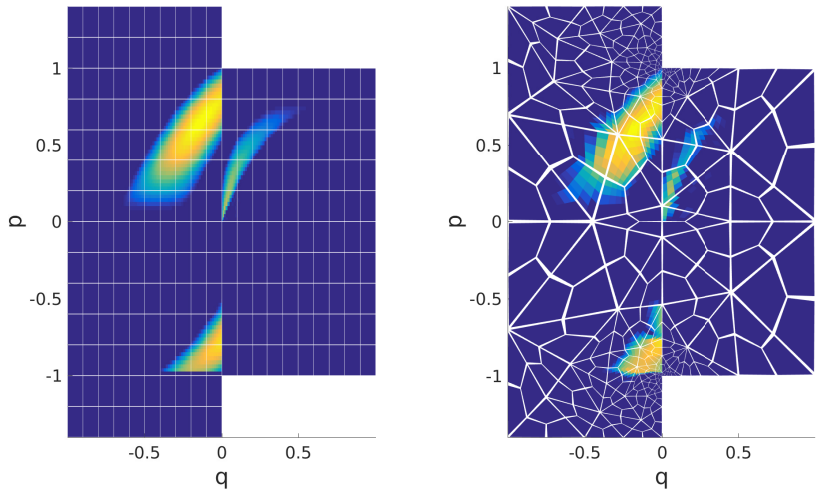
Let's first turn to the question whether or not discontinuous Galerkin methods works as spectacularly as it does on more common problems. Typically, the error behaviour of DG scales as

$$e = \mathcal{O}(\Delta y^{N+1}), \tag{10.12}$$

where Δy a representative mesh size while N is the polynomial degree. In particular, the error should scale as $\mathcal{O}\left(E^{-\frac{N+1}{2}}\right)$ with E the number of elements while we should observe exponential convergence as N is increased.

First, we verify that a DG method of polynomial degree N converges with order $N+1$ in space. To do so, we solve the bucket of water problem using an initial condition that remains smooth throughout the computation. In particular, we use the same smooth initial condition (9.11) introduced in the previous chapter, i.e.,

$$\rho_0(q,p) = \psi\left(\frac{q-q_0}{\lambda_q}\right)\left[\psi\left(\frac{p-p_0}{\lambda_p}\right) + \psi\left(\frac{p-p_1}{\sigma_p}\right)\right]. \tag{10.13}$$



(a) Structured grid of 240 elements with $N = 10$.

(b) Unstructured grid of 372 elements with $N = 5$.

Figure 10.3: Numerical solution to the bucket of water problem on a structured and unstructured grid.

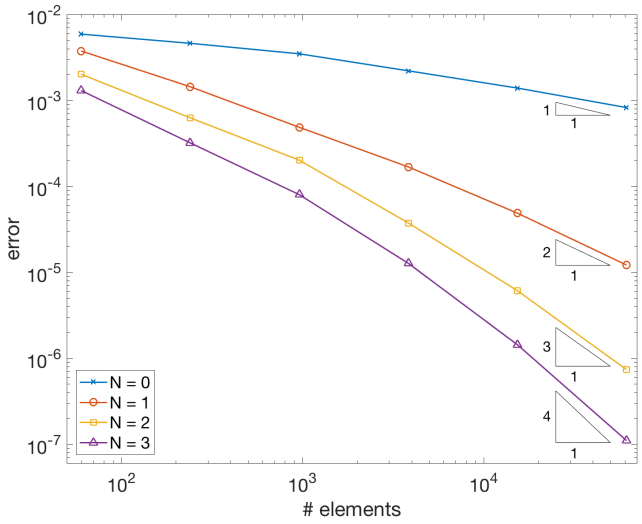


Figure 10.4: Error curves for several discontinuous Galerkin schemes with polynomial order N. Several triangles are drawn to indicate the slopes.

Here ψ is the unnormalised bump function, which is (10.4) with $C = 1$. We use a regular mesh as shown in Figure 10.3a. Integration in z is performed using the standard RK4 integrator, while the z-step is chosen to satisfy the CFL condition. This means the error in z-discretisation will be dominated by the spatial error. As we use nodal DG, the coefficients directly represent the function values on the nodes. Therefore, we simply use the absolute difference averaged over all the nodes as our error, i.e.,

$$e = \frac{1}{E(N+1)^2} \sum_{k=1}^{E} \sum_{i,j=0}^{N} |\rho_{ij}^k(Z) - \rho(Z, q_i, p_j)|, \tag{10.14}$$

where ρ_{ij}^k is the value of the numerical solution on node (i, j) of element k. Some convergence data is shown in Figure 10.4. The results show that DG indeed converges with order N.

Next, we verify exponential convergence as a function of N, keeping the number of elements fixed at 240. We use a constant number of 15000 z-steps so that the z-integration error will be much smaller than the spatial error. Once

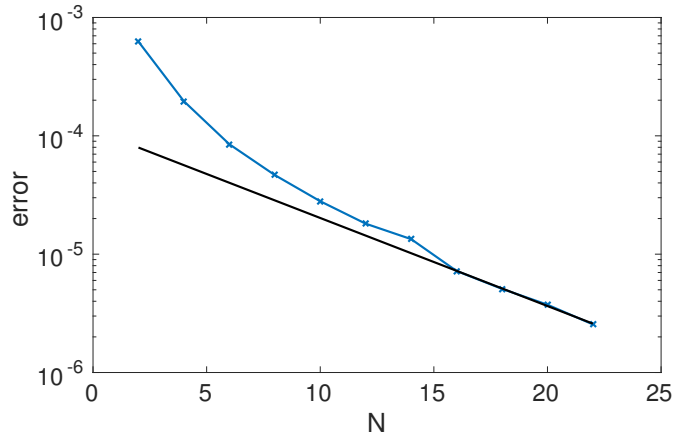


Figure 10.5: Error curve for the discontinuous Galerkin schemes as a function of polynomial order N for the bucket of water problem. The black line is an exponential function to guide the eye.

again, z-integration will not interfere with the convergence test. The results are displayed in Figure 10.5. Clearly, exponential convergence is preserved, even in the presence of an interface.

Finally, we can compare the DG method to Monte Carlo ray tracing for the bucket of water problem. The number of rays is once again chosen to scale quadratically with the number of elements, $N_{\mathrm{RT}} = \frac{1}{10}E^2$. This is the optimal scaling as outlined in Chapter 4. The constant $\frac{1}{10}$ is once again chosen to give reasonable results. If too few rays are used, the situation may occur where not every bin has at least one ray, resulting in a $\mathcal{O}(1)$ error. The constant $\frac{1}{10}$ results in a couple of rays per bin, 6 on average to be precise, for the coarsest mesh. We choose a polynomial order of $N = 4$ and the RK4 integrator, resulting in a fifth-order method in space and a fourth-order method in z. Hence, fourth-order accuracy should be achieved globally. We choose a fifth-order spatial discretisation, since we can see from Figure 10.4 that increasing the polynomial order always decreases the error. The order estimate is computed from the empirical relation

$$e = C_e E^{-\frac{\gamma}{2}}, \tag{10.15}$$

which is valid in the limit of large numbers of elements.

Table 10.1: Convergence results for the bucket of water convergence test using discontinuous Galerkin (DG) with $N = 4$ and ray tracing (RT) versus the number of elements E.

E	e_{DG}	γ_{DG}	t_{DG}	e_{RT}	γ_{RT}	t_{RT}
60	$8.6 \cdot 10^{-4}$		0.1 s	$7.0 \cdot 10^{-3}$		0.012 s
240	$1.9 \cdot 10^{-4}$	2.1	1.1 s	$3.6 \cdot 10^{-3}$	1.0	0.035 s
960	$3.5 \cdot 10^{-5}$	2.5	12.4 s	$1.9 \cdot 10^{-3}$	0.9	1.4 s
3,840	$5.5 \cdot 10^{-6}$	2.7	2 min 33s	$1.0 \cdot 10^{-3}$	0.9	1 min 25 s
15,360	$4.3 \cdot 10^{-7}$	3.7	28 min 55 s	$6.0 \cdot 10^{-4}$	0.8	1 h 30 min 15 s

Both methods, the DG-SEM and first-order ray tracing, were implemented in Fortran and run on a single core of a Dell Optiplex 7010 desktop computer. We choose not to use parallel implementations, since both methods can be heavily parallelised. However, the effects of parallelisation are dependent on the implementation and hardware. Using single-thread implementations therefore gives a more fair comparison. The error of the ray tracer is defined by first interpolating the solution obtained by ray tracing to the nodes of the DG method and then applying (10.14). The results are displayed in Table 10.1.

As expected, DG completely blows ray tracing out of the water. In accordance with theory, this variant of DG exhibits fourth-order convergence, which means the error is dominated by the RK4 method. At the same time, the computation time has a much better scaling. Ray tracing is faster for a small number of rays, but for the largest number of elements, DG is also faster. Comparing errors, however, we see that DG with the smallest number of elements obtains a similar result to ray tracing on the largest number of elements. Hence, DG can do in about a tenth of a second what ray tracing does in an hour-and-a-half, a factor well over $50,000$ faster.

10.4.2 Spherical lens

To demonstrate that the strategy outlined in Section 10.3 works properly, we compute the solution to Liouville's equation for a spherical lens, see Figure 10.6. An ideal lens is defined as a rotation by $\frac{\pi}{2}$ going from one focal plane to the other, so that $q \to p$ and $p \to -q$. Consider the example of a perfectly collimated beam of light, which is focussed into a single point for an ideal lens. Conversely, a point source placed in the focal point of a perfect lens will produce a perfectly collimated beam. Spherical lenses are far from ideal and will therefore exhibit

aberrations and defects.

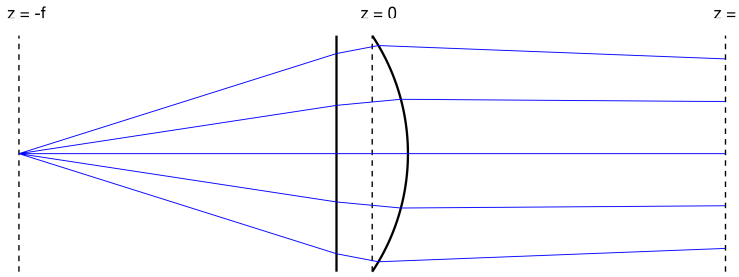


Figure 10.6: Sketch of the situation for the spherical lens case. The initial condition is specified on the $z = -f$ focal plane. Several rays (blue) are sketched as well.

We assume distances in the problem are normalised with some suitable length scale. The lens is a plano-convex with the curved side having a radius of 1.8 units and a refractive index of 1.6, resulting in a focal length of 3 units. The lens has a height of 2 units and no light can get around it. Think of the lens as filling an aperture in a wall of some dark material. The initial condition is chosen to be a normalised Gaußian centred on $(q, p) = (0, 0)$, i.e.,

$$\rho_0(q,p) = \frac{1}{2\pi\sigma_q\sigma_p} \exp\left(-\frac{q^2}{2\sigma_q^2}\right) \exp\left(-\frac{p^2}{2\sigma_p^2}\right), \tag{10.16}$$

where we use the standard deviations $\sigma_q = \frac{1}{10}$ and $\sigma_p = \frac{1}{5}$. The initial condition is plotted in Figure 10.7.

The flat side of the lens is directed towards the light source. This is the "bad" way of using a plano-convex, as fewer aberrations are incurred when the curved side is directed towards the source. However, the point is not to analyse the optical system; rather, we wish to demonstrate the method. Using the lens in this fashion brings out more spectacular results.

We compute the distribution on the focal plane, marked as $z = f$, with both methods. The results are shown in Figure 10.8. As DG is much more accurate, only very few elements are needed, whereas many more elements are needed for ray tracing. As such, we use 10×10 elements with a polynomial degree of 10 for DG, whereas we use 100×100 bins with 10^7 rays for RT. The step size $\Delta\lambda$ is determined by (10.11).

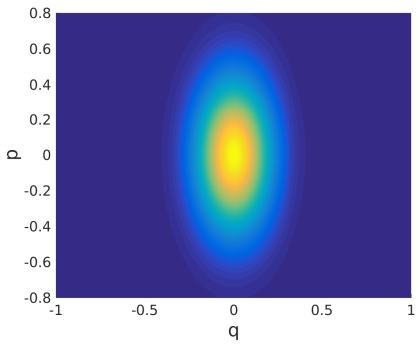


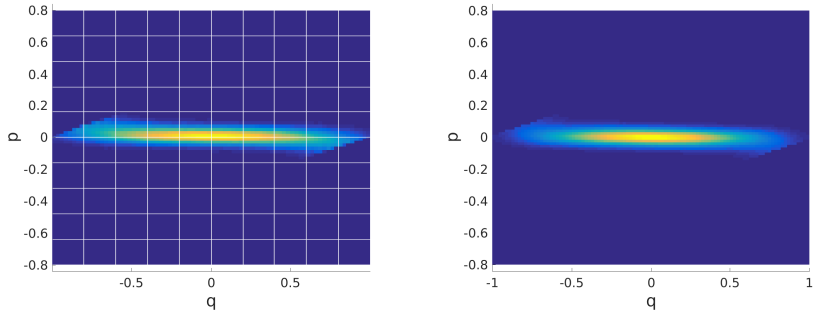
Figure 10.7: Initial condition for the spherical lens problem.

As becomes clear from the figure, the two solutions are virtually the same. However, whereas the ray tracing solutions takes 2 minutes and 50 seconds to compute, the DG solution takes 23 seconds. Hence, also in this case, DG is faster. Note that the distribution at $z = f$ is not a rotation of the initial condition, which is due to the fact that spherical lenses are not ideal. Hence, all sorts of aberrations can be identified, such as the loss of symmetry and the stretching in the q-direction combined with compression in the p-direction. Furthermore, the finite size of the lens is apparent from the sharp cut-off.

Another interesting observation to mention is due to the stretching and compressing effect. As we've already seen in Chapter 8, the gradient of a distribution undergoing an optical transformation is also affected. In this example, the gradient in the p-direction becomes steeper whereas the gradient in the q-direction becomes shallower. The maximum gradient that can be represented by DG is limited by both the polynomial degree and the typical mesh size. As such, the stability of the DG method when applied to lenses depends on both. The polynomial degree needs to be sufficiently rich or the mesh size needs to be sufficiently small.

Illuminance comparison

As DG is based on quadrature rules, it's quite easy to compute integrals of the numerical solution and thereby obtain the luminous intensity and illuminance distributions. As a reminder, luminous intensity is the luminous power per



(a) DG with $E = 100$ and $N = 10$. (b) RT with $E = 10^4$ and 10^7 rays.

Figure 10.8: Numerical solution of a spherical lens with a Gaußian as input on the focal plane.

solid angle and illuminance is the luminous power per unit area. In Chapter 4, we've also discussed that either can be found by integrating out the unwanted dimension in phase space, with intensity being the q-integral of the brightness ρ and illuminance being the p-integral. For the other Liouville solvers we've discussed, upwind and active flux, these lower-dimensional distributions can of course be computed as well from the numerical solution. However, DG lends itself exceptionally well to this purpose.

Adopting the notation from Section 4.3.1, we let F be the number of bins in one dimension and use $F \times F$ elements on phase space. Besides from a DG-SEM, we'll use a second-order low-dimensional ray tracer discussed in Section 4.3.1 to compute the illuminance distribution. As argued, the correct bin can be computed in constant time provided the bin size is constant. To this effect, we define the set of points Q_j, which will serve as the bin edges, as

$$Q_j = (j-1)\Delta q - 1, \quad j = 1, \dots, F+1, \tag{10.17}$$

with $\Delta q = \frac{2}{F}$. The bins themselves are then defined by $B_j = [Q_j, Q_{j+1}]$ and the midpoints are called $q_j = \frac{Q_j + Q_{j+1}}{2}$ for $j = 1, \dots, F$. As explained in Section 4.3.1, the solution found by ray tracing converges with second-order accuracy on the midpoint provided we use $N_{\text{RT}} \sim F^5$. We therefore define the error on the midpoints, i.e.,

$$e = \frac{1}{F} \sum_{j=1}^{F} |E_j - E(q_j)|, \tag{10.18}$$

where $E(\cdot)$ is the reference solution and E_j is the numerical solution at midpoint ${q_j}^1$.

The solution to Liouville's equation is computed on a mesh of $F \times F$ elements and we choose a polynomial order of $N = 2$ and the RK4 method, resulting in a third-order method. The reference solution is computed on a grid of 300×300 elements using a DG method with $N = 4$, also integrated with the RK4 method. We'll compute the illuminance distributions using both methods for 25, 50, 100 and 200 elements. Therefore, the reference solution provides a higher-order solution on a finer grid than either method. To compute the error, the reference solution and the numerical solution itself are first interpolated to the cell midpoints q_j. As an initial condition, we now take a Gaußian centred on $(0, 0)$ with standard deviations $\sigma_q = \frac{1}{5}$ and $\sigma_p = \frac{1}{10}$.

Both methods are, as per usual, run on a single core of a Dell Optiplex 7010 desktop computer. The number of rays was fixed as $N_{\mathrm{RT}} = \frac{F^5}{8.7}$, which was chosen so that both methods start out with roughly the same error on the coarsest mesh. Coincidentally, both methods also start out with roughly the same computation time. The results are displayed in Table 10.2. The first two computations are more or less comparable in terms of computation time, but we must point out that the DG-SEM already achieves an error that's smaller. As we increase the number of bins, ray tracing takes more time and gives a worse numerical solution. For the final computation, DG-SEM takes roughly 11.5 times less time than ray tracing while achieving an accuracy that's about 17 times better.

Table 10.2: Results for DG-SEM versus low-dimensional ray tracing.

F	e_{DG}	γ_{DG}	t_{DG}	e_{RT}	γ_{RT}	t_{RT}
25	$8.2 \cdot 10^{-3}$		0.564 s	$8.1 \cdot 10^{-3}$		0.534 s
50	$9.9 \cdot 10^{-4}$	3.05	14.4 s	$2.7 \cdot 10^{-3}$	1.60	13.9 s
100	$1.3 \cdot 10^{-4}$	2.97	2 min 11 s	$6.8 \cdot 10^{-4}$	1.98	6 min 23 s
200	$1.0 \cdot 10^{-5}$	3.64	19 min 12 s	$1.7 \cdot 10^{-4}$	1.99	3 h 40 min 33 s

The results clearly demonstrate that solving Liouville's equation using DG-SEM can also compete with low-dimensional ray tracing. We must point out, as we did in Chapter 4, that the brightness ρ provides a complete description of light distribution, as it covers both positions and angles. Low-dimensional ray tracing, on the other hand, provides an incomplete picture, even if we compute

[1] In this small section, the symbol E is reserved for the illuminance.

both illuminance and luminous intensity. Hence, solving Liouville's equation through the DG-SEM provides complete information in roughly the same time or faster and at higher accuracies than low-dimensional ray tracing, which provides deficient information.

10.5 Onward to applications

The DG method is the most efficient method we've considered so far for solving Liouville's equation. As such, this opens up new prospects for applications. For instance optimal design, which is the topic of the next part of this thesis. This approach uses iterative improvements to optical systems where each iteration requires the solution of Liouville's equation. Therefore, a fast and high-order Liouville solver is needed as the basic workhorse. Fortunately, we're now in possession of such a method.

Chapter 11

Summary of Liouville solvers

Here, we briefly summarise the performance of the Liouville solvers compared to the current industry standard, which is Monte Carlo ray tracing. The benchmark test throughout this work has been our favourite toy problem, the bucket of water. For a detailed discussed of the theoretical scaling arguments, see Chapter 4.3.

In short, a first-order ray tracer requires a number of rays that scales quadratically with the number of bins, i.e., $N_{\mathrm{RT}} \sim E^2$. Ordinarily, a linear search is also needed to find the correct bin, leading to a time scaling of $t \sim E^3$. A first-order error on two-dimensional phase space means $e_{\mathrm{RT}} \sim E^{-1/2}$, so that the time-to-error scaling results in

$$e_{\mathrm{RT}} = \mathcal{O}\left(t^{-1/6}\right). \tag{11.1}$$

For a Liouville solver of order γ, meaning $e_{\mathrm{L}} \sim E^{-\gamma/2}$, we derived the time scaling $t \sim E^2$. Hence, the time-to-error scaling results in

$$e_{\mathrm{L}} = \mathcal{O}\left(t^{-\gamma/4}\right). \tag{11.2}$$

The order γ depends on the details of the scheme, which in general will affect the constants of the scaling as well. In this work, we've seen a first-order method, $\gamma = 1$, in the upwind solver of Chapter 8. A third-order method, $\gamma = 3$, was explored with the active flux scheme in Chapter 9. Finally, we've seen a

variable-order scheme with the discontinuous Galerkin method of Chapter 10, of which we've investigated the fourth-order option, $\gamma = 4$. We've compiled all the convergence tests of the previous three chapters in Figure 11.1.

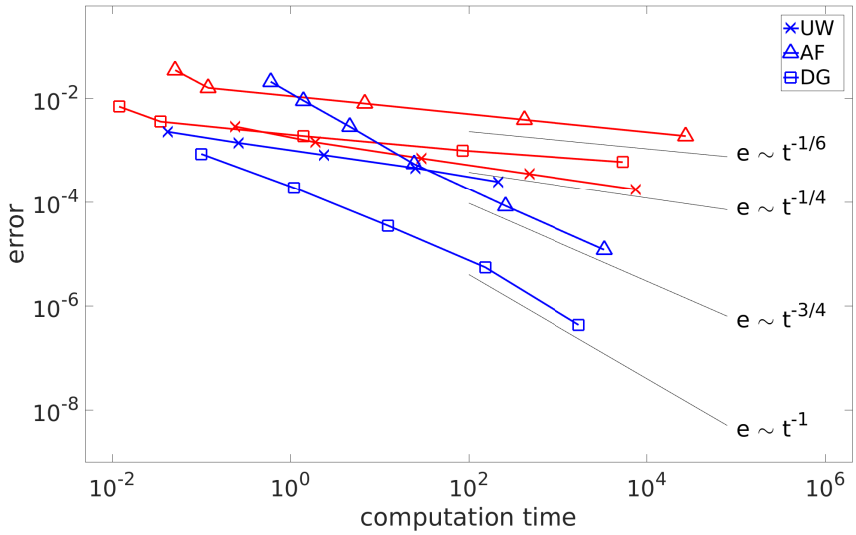


Figure 11.1: Compilation of all the convergence tests between ray tracing (red) and the Liouville methods (blue) on phase space.

The figure also shows three sets of ray tracing data, as the details of the implementation of each comparison are slightly different. For instance, the upwind (UW) solver was implemented in Matlab, while the others are implemented in Fortran. Moreover, the upwind method works on a regular grid, which means the correct element can be found in constant time, resulting in a slightly more favourable $e \sim t^{1/4}$. Additionally, the ray tracer that we've used to compare the active flux (AF) scheme to is implemented on triangular bins, while discontinuous Galerkin (DG) and its ray tracer work on quadrilateral elements/bins. Evidently, more effort is required when using triangular elements to identify the correct bin.

Figure 11.1 shows several interesting things, first among which is that the theoretical scaling arguments were essentially correct. The scaling arguments are based on asymptotic behaviour, so especially for larger computations they hold up. For smaller computations, we see a bit of a departure from the the-

oretical line. The second important thing to note about the figure is that the Liouville solvers are *always better*. In particular, DG proves to be superior giving the lowest error for a fixed computation time. Ray tracing can compete with the active flux scheme for small to medium computations. However, both the upwind method and the DG method are better over the whole range compared to their ray-tracing counterparts.

In Chapter 10, we've also compared the DG Liouville solver with low-dimensional ray tracing. The set-up was a single lens, computing the illuminance on the focal plane. The source was positioned on the other focal plane. We denoted the number of bins in the low-dimensional space as F. Second-order ray tracing, then, the error should scale as $e_{\mathrm{LRT}} \sim F^{-2}$, while we've derived a computation time of $t_{\mathrm{LRT}} \sim F^5$. Hence, low-dimensional ray tracing should give a scaling of $e_{\mathrm{LRT}} \sim t_{\mathrm{LRT}}^{-2/5}$. For comparison we've solved Liouville's equation on an $F \times F$ mesh with a third-order DG method, which results in $e_{\mathrm{DG}} \sim t_{\mathrm{DG}}^{-3/4}$. The results of the convergence test is displayed in Figure 11.2.

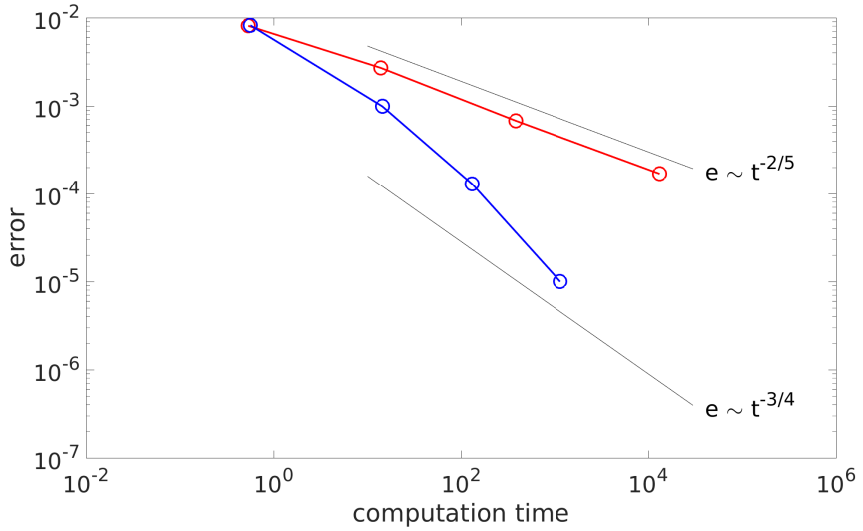


Figure 11.2: Compilation of the convergence tests between low-dimensional ray tracing (red) and DG (blue).

Interesting here is that both error curves more or less start in the same point. Much shorter computation times would probably give meaningless results. We

can therefore conclude that the DG Liouville solver is also better or equally good in the low-dimensional case. Another point is that the theoretical arguments also hold up for low-dimensional ray tracing. The DG Liouville solver, however, seems to beat the theoretical scaling arguments by a slight margin, as the curve is a little steeper. It may be that the line becomes more shallow for even longer computation times.

11.1 Final thoughts on Liouville solvers

The evidence we've compiled in this section suggests that Liouville solvers are indeed remarkably good computational tools for illumination optics. For relatively simple optical problems, they outperform Monte Carlo ray tracing in the computation of phase space distributions as well as the lower-dimensional distributions. The difference becomes especially pronounced for larger numbers of elements/bins. This is because Liouville solvers have a more favourable scaling compared to ray tracing. We expect this behaviour to persist when more complex or practical problems are tackled.

Part IV

Optimal design

In the previous part, we've presented several solvers for Liouville's equation that can be applied to illumination optics problems. All of these solvers exhibit staggering improvements over the current industry standard, Monte Carlo ray tracing. The upwind scheme only provides an improvement in time, while the active flux scheme and the discontinuous Galerkin method are, on top of that, also much more accurate.

DG is the most efficient method we've examined so far, which leads to an interesting possibility that's precluded by ray tracing: applying ideas from optimal control theory. In this context we reinterpret the problem of designing an optic as the problem of designing an open loop controller.

Chapter 12

Basics of optimal control theory

> It's impossible to use the word dynamic in a pejorative sense.
>
> Richard Bellman

We recall here briefly the theory of optimal control, which treats the control of a dynamical system in such a way as to optimise a given objective function. The first optimal control problem was posed by Goddard in 1919, asking the question of how to throttle a rocket such that it reaches maximum altitude [143]. The general theory of optimal control was developed throughout the 1950s by Pontryagin and his group in the Soviet Union concurrently with Richard Bellman in the United States [144, 145]. The time frame being the cold war, both were predominantly working towards military applications[1].

Bellman and Pontryagin were concerned with dynamical systems, i.e.,

$$\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}, \mathbf{u}), \tag{12.1}$$

where $\mathbf{x}$ is the state variable and $\mathbf{u}$ is the control [146]. The governing equation (12.1) is furnished with a suitable initial condition. Bellman's work includes the celebrated Hamilton-Jacobi-Bellman equation, which is solved on the entire

[1] I've always had a soft spot for rocket science, so I was very happy many examples in Pontryagin's book involve missiles and orbital mechanics.

state space. As a consequence, it allows for the design of a controller that can respond to any input. Pontryagin's work allows for single-trajectory optimisation, allowing to optimise for a single specific input. This is more suitable to our application, as we're concerned with Liouville's equation with a fixed initial state. Put differently, we design an optic around a given light source, we don't care for finding all possible optics for all possible light sources, just the one will do. Finally, Liouville's equation is a PDE, so we'll be therefore using the theory of PDE constrained optimisation, also known as optimal control on PDEs [147].

12.1 Constrained optimisation

In constrained optimisation, we're asked to find an optimal value of an objective function while simultaneously satisfying a set of constraints. For example, any business or firm is trying to maximise profit while at times obeying the law. Profit here plays the part of the objective function while legislature provides a set of constraints.

In mathematics, such systems are solved using Lagrange multipliers when dealing with equality constraints [148], while Karush-Kuhn-Tucker multipliers are used when dealing with inequality constraints [149, 150]. In either case, an auxiliary function is formed, called the Lagrangian, which has an extra variable for every constraint. The unconstrained minimum of the Lagrangian is then equal to the constrained minimum of the original objective function. These ideas can be found in any modern textbook on optimisation [151].

The concept of Lagrange multipliers translates rather straightforwardly to dynamical systems, where the constraint is the governing equation (12.1). However, as the constraint depends on time now, the Lagrange multiplier should also depend on time. Suppose the objective function is given by

$$\Psi(\mathbf{x}(T)) + \int_0^T L(t, \mathbf{x}, u)\,\mathrm{d}t,$$

where T is the fixed final time, L is some function that provides a performance measure and Ψ measures the optimality of the final state. The Lagrangian is defined as

$$\mathcal{L}[\mathbf{x}, u, \boldsymbol{\lambda}] = \Psi(\mathbf{x}(T)) + \int_0^T L(t, \mathbf{x}, u)\,\mathrm{d}t - \int_0^T \boldsymbol{\lambda} \cdot (\dot{\mathbf{x}} - \mathbf{f}(t, \mathbf{x}, u))\,\mathrm{d}t, \tag{12.2}$$

where $\boldsymbol{\lambda}$ is the Lagrange multiplier, which is now also a function of time. In the context of optimal control theory, $\boldsymbol{\lambda}$ is called the adjoint or the costate. The Lagrangian $\mathcal{L}$ is a functional, taking several functions as input and attaching a real number to the triple. Abstractly, there's no difference between functionals and functions, both attaching a value to each point in its domain. The difference, roughly speaking, is that a functional takes input values from a function space, while a function takes input values from some spatio-temporal domain. Colloquially, therefore, one could say a functional is a function of functions. To distinguish them, we'll write functionals with script characters and the input in square brackets, while functions will be written using lower case characters with input in round brackets.

Functionals are in a way a generalisation of your run-of-the-mill function, which means we also have to use a suitable generalisation of the derivative. In practice this entails the calculus of variations, while more rigorously we should use the Fréchet and Gâteaux derivatives. Let X be a Banach space with norm $|\cdot|$, then a functional $\mathcal{F} : X \to \mathbb{R}$, is said to be *Fréchet differentiable* at $f \in X$ if there exists some $\ell \in X^*$, with X^* the space of bounded linear functionals operating on X, so that

$$\mathcal{F}[f + \zeta] = \mathcal{F}[f] + \ell[\zeta] + o(|\zeta|) \quad \text{as } |\zeta| \to 0, \tag{12.3}$$

for arbitrary $\zeta \in X$ [13, 14]. The bounded linear functional ℓ is the Fréchet derivative of $\mathcal{F}$ at f. Compare this with the linearisation of a function f around $\mathbf{x}$, i.e., $f(\mathbf{x} + \mathbf{y}) = f(\mathbf{x}) + \nabla f(\mathbf{x}) \cdot \mathbf{y} + o(\|\mathbf{y}\|)$ as $\|\mathbf{y}\| \to 0$. Clearly, the Fréchet derivative is defined by a linearisation for small ζ. The Gâteaux derivative is slightly more general and is defined as follows. A functional $\mathcal{F}$ is said to be *Gâteaux differentiable* if there exists some $\ell \in X^*$ so that

$$\mathcal{F}[f + \varepsilon\zeta] = \mathcal{F}[f] + \varepsilon\ell[\zeta] + o(\varepsilon) \quad \text{as } \varepsilon \to 0. \tag{12.4}$$

The bounded linear functional ℓ is now called the Gâteaux derivative of $\mathcal{F}$ at f. From (12.4), we find the Gâteaux derivative as

$$\mathcal{F}'[f, \zeta] = \lim_{\varepsilon \to 0} \frac{\mathcal{F}[f + \varepsilon\zeta] - \mathcal{F}[f]}{\varepsilon} = \frac{\mathrm{d}}{\mathrm{d}\varepsilon} \mathcal{F}[f + \varepsilon\zeta]\Big|_{\varepsilon=0}. \tag{12.5}$$

If the Fréchet derivative exists, it's equal to the Gâteaux derivative. However, it is possible that the Gâteaux derivative exists without the Fréchet derivative being defined. The derivative of a functional is also called the first variation, with higher derivatives of course identified with higher variations. A useful

special case to remember is if $\mathcal{F}$ is linear, so that the Gâteaux and Fréchet derivatives agree upon

$$\mathcal{F}'[f,\zeta] = \lim_{\varepsilon\to 0}\frac{\mathcal{F}[f+\varepsilon\zeta]-\mathcal{F}[f]}{\varepsilon} = \lim_{\varepsilon\to 0}\frac{\mathcal{F}[\varepsilon\zeta]}{\varepsilon} = \lim_{\varepsilon\to 0}\mathcal{F}[\zeta] = \mathcal{F}[\zeta]. \tag{12.6}$$

Let's now also assume that X is rigged with an inner product $\langle\cdot,\cdot\rangle$, so that it's in fact a Hilbert space. In this case, we can associate a unique element from X with the Fréchet or Gâteaux derivatives by the Riesz Representation Theorem [152]. Indeed, we have that $\mathcal{F}'[f,\zeta]$ is a bounded linear functional in ζ, so that there exists a unique element $g \in X$ so that

$$\mathcal{F}'[f,\zeta] = \langle \zeta, g\rangle\,. \tag{12.7}$$

The element g is called the gradient of $\mathcal{F}$ at f with respect to the Hilbert space. We can again compare this with a Taylor series of an ordinary function, so that this is indeed a natural generalisation of the familiar gradient operator. Here, we'll be using L^2 as our Hilbert space of functions [71]. The gradient in this case, which we'll denote $\frac{\delta\mathcal{F}}{\delta f}$, is therefore called the L^2-gradient[2]. As a final note, a functional may naturally depend on several input functions. The straightforward generalisation is to simply calculate the derivative with respect to each input function independently, much like computing a gradient. In such cases, it's of course also possible to calculate the Gâteaux derivative only with respect to one input function, much like partial differentiation. Our notation will be to attach a subscript to the derivative to signify which input function is varied. The subscript notation will be used, for instance, in Section 12.3.1.

After this short intermezzo concerning derivatives of functionals, we'll get back to finding an optimiser to the control problem. To find a constrained minimum of the objective, all of the derivatives of the Lagrangian $\mathcal{L}$ with respect to the input functions should vanish. For (12.2), this will lead to two ODEs, the original one for $\mathbf{x}$ (12.1), one for the costate $\boldsymbol{\lambda}$ and an algebraic equation for u. Thus, the necessary optimality system takes the form of a differential-algebraic system of equations [153].

The system can be neatly collected by introducing the *control Hamiltonian*[3], i.e.,

$$H\big(t,\mathbf{x}(t),u(t),\boldsymbol{\lambda}(t)\big) = L(t,\mathbf{x},u) + \boldsymbol{\lambda}\cdot\mathbf{f}(t,\mathbf{x},u). \tag{12.8}$$

[2]There are various notations used in the literature, for instance $\operatorname{grad}_{L^2}\mathcal{F}$, which I find positively abhorrent. The notation I use here is borrowed from physics and I think it's much more suggestive.

[3]Unfortunately, this auxiliary function is also called a Hamiltonian, I won't use it very much, but to distinguish the two I'll call this one the control Hamiltonian.

We'll suppress the time dependence in the input functions of the control Hamiltonian for brevity. In the context of optimal control, the quantity $\boldsymbol{\lambda}$ is called the adjoint variable or costate. The Lagrangian can now be rewritten using the control Hamiltonian, yielding

$$\mathcal{L}[\mathbf{x}, u, \boldsymbol{\lambda}] = \Psi(\mathbf{x}(T)) + \int_0^T H(t, \mathbf{x}, u, \boldsymbol{\lambda}) - \boldsymbol{\lambda} \cdot \dot{\mathbf{x}} \, \mathrm{d}t. \tag{12.9}$$

The Gâteaux derivative of the Lagrangian (12.9) in the direction $(\delta\mathbf{x}, \delta u, \delta\boldsymbol{\lambda})$ is given by

$$\mathcal{L}'[\mathbf{x}, u, \boldsymbol{\lambda}, \delta\mathbf{x}, \delta u, \delta\boldsymbol{\lambda}] = \int_0^T \left(\frac{\partial H}{\partial \boldsymbol{\lambda}} - \dot{\mathbf{x}} \right) \cdot \delta\boldsymbol{\lambda} + \frac{\partial H}{\partial \mathbf{x}} \cdot \delta\mathbf{x} - \boldsymbol{\lambda} \cdot \delta\dot{\mathbf{x}} + \frac{\partial H}{\partial u} \delta u \, \mathrm{d}t. \tag{12.10}$$

Since we assume $\mathbf{x}(0)$ is prescribed by an initial condition, the variation of $\mathbf{x}$ must vanish at $t = 0$, hence $\delta\mathbf{x}(0) = \mathbf{0}$. Integration by parts in time for the term $\lambda \cdot \delta\dot{\mathbf{x}}$ therefore only results in a boundary term at $T = 0$, i.e.,

$$\begin{aligned} \mathcal{L}'[\mathbf{x}, u, \boldsymbol{\lambda}, \delta\mathbf{x}, \delta u, \delta\boldsymbol{\lambda}] = \int_0^T \left(\frac{\partial H}{\partial \boldsymbol{\lambda}} - \dot{\mathbf{x}} \right) \cdot \delta\boldsymbol{\lambda} + \left(\frac{\partial H}{\partial \mathbf{x}} + \dot{\boldsymbol{\lambda}} \right) \cdot \delta\mathbf{x} + \frac{\partial H}{\partial u} \delta u \, \mathrm{d}t \\ + \left(\frac{\partial \Psi}{\partial \mathbf{x}}\big(\mathbf{x}(T)\big) - \boldsymbol{\lambda}(T) \right) \cdot \delta\mathbf{x}(T). \end{aligned} \tag{12.11}$$

In this form, the L^2-gradient of the Lagrangian can be read off easily. Requiring that the first variation vanishes is a necessary condition for an extremum, just as with ordinary functions. Sufficient conditions can be obtained by investigating the second variation.

The name control Hamiltonian is actually quite apt, as the system can be compiled into a Hamiltonian system with one extra algebraic constraint, i.e.,

$$\dot{\mathbf{x}} = \frac{\partial H}{\partial \boldsymbol{\lambda}} = \mathbf{f}(t, u, \mathbf{x}), \tag{12.12a}$$

$$\dot{\boldsymbol{\lambda}} = -\frac{\partial H}{\partial \mathbf{x}} = -\frac{\partial L}{\partial \mathbf{x}} - \boldsymbol{\lambda} \cdot \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \tag{12.12b}$$

$$0 = \frac{\partial H}{\partial u} = \frac{\partial L}{\partial u} + \boldsymbol{\lambda} \cdot \frac{\partial \mathbf{f}}{\partial u}. \tag{12.12c}$$

The costate satisfies a terminal condition given by

$$\boldsymbol{\lambda}(T) = \frac{\partial \Psi}{\partial \mathbf{x}}\big(\mathbf{x}(T)\big). \tag{12.13}$$

In most practical cases, the control input is itself constrained to some range $u_{\min} \leq u \leq u_{\max}$. Think, for instance, about driving a car, you can only press down the accelerator so much. In optics, there's a limited range of refractive indices, as one cannot go lower than 1 and not much higher than 3. In such a case, the conditions for an optimum have to be slightly adjusted.

When dealing with constrained controls, it turns out that the control Hamiltonian is the key to finding an optimal trajectory, i.e., the algebraic constraint in (12.12) is replaced by

$$H(t, \mathbf{x}^*, u^*, \boldsymbol{\lambda}^*) \leq H(t, \mathbf{x}^*, u, \boldsymbol{\lambda}^*), \tag{12.14}$$

for all admissible u and all $t \in [0, T]$. Here, quantities with an asterisk optimise the objective function. This statement is known as *Pontryagin's minimum principle* and it can be rigorously proved, see any of the aforementioned textbooks. Roughly speaking, the control Hamiltonian contains all terms that depend on u in (12.9), so that the minimum in u is obtained if H is minimal.

For PDEs, there are two approaches possible: remain in the more difficult setting of PDEs and work out the theory there, or discretise the Lagrangian and then use the more familiar optimisation theory on functions. Rather unimaginatively, these two approaches are known as optimise-then-discretise and discretise-then-optimise, respectively. Here, we'll use the first option and derive the optimality system in the PDE setting, as this allows more freedom in the discretisation. Looking back to Section 10.3 and considering that we'd like to optimise lens-like surfaces, more flexibility in the discretisation will be advantageous indeed. Moreover, the resulting formulation is independent of which solver is used for the PDEs.

12.2 Optimal control on hyperbolic PDEs

We'll now review the theory of optimal control on hyperbolic PDEs [154]. As it turns out, the theory of dynamical systems can be almost used verbatim, if we allow the inner product to work on a function space, which will be $L^2([0, T] \times \Omega)$. Let's start, however, with a disclaimer: in the following, we provide a formal derivation of the optimality system by means of the formal Lagrange multiplier method. Tröltzsch points out that this approach should only be used as a

mnemonic, as it's not rigorous even though it does give the right answer [147]. Heeding the cautionary advice, we'll use the Lagrange multiplier method to derive the optimality system, keeping in the back of our minds that only the result is correct[4].

Consider the first-order linear hyperbolic PDE given by

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}(t, \mathbf{x}, u)) = 0, \tag{12.15}$$

which is satisfied in the domain $(0, T] \times \Omega$, where $\mathbf{v}$ is the velocity field subject to the control input u. The application we aim for is Liouville's equation and geometric optics, where we can control the refractive index. The evolution equation is furnished with initial and boundary conditions, and for simplicity of presentation we assume that the boundary flux $\rho\mathbf{v}$ does not depend on u. Suppose next that we have an objective functional that we wish to optimise, given by

$$\Psi\big[\rho(T)\big] + \int_0^T \int_\Omega L(t, \mathbf{x}, \rho, u)\, \mathrm{d}V\, \mathrm{d}t, \tag{12.16}$$

where ρ is the solution to (12.15), $\rho(T)$ is shorthand for $\rho(T, \cdot)$ and Ψ is assumed to be a Gâteaux differentiable functional. Note that $\rho(T)$ is free and hence Ψ measures the optimality of the output state, e.g., the L^2-distance with respect to some desired output state. We'll later present two possible choices for this functional.

In many cases, it's necessary to add a regulator term $\frac{\alpha}{2}\|u\|^2$, with $\alpha > 0$, for some suitable norm such as the L^2-norm on $(0, T] \times \Omega$. Constructing the Lagrangian, we obtain

$$\begin{aligned} \mathcal{L}[\rho, u, \varphi] =& \Psi\big[\rho(T)\big] + \int_0^T \int_\Omega L(t, \mathbf{x}, \rho, u) + \tfrac{\alpha}{2} u^2\, \mathrm{d}V\, \mathrm{d}t \\ & - \int_0^T \int_\Omega \varphi\big(\rho_t + \nabla \cdot (\rho \mathbf{v})\big)\, \mathrm{d}V\, \mathrm{d}t, \end{aligned} \tag{12.17}$$

where φ is the Lagrange multiplier, which we'll also refer to as the costate or adjoint. Compare this functional with (12.2). For instance, Ψ is now a

[4]The argument is rather subtle, but it involves selecting the proper function spaces. In the formal derivation, everything is assumed sufficiently smooth.

functional instead of a function, the dynamical system has been replaced with a PDE and the Euclidean inner product has been interchanged with the $L^2(\Omega)$ inner product. Entirely similar to optimal control on dynamical systems, we now look for an unconstrained minimum of $\mathcal{L}$. Calculating the variations with respect to u and φ is fairly straightforward, while the variation with respect to ρ requires a bit more effort. Let's therefore investigate the third term in $\mathcal{L}$, i.e.,

$$\begin{aligned} -\int_0^T \int_\Omega \varphi\big(\rho_t + \nabla\cdot(\rho\mathbf{v})\big)\,\mathrm{d}V\,\mathrm{d}t = \int_0^T \int_\Omega \rho\,(\varphi_t + \mathbf{v}\cdot\nabla\varphi)\,\mathrm{d}V\,\mathrm{d}t \\ - \int_\Omega \rho\varphi\,\mathrm{d}V\Big|_0^T - \int_0^T \int_{\partial\Omega} \rho\,\varphi\,\mathbf{v}\cdot\mathbf{n}\,\mathrm{d}S\,\mathrm{d}t, \end{aligned} \tag{12.18}$$

where we've used integration by parts in time and the divergence theorem in space. Setting the left-hand side equal to zero leads to the variational or weak form of the hyperbolic conservation law (12.15). Note that the right-hand side functional is linear in ρ, so that the Gâteaux derivative with respect to ρ can be found using (12.6). We must again note that ρ has a prescribed initial condition, so that its variation must vanish at $t = 0$, i.e., $\delta\rho(0,\cdot) = 0$. Finding the variation of $\mathcal{L}$ is now clear-cut, i.e.,

$$\begin{aligned} \mathcal{L}'[\rho,u,\varphi,\delta\rho,\delta u,\delta\varphi] = \int_0^T \int_\Omega \delta u\left(\frac{\partial L}{\partial u} + \rho\frac{\partial \mathbf{v}}{\partial u}\cdot\nabla\varphi + \alpha u\right)\mathrm{d}V\,\mathrm{d}t \\ - \int_0^T \int_\Omega \delta\varphi\,(\rho_t + \nabla\cdot(\mathbf{v}\rho))\,\mathrm{d}V\,\mathrm{d}t \\ + \int_0^T \int_\Omega \delta\rho\left(\varphi_t + \mathbf{v}\cdot\nabla\varphi + \frac{\partial L}{\partial\rho}\right)\mathrm{d}V\,\mathrm{d}t \\ + \int_\Omega \delta\rho(T)\left(\frac{\delta\Psi}{\delta\rho}\big(\rho(T)\big) - \varphi(T)\right)\mathrm{d}V \\ - \int_0^T \int_{\partial\Omega} \delta\rho\,\varphi\,\mathbf{v}\cdot\mathbf{n}\,\mathrm{d}S\,\mathrm{d}t, \end{aligned} \tag{12.19}$$

where $\frac{\delta\Psi}{\delta\rho}$ is the L^2-gradient of Ψ. Again, a necessary condition for a minimum is that the Gâteaux derivative vanishes for all small variations in the input. Naturally, the variation with respect to the Lagrange multiplier leads to the constraint equation (12.15). If the second integral in (12.19) is to vanish for all small variations $\delta\varphi$, we must insist that (12.15) must be satisfied weakly. Likewise, the third integral should vanish, so that φ satisfies

$$\frac{\partial\varphi}{\partial t} + \mathbf{v}\cdot\nabla\varphi = -\frac{\partial L}{\partial\rho}, \tag{12.20}$$

also weakly. Furthermore, the first integral in (12.19) leads to an algebraic equation for the control u, i.e.,

$$u = -\frac{1}{\alpha}\left(\rho\frac{\partial\mathbf{v}}{\partial u}\cdot\nabla\varphi + \frac{\partial L}{\partial u}\right). \tag{12.21}$$

In the case of constrained control input, we again have to formulate the control Hamiltonian and apply Pontryagin's minimum principle (12.14). If we write the inner product on $L^2(\Omega)$ as $\langle\cdot,\cdot\rangle$, the control Hamiltonian can be expressed in the suggestive form of

$$\mathcal{H}[t,\rho(t,\cdot),u(t,\cdot),\varphi(t,\cdot)] = \int_\Omega \tfrac{\alpha}{2}u^2 + L(t,\mathbf{x},\rho,u)\,\mathrm{d}V - \langle\varphi,\nabla\cdot(\rho\mathbf{v})\rangle\,, \tag{12.22}$$

note that the control Hamiltonian is now a function of its first argument but a functional of the others. From now on, we'll suppress the t argument in the input functions of the control Hamiltonian. We can again express the Lagrangian in a form very close to (12.9), i.e.,

$$\mathcal{L}[\rho,u,\varphi] = \Psi[\rho(T)] + \int_0^T \mathcal{H}[t,\rho,u,\varphi] - \left\langle\varphi,\frac{\partial\rho}{\partial t}\right\rangle \mathrm{d}t. \tag{12.23}$$

This also suggests that the optimality system can be written in a form much like (12.12), provided we replace partial differentiation with a partial Gâteaux derivative. As before, this compiles the optimality system neatly, i.e.

$$\frac{\partial\rho}{\partial t} = \frac{\delta\mathcal{H}}{\delta\varphi} = -\nabla\cdot(\rho\mathbf{v}), \tag{12.24a}$$

$$\frac{\partial\varphi}{\partial t} = -\frac{\delta\mathcal{H}}{\delta\rho} = -\mathbf{v}\cdot\nabla\varphi - \frac{\partial L}{\partial\rho}, \tag{12.24b}$$

$$0 = \frac{\delta\mathcal{H}}{\delta u} = \alpha u + \rho\frac{\partial\mathbf{v}}{\partial u}\cdot\nabla\varphi + \frac{\partial L}{\partial u}. \tag{12.24c}$$

Whenever control constraints are present, e.g. $u_{\min} \leq u \leq u_{\max}$, the control Hamiltonian should be minimal, so that

$$\mathcal{H}[t, \rho^*, u^*, \varphi^*] \leq H(t, \rho^*, u, \varphi^*), \tag{12.25}$$

which should hold for all admissible u and all $t \in [0, T]$. Quantities marked with an asterisk optimise the objective function.

The costate has to satisfy a terminal condition, the same as in the ODE case. Note that the initial condition for ρ is fixed, so that $\delta\rho(0)$ should be identically zero. However, $\delta\rho(T)$ is free, whence we find that φ must fulfil

$$\varphi(T) = \frac{\delta\Psi}{\delta\rho}\big(\rho(T)\big). \tag{12.26}$$

The boundary conditions for ρ and φ are coupled, for instance if ρ is subject to Dirichlet boundary conditions, it follows that $\delta\rho = 0$ on the boundary and consequently φ is free on the boundary. Contrariwise, if ρ is free on the boundary then $\delta\rho$ is left free and we should have $\varphi = 0$ on the boundary. An interesting case is when ρ is subject to inflow conditions, meaning it's free where the velocity field is outward and specified where the velocity is inward. Therefore, on inflow pieces of the boundary, ρ is specified and φ is free, while on outflow pieces of the boundary, ρ is free and thus φ must vanish.

The optimality system is composed of (12.24) supplied with initial and boundary conditions for ρ, the terminal condition (12.26) and a corresponding set of boundary conditions for φ. Satisfying this system is a first-order necessary condition for an optimum. Sufficient conditions require knowledge of the second variation of the objective functional. The boundary conditions are slightly unusual, especially in time. The state ρ has an initial condition while the costate φ has a terminal condition, which in turn depends on the terminal condition of the state ρ. The costate can therefore be interpreted as travelling backwards in time. Moreover, the terminal condition (12.26) reinforces this interpretation, as the costate then seems to communicate an error backwards.

The PDE for ρ is usually referred to as the forward problem; given an initial condition and a velocity field, find the final state. As such, we'll call the PDE for φ the backward problem, as it propagates backwards in time carrying information on how to improve the objective. This observation also ties in with the boundary conditions. As we've discussed, from the fifth integral in (12.19), it follows that wherever ρ has inflow conditions, φ is free and vice versa. If we interpret φ to travel backward in time, the velocity field appears to reverse. As a consequence, wherever the velocity field is inward, φ experiences a velocity

field that points outward. This naturally leads to the corresponding boundary conditions discussed earlier.

12.3 Solution strategy

Now that an optimality system has been identified, we'll go about solving it numerically. For simplicity, we assume the sufficient conditions for a minimum are satisfied. The optimality system is a two-point boundary value problem with an algebraic constraint and it is, to adopt an understatement, quite hard to solve directly. Our approach is therefore iterative, where we make an initial guess of the control input and update it slightly, preferably in such a way that the new control is closer to optimal.

Essentially, we'll use a descent method on the functional $\mathcal{L}$, for instance a simple gradient descent step, but Newton's method can be used as well. At the very least, then, we need the L^2-gradient of $\mathcal{L}$ with respect to u, given by

$$\frac{\delta \mathcal{L}}{\delta u} = \alpha u + \frac{\partial L}{\partial u} + \rho \frac{\partial \mathbf{v}}{\partial u} \cdot \nabla \varphi. \tag{12.27}$$

Note that this gradient depends on all three inputs of $\mathcal{L}$. As such, given a control input u, we need to compute both ρ and φ to determine the gradient. The solution strategy can therefore be summarised by the following set of instructions:

1. Make an initial guess for the control u.
2. Solve the forward problem (12.15).
3. Solve the backward problem (12.20) with terminal condition (12.26).
4. Compute the gradient (12.27).
5. If the norm of the gradient is sufficiently small, exit. Otherwise, update the control u and go to step 1.

If there are control constraints present such as outlined above, the updated control input should be projected back onto the admissible range. In practice, this just means if the control is greater or smaller than a bound, it's set equal to the bound. Control constraints also entail using a different exit condition, as the norm of the gradient may never go to zero. An alternative is to stop when the relative update to the objective is sufficiently small.

The control u can be updated in various ways, for instance using a simple gradient descent with fixed step size. Another possibility is a Newton-like update, using information from the second variation of the Lagrangian.

12.3.1 Newton minimisation

To achieve fast convergence of the minimisation problem, we prefer to use Newton minimisation instead of a simple gradient descent step. For ordinary optimisation problems, where the inputs are real numbers instead of functions, Newton's method is known to converge quadratically whereas gradient descent converges linearly. Newton's method will take more work per step, but the faster convergence will most likely pay off in fewer steps. In the above section, we've tacitly introduced the reduced objective functional $j(u)$, which is defined by

$$j[u] = \mathcal{L}\big[R_\rho(u), u, R_\varphi(u)\big], \tag{12.28}$$

where R_ρ and R_φ are the solution operators defined by (12.15) and (12.20), respectively. Abstractly, we may consider a function space for the pair (ρ, φ), where we're looking for a minimum of the objective on a manifold defined by solutions to the continuity equation (12.15) and the adjoint equation (12.20). The reduced functional provides the objective value on this solution manifold. At the same time, if we're looking for a minimum, we should compute the gradient direction along the manifold. The Gâteaux derivative of j is given by

$$j'[u, \delta u] = \int_0^T \int_\Omega \left(\alpha u + \frac{\partial L}{\partial u} + R_\rho(u) \frac{\partial \mathbf{v}}{\partial u} \cdot \nabla R_\varphi(u) \right) \delta u \, \mathrm{d}V \, \mathrm{d}t, \tag{12.29}$$

which is equal to $\mathcal{L}'$ with $\rho = R_\rho(u)$ and $\varphi = R_\varphi(u)$. In a Newton step, we incorporate information from the second variation of j. The second variation should also be computed along the solution manifold. Straightforwardly computing the

variation j' yields

$$\begin{aligned} j''[u,\delta u,\tau u] = \int_0^T \int_\Omega \tau u\,\delta u \left(\alpha + \frac{\partial^2 L}{\partial u^2} + \rho\frac{\partial^2 \mathbf{v}}{\partial u^2}\cdot\nabla\varphi\right) \\ + \tau u\,\delta\rho\left(\frac{\partial^2 L}{\partial u\partial\rho} + \frac{\partial \mathbf{v}}{\partial u}\cdot\nabla\varphi\right) \\ - \tau u\,\delta\varphi\nabla\cdot\left(\rho\frac{\partial \mathbf{v}}{\partial u}\right)\mathrm{d}V\,\mathrm{d}t \\ + \int_0^T \int_{\partial\Omega} \tau u\,\delta\varphi\,\rho\frac{\partial \mathbf{v}}{\partial u}\cdot\mathbf{n}\,\mathrm{d}A\,\mathrm{d}t. \end{aligned} \tag{12.30}$$

Abstractly, the variations $\delta\rho$ and $\delta\varphi$ should also lie in the tangent space to the solution manifold. Concretely this means the variations $\delta\rho$ and $\delta\varphi$ should satisfy linearised versions of the solution operators. The solution operators are defined by demanding that the partial Gâteaux derivative of $\mathcal{L}$ with respect to ρ and φ vanish for all variations. Linearisation of the solution operators is therefore done by calculating the variations of $\mathcal{L}'_\rho$ and $\mathcal{L}'_\varphi$ and demanding that the result vanishes. Thus, the variation $\delta\rho$ should satisfy

$$0 = -\int_0^T \int_\Omega \delta\varphi\left(\delta\rho_t + \nabla\cdot(\mathbf{v}\delta\rho) + \nabla\cdot\left(\rho\frac{\partial \mathbf{v}}{\partial u}\delta u\right)\right)\mathrm{d}V\,\mathrm{d}t, \tag{12.31}$$

for all variations $\delta\varphi$. Therefore, $\delta\rho$ satisfies

$$\delta\rho_t + \nabla\cdot(\mathbf{v}\delta\rho) = -\nabla\cdot\left(\rho\frac{\partial \mathbf{v}}{\partial u}\delta u\right), \tag{12.32}$$

which is a linear hyperbolic conservation law where δu serves as a source term. The initial condition should be identically zero, as ρ has specified initial conditions. The variation $\delta\varphi$ follows from the variation of $\mathcal{L}'_\rho$, i.e.,

$$\begin{aligned} 0 = \int_0^T \int_\Omega \tau\rho\left(\delta\varphi_t + \mathbf{v}\cdot\nabla\delta\varphi\right) + \tau\rho\,\delta u\left(\frac{\partial \mathbf{v}}{\partial u}\cdot\nabla\varphi + \frac{\partial^2 L}{\partial\rho\partial u}\right) + \tau\rho\,\delta\rho\,\frac{\partial^2 L}{\partial\rho^2}\,\mathrm{d}V\,\mathrm{d}t \\ + \int_\Omega \tau\rho(T)\left(\psi''\delta\rho(T) - \delta\varphi(T)\right)\mathrm{d}V, \end{aligned} \tag{12.33}$$

for all variations $\tau\rho$. Hence, we find that $\delta\varphi$ should satisfy

$$\delta\varphi_t + \mathbf{v}\cdot\nabla\delta\varphi = -\delta u\left(\frac{\partial \mathbf{v}}{\partial u}\cdot\nabla\varphi + \frac{\partial^2 L}{\partial\rho\partial u}\right) - \delta\rho\,\frac{\partial^2 L}{\partial\rho^2}. \tag{12.34}$$

The terminal condition of $\delta\varphi$ is given by

$$\int_\Omega \delta\varphi(T)\,\tau\rho(T)\,\mathrm{d}V = \Psi''[\rho(T),\delta\rho(T),\tau\rho(T)], \tag{12.35}$$

which should hold for all variations $\tau\rho(T)$. We can apply the Riesz Representation Theorem associate a unique function with $\Psi''[\rho(T),\delta\rho(T),\cdot]$, which will depend on both $\rho(T)$ and $\delta\rho(T)$. Thus, to compute the second variation for a given δu we thus need to solve $\delta\rho$ forward in time, after which we can compute $\delta\varphi$ backward in time.

The Newton update δu is defined as the solution of the equation

$$j''[u,\delta u,\tau u] = -j'[u,\tau u], \tag{12.36}$$

which should hold for all variations τu. It's important to note that $\delta\rho$ satisfies a linear hyperbolic PDE that depends linearly on δu, while φ also depends linearly on δu and $\delta\rho$. Thus, we conclude that $j''[u,\delta u,\tau u]$ is linear in δu. Applying the Riesz Represetation Theorem once more, we can associate a unique function on both sides in (12.36) to identify a linear operator on δu. For j', we find the L^2 gradient given by (12.27), while for j'' the result can be read off from (12.30), call it $A(u,\delta u)$. However, $A(u,\cdot)$ is partially defined in terms of solutions to PDEs, namely (12.32) and (12.34). Therefore, the linear operator $A(u,\cdot)$ cannot be formulated in terms of a matrix, even after discretisation of all the PDEs involved. Hence, if we are to solve (12.36), we'll need to employ matrix-free linear solvers. Many Krylov-subspace methods have a matrix-free version, like conjugate gradients and generalised minimal residual (GMRES) [155].

12.4 Choice of Ψ

There are many possible choices for Ψ, the functional that measures the optimality of the final state of ρ. Commonly, we'd like to achieve some desired output state $\rho^\star \in L^2(\Omega)$, in which case we want to somehow measure the distance between the two. An obvious choice is to use the L^2-distance, which immediately

fits the bill, i.e., it vanishes if the distributions are equal and gives a positive real number otherwise. Thus, one possible choice for $\Psi : L^2(\Omega) \to \mathbb{R}$ is simply

$$\Psi[f] = \tfrac{1}{2} \int_\Omega (f - \rho^\star)^2 \, \mathrm{d}V, \tag{12.37}$$

where of course f should be an element of $L^2(\Omega)$. The factor $\frac{1}{2}$ doesn't really do much besides from slightly simplifying the Gâteaux derivative, which is given by

$$\Psi'[f, \delta f] = \int_\Omega (f - \rho^\star) \delta\rho \, \mathrm{d}V. \tag{12.38}$$

As a result, according to (12.26) the costate has a terminal condition

$$\varphi(T) = \rho(T) - \rho^\star. \tag{12.39}$$

The second derivative, or second variation if you prefer, is given by

$$\Psi''[f, \delta f, \tau f] = \int_\Omega \delta f \, \tau f \, \mathrm{d}V. \tag{12.40}$$

By (12.35) and the Fundamental Lemma of Calculus of Variations, the terminal condition for the variation of the costate becomes

$$\delta\varphi(T) = \delta\rho(T). \tag{12.41}$$

The L^2-distance metric has several advantages, predominantly its simplicity. However, it also has some drawbacks which are inherited from the L^2-norm. For instance, very spiky functions with small supports have a small L^2-norm, while smoothness is usually a desirable property in most PDE solutions. Furthermore, consider the situation where the supports of ρ and $\rho^\star$ are completely disjoint. Any such configuration will have the same norm. As such, the L^2-distance as our measure of optimality will "focus its efforts" on local optimisation, where the supports are already overlapping.

When the properties of the L^2-distance fall short of producing the desired outcome, alternative metrics must be considered. Another suggestion is to interpret ρ and $\rho^\star$ as distributions and to minimise the difference in moments[5].

[5]This metric was suggested to me by Oliver Tse after he saw my presentation on CASA day.

Let m_k be a set of monomials for $k = 1, \ldots, K$, e.g. x, y, x^2, y^2, etc. and let $\mathcal{M}_k : L^1(\Omega) \to \mathbb{R}$ be the corresponding set of functionals that return the moments, so that

$$\mathcal{M}_k[f] = \int_\Omega m_k f \, \mathrm{d}V, \tag{12.42}$$

where we assume that Ω is bounded, so that all monomials are bounded. Clearly, $\mathcal{M}_k$ is a linear functional for all $k = 1, \ldots, K$. The difference of the moments is minimised by setting

$$\Psi[f] = \tfrac{1}{2} \sum_{k=1}^{K} \left(\mathcal{M}_k[f] - \mathcal{M}_k[\rho^\star] \right)^2 = \tfrac{1}{2} \sum_{k=1}^{K} \left(\mathcal{M}_k[f - \rho^\star] \right)^2 . \tag{12.43}$$

When two distributions have equal moments, they are equal provided Ω is bounded [156], which it indeed is by assumption. If Ω happens to be an unbounded space, this is no longer true [157]. Furthermore, when the two distributions are not equal, the measure will return a positive real number, thus resulting in a positive definite objective function. Note, however, that it's not a norm or length since the triangle inequality isn't satisfied. The first variation of Ψ is given by

$$\Psi'[f, \delta f] = \sum_{k=1}^{K} \int_\Omega \left(\mathcal{M}_k[f - \rho^\star] \right) m_k \, \delta f \, \mathrm{d}V. \tag{12.44}$$

We can see that the terminal condition for the costate becomes

$$\varphi(T) = \sum_{k=1}^{K} \mathcal{M}_k[\rho(T) - \rho^\star] m_k, \tag{12.45}$$

which follows from (12.26). Since the $\mathcal{M}_k$ are linear, the second variation of Ψ is in this case given by

$$\Psi''[f, \delta f, \tau f] = \sum_{k=1}^{K} \int_\Omega \mathcal{M}_k[\delta f] m_k \, \tau f \, \mathrm{d}V, \tag{12.46}$$

where we've used (12.6). The variation of the costate $\delta\varphi$ will therefore have the terminal condition

$$\delta\varphi(T) = \sum_{k=1}^{K} \mathcal{M}_k[\delta\rho(T)] m_k, \tag{12.47}$$

where we've applied (12.35) and the Fundamental Lemma of Calculus of Variations. With this metric, we sacrifice some of the uniqueness by using a finite number of moments, but any $K > 0$ will in fact provide a suitable positive definite objective function. Moreover, this measure doesn't suffer from the shortcomings of the L^2-distance. A slight drawback is that this measure is focussed on the global shape of the distribution for small K, ignoring tiny details that might be important in some cases. Another advantage is that (12.45) simply defines a polynomial. If K isn't too large, the costate will therefore be much smoother compared to using the L^2-distance metric.

The connection between the two suggestions is provided by Parseval's identity [152], which states that on a separable Hilbert space, the following holds

$$\sum_{k=1}^{\infty} |\langle x, e_k \rangle|^2 = \|x\|^2, \tag{12.48}$$

if e_k is an orthonormal basis. In our case, we see that if we choose m_k to be an orthonormal basis of $L^2(\Omega)$ and let $K \to \infty$, (12.43) becomes equal to (12.37). The monomials form a basis of $L^2(\Omega)$, however, they aren't orthonormal at all. In fact, monomials make up a rather poor basis since they'll increasingly start to look for each other for large k. The difference between x^{100} and x^{101} is hard to see by eye. However, if K is small, they're distinguishable enough to provide a good objective. A possible hybrid between the two suggestions would be to take m_k an orthonormal basis, but use a finite K.

In general, which measure is most appropriate will depend on the problem and the desired outcome. For instance, lighting a wall uniformly will probably benefit more from a global measure, so that (12.43) would be preferred. If some detailed projection is to be created, small details of the light distribution could be important, so that (12.37) is more fitting. Designing the objective function is an art in itself, a human might simply say that they want a distribution to closely resemble another, while judging somehow by eye what's close. However, to mathematise[6] the eyeball norm is often a challenge.

12.5 A note on optical interfaces

Our exegesis of the theory so far assumed that everything is reasonably smooth. Clearly, when optical interfaces are present this will not be true. In that case,

[6] I first saw this word in Mattheij, Rienstra and Ten Thije Boonkkamp in [70]. The definition they use is "the process of translating a real-world problem into mathematical terms." I've used a slightly expanded meaning: to express in mathematical terms.

we'll have to treat the two domains defined by the interface separately, while the shape of the interface poses another control input. Optical interfaces therefore necessarily complicate matters slightly, which is why we'll first treat smoothly varying refractive index fields, GRIN optics for short. This will be our project in the next chapter.

Chapter 13

Optimising GRIN optics

> 'All right,' said the Cat; and this time it vanished quite slowly, beginning with the end of the tail, and ending with the grin, which remained some time after the rest of it had gone.
>
> Lewis Caroll
> Alice in Wonderland

In this chapter, we'll make a start with applying the theory of optimal control to Liouville's equation in a geometric optics setting. The crucial insight here is to interpret an optical system as an open loop controller [158]. Indeed, the an optical system can be rightly viewed as a feed-forward controller acting on either many rays simultaneously or the phase space distribution as a whole. Therefore, the dynamical system we keep in mind is the Hamiltonian system of a large collection of rays, or alternatively, Liouville's equation. As a first exploration, we discuss the design of smoothly varying refractive index fields, also called GRIN optics [5].

GRIN is an acronym for gradient index, accentuating the fact that the refractive index field changes smoothly. This as opposed to the more common shaped surfaces delineating regions of constant refractive index. Shaped surfaces are, of course, easier to manufacture than shaped three-dimensional smoothly varying fields. However, 3d-printing may have the potential to produce arbitrary GRIN optics in the future [26–30]. In principle, of course, there's nothing stopping

us from imagining and designing GRIN optics. We formulate the problem as follows: minimise

$$\Psi\big[\rho(Z)\big] + \int_0^Z \int_{\mathcal{P}} L(\rho, u) + \tfrac{\alpha}{2} u^2 \,\mathrm{d}y \,\mathrm{d}z, \tag{13.1}$$

where $\alpha > 0$ is a constant and $\rho(Z)$ is shorthand for $\rho(Z, \cdot)$ and Ψ is a Gâteaux differentiable functional. Furthermore $\mathrm{d}y$ is the volume element on phase space $\mathcal{P}$, so that $\mathrm{d}y = \mathrm{d}q\,\mathrm{d}p$. This functional is exactly the same as we've introduced in the previous chapter, see e.g. (12.16). The minimisation is subject to the constraint that ρ satisfies Liouville's equation, i.e.,

$$\frac{\partial \rho}{\partial z} + \nabla \cdot \big(\rho \boldsymbol{S} \nabla h\big) = 0, \tag{13.2}$$

with initial condition $\rho(0, q, p) = \rho_0(q, p)$. For two-dimensional optical systems, the gradient operator reads $\nabla = (\partial_q, \partial_p)^T$ and the matrix $\boldsymbol{S}$ is given by

$$\boldsymbol{S} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}. \tag{13.3}$$

Finally, the Hamiltonian h is given by

$$h(u, p) = -\sqrt{u^2 - p^2}, \tag{13.4}$$

where u is the refractive index field, which is our control input.

13.1 Application of optimal control theory

The theory exhibited in the previous chapter may be applied almost verbatim, the only difference is that the velocity field does not just depend on u, see (12.15), but also on it's gradient. Therefore, only a minor adjustment in the gradient of the augmented function with respect to u is needed. In particular, since the control input u is embedded in h and only the gradient of h appears in Liouville's equation, we'll need to perform an additional integration by parts.

The augmented functional is given by

$$\begin{aligned}\mathcal{L}[\rho,u,\varphi] =&\Psi\big[\rho(Z)\big] + \int_0^Z \int_{\mathcal{P}} L(t,\mathbf{y},\rho,u) + \tfrac{\alpha}{2}u^2 \,\mathrm{d}y\,\mathrm{d}z \\ &- \int_0^Z \int_{\mathcal{P}} \varphi\big(\rho_z + \nabla\cdot(\rho\boldsymbol{S}\nabla h)\big)\,\mathrm{d}y\,\mathrm{d}z.\end{aligned} \tag{13.5}$$

We'll now calculate the Gâteaux derivative of $\mathcal{L}$. Just as in the previous chapter, the variation with respect to φ will yield the constraint, i.e., Liouville's equation. Variation with respect to ρ will yield the adjoint equation, which is unaltered compared to the previous chapter. As such, the costate satisfies the advective form of Liouville's equation with a source term coming form L, see (12.20) in Section 12.2. Variation with respect to u will give an algebraic equation for the control. The only difference with the theory of the previous chapter is that now the gradient of h appears, while h depends on u. We'll therefore treat the variation with respect to u in more detail.

Recall that the Poisson bracket $\{\cdot,\cdot\}$ is defined as

$$\{\rho,\varphi\} = \nabla\rho\cdot(\boldsymbol{S}\nabla\varphi)\,. \tag{13.6}$$

Whenever φ is sufficiently smooth, e.g. twice differentiable on phase space, we also have

$$\nabla\cdot(\rho\boldsymbol{S}\nabla\varphi) = \{\rho,\varphi\}, \tag{13.7}$$

since

$$\rho\nabla\cdot(\boldsymbol{S}\nabla\varphi) = \rho\left(\frac{\partial^2\varphi}{\partial q\partial p} - \frac{\partial^2\varphi}{\partial p\partial q}\right) = 0. \tag{13.8}$$

Note that $\{\rho,\varphi\} = -\{\varphi,\rho\}$. The Poisson bracket will serve to compactify many expressions in this chapter, though we'll use the other forms when they're more suggestive.

Now we're in a position to calculate the variation of $\mathcal{L}$ with respect to u. Consider the divergence term in $\mathcal{L}$, which we'll call $\mathcal{D}$ for brevity. Integration

by parts yields

$$\begin{aligned}\mathcal{D}[\rho,u,\varphi] &= -\int_0^Z \int_{\mathcal{P}} \varphi\nabla\cdot(\rho\boldsymbol{S}\nabla h)\,\mathrm{d}y \\ &= \int_0^Z \int_{\mathcal{P}} \rho\nabla\varphi\cdot(\boldsymbol{S}\nabla h)\,\mathrm{d}y - \int_0^Z \int_{\mathcal{P}} \varphi\rho(\boldsymbol{S}\nabla h)\cdot\mathbf{n}\,\mathrm{d}S.\end{aligned} \tag{13.9}$$

Next, we transfer the matrix $\boldsymbol{S}$ to $\nabla\varphi$, which incurs a transposition, but $\boldsymbol{S}^T = -\boldsymbol{S}$, so that

$$\mathcal{D}[\rho,u,\varphi] = -\int_0^Z \int_{\mathcal{P}} \nabla h\cdot(\rho\boldsymbol{S}\nabla\varphi)\,\mathrm{d}y\,\mathrm{d}z - \int_0^Z \int_{\partial\mathcal{P}} \varphi\rho(\boldsymbol{S}\nabla h)\cdot\mathbf{n}\,\mathrm{d}S\,\mathrm{d}z. \tag{13.10}$$

We must note here that there's no flux across the p-boundary of phase space, i.e. $p = \pm u$. To prove this, we'll show that the limit of $(\boldsymbol{S}\nabla h)\cdot\mathbf{n}$ vanishes as $p \to u$. First we calculate the normal on the p-boundary of phase space, which is given by

$$\mathbf{n} = \frac{1}{\sqrt{1+u'^2}}\begin{pmatrix}-u'(q)\\ 1\end{pmatrix}. \tag{13.11}$$

Therefore, we find that

$$\begin{aligned}(\boldsymbol{S}\nabla h)\cdot\mathbf{n} &= \frac{1}{\sqrt{1+u'^2}}\left(-\frac{\partial h}{\partial p}u' - \frac{\partial h}{\partial p}\right) \\ &= \frac{1}{\sqrt{1+u'^2}}\left(-\frac{p}{\sqrt{u^2-p^2}}u'(q) + \frac{uu'(q)}{\sqrt{u^2-p^2}}\right) \\ &= \frac{u'}{\sqrt{1+u'^2}}\frac{u-p}{\sqrt{u^2-p^2}}.\end{aligned} \tag{13.12}$$

Passing to the limit of $p \to u$, we see that the normal advection velocity vanishes. This allows us to simplify the boundary term, so that we obtain

$$\mathcal{D}[\rho,u,\varphi] = -\int_0^Z \int_{\mathcal{P}} \nabla h\cdot(\rho\boldsymbol{S}\nabla\varphi)\,\mathrm{d}y + \int_0^Z \int_{P} \rho\varphi\, n_q \frac{p}{h}\,\mathrm{d}S\,\mathrm{d}z\Big|_{q=-1}^{q=1}, \tag{13.13}$$

where $n_q = \pm 1$ is the q-component of the unit normal and P is the momentum space. Now, another integration by parts is needed to isolate h, yielding

$$\begin{aligned}\mathcal{D}[\rho, u, \varphi] = \int_0^Z \int_{\mathcal{P}} h\nabla\cdot(\rho \boldsymbol{S}\nabla\varphi)\,\mathrm{d}y\,\mathrm{d}z + \int_0^Z \int_P \rho\varphi\, n_q \frac{p}{h}\,\mathrm{d}S\,\mathrm{d}z\Big|_{q=-1}^{q=1} \\ - \int_0^Z \int_{\partial\mathcal{P}} h\rho(\boldsymbol{S}\nabla\varphi)\cdot\mathbf{n}\,\mathrm{d}S\,\mathrm{d}z,\end{aligned} \tag{13.14}$$

where we recognise the Poisson bracket of ρ and φ. To ease the presentation, we assume that u is fixed on the boundary, which fixes the value of h on the boundary as well. Moreover, $h = 0$ on the p-boundaries of phase space, i.e., when $|p| = u$, which follows from (13.4). Hence, the variation of h will be zero on the entire boundary of phase space. Finally, we can calculate the variation with respect to u, yielding

$$\mathcal{D}'_u[\rho, u, \varphi, \delta u] = \int_0^Z \int_{\mathcal{P}} \delta u \frac{\partial h}{\partial u}\{\rho, \varphi\}\,\mathrm{d}y\,\mathrm{d}z, \tag{13.15}$$

where both boundary terms vanish in the variation. Our last remark is that $\frac{\partial h}{\partial u} = \frac{u}{h}$.

We've dealt with the difficult part of the Gâteaux derivative of $\mathcal{L}$, the other variations are exactly the same as in the previous chapter. For a detailed derivation we refer the reader to Section 12.2. Therefore, we present the variation of

the Lagrangian without further ado:

$$\begin{aligned}\mathcal{L}'[\rho,u,\varphi,\delta\rho,\delta u,\delta\varphi] = &\int_0^Z \int_{\mathcal{P}} \delta u \left(\alpha u + \frac{\partial L}{\partial u} + \frac{u}{h}\{\rho,\varphi\}\right) \mathrm{d}y\,\mathrm{d}z \\ &+ \int_{\mathcal{P}} \delta\rho(Z) \left(\frac{\delta\Psi}{\delta\rho}\big(\rho(Z)\big) - \varphi(Z)\right) \mathrm{d}y \\ &+ \int_0^Z \int_{\mathcal{P}} \delta\rho \left(\varphi_z + (\boldsymbol{S}\nabla h)\cdot\nabla\varphi + \frac{\partial L}{\partial\rho}\right) \mathrm{d}y\,\mathrm{d}z \\ &- \int_0^Z \int_{\mathcal{P}} \delta\varphi\big(\rho_z + \nabla\cdot(\rho\boldsymbol{S}\nabla h)\big)\,\mathrm{d}y\,\mathrm{d}z. \\ &- \int_0^Z \int_{\partial\mathcal{P}} \delta\rho\varphi(\boldsymbol{S}\nabla h)\cdot\mathbf{n}\,\mathrm{d}S\,\mathrm{d}z.\end{aligned} \tag{13.16}$$

Of course, demanding that the variation with respect to the costate φ vanishes yields Liouville's equation (13.2). Demanding that the second and third integrals in (13.16) vanish, we find the following problem for the costate,

$$\begin{cases} \frac{\partial\varphi}{\partial z} + (\boldsymbol{S}\nabla h)\cdot\nabla\varphi = -\frac{\partial L}{\partial\rho} & , Z \geq z > 0, \\ \varphi(Z) = \frac{\delta\Psi}{\delta\rho}\big(\rho(Z)\big). \end{cases} \tag{13.17}$$

Lastly, as in the previous chapter, the control u satisfies an algebraic equation, which follows from the first integral in (13.16), i.e.,

$$\mathcal{L}'_u[\rho,u,\varphi,\delta u] = \int_0^Z \int_{\mathcal{P}} \delta u \left(\alpha u + \frac{\partial L}{\partial u} + \frac{u}{h}\{\rho,\varphi\}\right) \mathrm{d}y\,\mathrm{d}z. \tag{13.18}$$

which should vanish when u is optimal.

The solution strategy can also be copied from the previous chapter. Before any computation is performed, we make an initial guess for the GRIN field u. The first step is to solve the forward problem presented by Liouville's equation (13.2) together with suitable initial and boundary conditions. Next, the costate is solved backward in z from (13.17). The state ρ and costate φ together allow

us to compute the L^2-gradient and consequently perform a descent step on $\mathcal{L}$. This can again be done by, for instance, a simple gradient descent with a fixed time step or the slightly more complicated Newton descent.

To obtain the gradient, we must be careful to take into account the variables on which u is allowed to depend. In any case, if u is to represent a physical GRIN field, it may not depend on p, so that δu can be taken outside of the integral over the momentum, i.e.,

$$\mathcal{L}'_u[\rho, u, \varphi, \delta u] = \int_0^Z \int_Q \delta u \left(\int_P \left(\alpha u + \frac{\partial L}{\partial u} + \frac{u}{h}\{\rho, \varphi\} \right) \mathrm{d}p \right) \mathrm{d}q \, \mathrm{d}z. \tag{13.19}$$

As a consequence, the L^2-gradient $\frac{\delta \mathcal{L}}{\delta u}$ contains an integral over the momentum coordinate for all terms that depend on p. If u, on top of that, also doesn't depend on z, for instance, we can also pull δu outside the z-integral and consequently the gradient will contain an integral over z. We'll see an instance of this in the numerical examples.

The control input u is supposed to be a GRIN field and is therefore constrained to a certain range by nature itself and the available materials. Vacuum, by definition, has a refractive index of 1, while there's no material with a lower refractive index. Moreover, air has an index very close to 1, while diamond has a refractive index of roughly 2.4 [159]. Depending on the resources and materials available, it therefore makes sense to restrict the control input u such that

$$1 \leq u \leq n_{\max}, \tag{13.20}$$

where $n_{\max}$ is the maximal refractive index. A reasonable value, and the one we'll use here, is $n_{\max} = 1.6$. Whenever the control input u fortuitously goes out of bounds due to a diligent update, we simply cap u on both sides. That is, all $u > n_{\max}$ are set to $n_{\max}$ and all $u < 1$ are set to 1. The stopping criterion has to be adjusted in this case, as the gradient will in general not vanish. A suitable alternative is to stop when the updates to the objective functional are sufficiently small.

13.2 Designing a GRIN lens

Recall that an ideal lens performs a perfect rotation by $\frac{\pi}{2}$ on phase space, so that $q \to p$ and $p \to -q$. It's a natural and central question to optical engineering to find the best possible lens shape that approximates this behaviour as closely as

possible. Likewise, we can ask what a smooth refractive index field should look like so that it emulates as perfectly as possible an ideal lens.

This problem, or at least some approximation of it, can be formulated by making suitable choices for Z, L and Ψ in (13.5). We simply choose Ψ to be related to the rotated initial condition, for instance by choosing $\rho^\star$ the rotated initial distribution when using either possibility suggested in Section 12.4. Additionally, Z should be chosen as the focal length and L can be set to 0, as we only care to approximate target output. The regularisation term results in an optimum with minimal control effort $\|u\|^2$ weighted by α. This problem, in principle, needs to be solved for all possible initial conditions, as the action of a lens is defined as a phase space operation. However, we simply choose one suitable initial condition and optimise for that particular distribution, thereby obtaining a GRIN profile that emulates an ideal lens for that particular input.

The specific problem that we solve is a two-dimensional GRIN lens, which means phase space is two-dimensional as well, the phase space coordinates being (q, p). We restrict u to depend only on q, as this makes it easier to visualise. Thus, we seek a refractive index profile $u = n(q)$ that rotates the input distribution over a length Z by $\frac{\pi}{2}$, or a close approximation to it.

To summarise the above discussion, we assemble the Lagrangian. As u now depends only on q, we leave out integration over momentum and z, as those will only supply a constant factor, which we can absorb into α, i.e.,

$$\mathcal{L}[\rho, u, \varphi] = \Psi\left[\rho(Z)\right] + \tfrac{\alpha}{2} \int_Q u^2 \,\mathrm{d}q - \int_0^Z \int_{\mathcal{P}} \varphi\big(\rho_z + \nabla\cdot(\rho \boldsymbol{S} \nabla h)\big) \,\mathrm{d}y \,\mathrm{d}z, \quad (13.21)$$

where $Q = [-1, 1]$ is the position space. For Ψ, we'll use both the L^2-distance and the squared difference of several moments as suggested in the previous chapter, see Section 12.4. Note that u now doesn't depend on p or z, so that δu can be pulled outside the p- and z-integrals in (13.18). As a result, the L^2-gradient of $\mathcal{L}$ with respect to u is given by

$$\frac{\delta \mathcal{L}}{\delta u} = u \left(\alpha + \int_0^Z \int_P \frac{1}{h} \{\rho, \varphi\} \,\mathrm{d}p \,\mathrm{d}z \right), \quad (13.22)$$

where P is momentum space.

13.2.1 Initial guess of the profile

As an initial guess to the profile u, we use the elliptic guide, see Section 2.1.1. The guide is given by

$$u_0(q) = \sqrt{n_0^2 - \kappa^2 q^2}. \tag{13.23}$$

The focal length of the GRIN lens depends on the length Z and the constant κ. The general solution for rays in the elliptic guide were derived in the relevant section. Rays follow elliptical paths, given by

$$\kappa^2 q^2 + p^2 = \text{const.}, \tag{13.24}$$

which follows from the fact that the Hamiltonian $h = \sqrt{n_0^2 - \kappa^2 q^2 - p^2}$ is a constant. We'd like to design an optic that rotates without stretching, hence we should choose $\kappa = 1$. Furthermore, we want our position space to be $Q = [-1, 1]$ with $u_0(\pm 1) = 1$, from which it follows that $n_0 = \sqrt{2}$. In Section 2.1.1, we showed that rays follow their circular paths with an angular frequency given by κ/h. The length of one period in a ray's oscillation[1] is therefore given by $l = 2\pi h/\kappa = 2\pi h$. This can be used to provide an approximate relation between focal length, optic length Z and constant κ.

The goal is to rotate the initial distribution by $\frac{\pi}{2}$ along the optic, which in terms of rays means angles are mapped to positions and positions are mapped to angles. This is achieved if each ray traverses a length down the optical axis of $\frac{l}{4}$. As then each ray is rotated by $\frac{\pi}{2}$ around a perfect circle due to our choice of κ. Observe that l depends on the initial position of the ray via h, which means that the elliptical guide is *not* an ideal lens. However, for paraxial rays, meaning rays with $|q| \ll 1$ and $|p| \ll 1$, the Hamiltonian satisfies $h = -n_0 + \mathcal{O}(q^2) + \mathcal{O}(p^2)$, hence rays close to the origin have approximately equal Hamiltonians and therefore equal angular frequencies. Therefore, choosing a length $Z = \frac{\pi n_0}{2\kappa} = \frac{\pi}{\sqrt{2}}$ results in an ideal lens in the paraxial approximation, see Figure 13.1. As becomes clear in the figure, the larger the initial momentum, the larger the error in this approximation.

13.2.2 Implementation

We aim to solve the first-order optimality system using a discontinuous Galerkin spectral element method, see Chapters 7 and 10. As such, we'll need to solve u on a mesh that fits to the mesh used for ρ and φ. Therefore, we'll cover the

[1] I'm trying to avoid the word "wavelength" on purpose here.

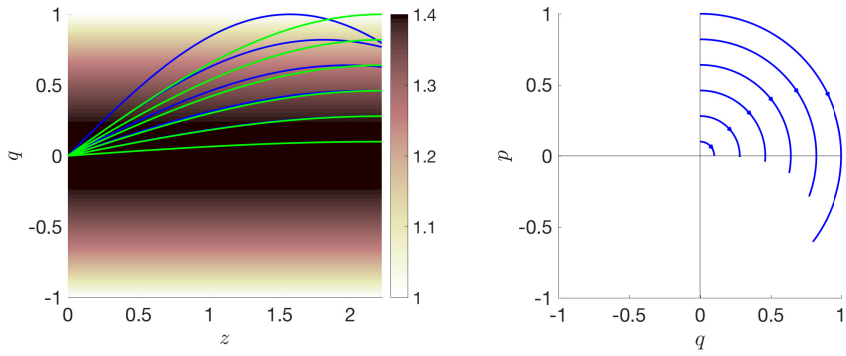


Figure 13.1: Initial guess for the GRIN profile (left) using parameters $n_0 = \sqrt{2}$ and $\kappa = 1$ chosen such that $u_0(1) = 1$. The blue lines are the exact solutions to Hamilton's equations (2.6) (left and right), while the green lines are rays where h is approximated with $-n_0$.

unit square $[-1, 1]^2$ with a regular Cartesian mesh. The Cartesian mesh can be viewed as a product mesh of two one-dimensional meshes. Since u depends only on q, we can solve u on only the horizontal axis. This choice of mesh also allows easy evaluation of the p-integral appearing in (13.22).

We want u to be smooth and continuous for a very simple reason, if u wouldn't be continuous, it wouldn't represent a GRIN profile. To enforce continuity, it's easier to define u on the Gauß-Lobatto nodes. Recall that in one dimension these nodes include the endpoints of an interval, which means there's a node on the edges of each element in the one-dimensional mesh. This makes it possible to easily enforce equality of cell edge values of neighbouring elements. The fact that the GRIN profile is defined on the Gauß-Lobatto nodes while the solution is defined on the Gauß nodes requires some translation between the two different set of nodes, as the PDEs will be solved using discontinuous Galerkin on the Gauß nodes. However, polynomial interpolation is a linear operation, so that we can construct a matrix that translates from one set of nodes to the other. This matrix can obviously be precomputed.

For two-dimensional optics, phase space is two-dimensional, where the momentum p lies on Descartes' disc, so that $|p| \leq u(q)$. Hence, the limits of the

momentum depend on the position. As such, we'll have to stretch the two-dimensional mesh in the p-direction so that the p-edges corresponds to $\pm u$. This will result in curved elements in phase space. Using curved elements entails a slight change to the bilinear transformation as outlined in Chapter 7. As the mapping now has to transform from a square reference domain to a square domain with curved edges, we'll use a *transfinite mapping*, i.e.,

$$\mathbf{y}(\xi, \eta) = \frac{1+\eta}{2}\mathbf{\Gamma}_t(\xi) + \frac{1-\eta}{2}\mathbf{\Gamma}_b(\xi), \tag{13.25}$$

where $(\xi, \eta) \in [-1, 1]^2$ are the coordinates in the reference domain. The curves $\mathbf{\Gamma}_t$ and $\mathbf{\Gamma}_b$ are the top and bottom curves, respectively. Note that $\eta = 1$, the top of the reference square, corresponds to the top of the element in phase space. Likewise, $\eta = -1$ corresponds to the bottom edge. The left and right sides are straight vertical edges, so that

$$q(\xi) = q_l \frac{1-\xi}{2} + q_r \frac{1+\xi}{2}, \tag{13.26}$$

where q_l and q_r are the left and right sides of the element. Also here, $\xi = 1$, the right edge of the reference square, corresponds to the right edge of the element. Similarly, $\xi = -1$ yields the left edge. For both curves, $\mathbf{\Gamma}_t$ and $\mathbf{\Gamma}_b$, the first component is simply the q-coordinate, hence it's given by (13.26). The p-coordinates of the top and bottom curves depend on u, where we use linear blending, so that each element has a curved top and bottom edge.

Next, we first introduce a Cartesian mesh that has E_q elements in the q-direction and E_p elements in the p-direction. The mesh furthermore spans $[-1, 1]$ in the q-direction and $[-\tilde{p}_{\max}, \tilde{p}_{\max}]$ in the p-direction. This results, after stretching, in a maximum momentum given by $u(q)\tilde{p}_{\max}$. Setting some maximum momentum $p_{\max}$ limits the maximum advection speed, allowing for larger z-steps and thus faster computations. This is merely convenience, as we could indeed choose to work with $\tilde{p}_{\max} = 1$. The p-coordinate of every top and bottom edge of the Cartesian mesh is now multiplied by $u(q)$, so that we obtain a curved mesh.

For a DG spectral element method to work on a curved mesh, the curved element boundaries can be polynomials of at most degree N [121]. This results in exact differentiation of the edge curves, such that no aliasing error is introduced. For our case, the edge curves are automatically polynomial, since u is given on the Gauß-Lobatto nodes on each one-dimensional element. If the solution to Liouville's equation is approximated with polynomial degree N, this simply means we should choose $N + 1$ or less nodes per element to represent u. For

maximal accuracy, we use the maximal $N + 1$ nodes per element in the one-dimensional mesh.

Newton minimisation

We'll use a Newton minimisation procedure to find the minimum of $\mathcal{L}$, see Section 12.3.1. Newton minimisation converges faster than, for instance, gradient descent with fixed step size. However, it does require more computational effort to find the updates. Optimistically, we expect that the benefits will outweigh the costs. In the previous chapter, we've defined the reduced objective functional j (12.28), which depends only on u, i.e.,

$$j[u] = \mathcal{L}[R_\rho(u), u, R_\varphi(u)], \tag{13.27}$$

where R_ρ is the solution operator to Liouville's equation (13.2) and R_φ is the solution operator to the adjoint equation (13.17). The value of j is equal to the objective functional where ρ is the to Liouville's equation under control input u. A Newton step is defined by equating the second variation of the reduced functional j to the gradient. With the gradient j' given by

$$j'[u, \delta u] = \int_Q \left(\alpha u + \int_0^Z \int_P \frac{\partial h}{\partial u} \{R_\rho(u), R_\varphi(u)\} \,\mathrm{d}p \,\mathrm{d}z \right) \delta u \,\mathrm{d}q, \tag{13.28}$$

with $\frac{\partial h}{\partial u} = \frac{u}{h}$. Consequently, the second variation of j is given by

$$\begin{aligned} j''[u, \delta u, \tau u] = &\int_Q \tau u \delta u \left(\alpha + \int_0^Z \int_P \frac{\partial^2 h}{\partial u^2} \{\rho, \varphi\} \,\mathrm{d}p \,\mathrm{d}z \right) \mathrm{d}q \\ &+ \int_Q \tau u \int_0^Z \int_P \frac{\partial h}{\partial u} \big(\{\delta\rho, \varphi\} + \{\rho, \delta\varphi\} \big) \,\mathrm{d}p \,\mathrm{d}z \,\mathrm{d}q, \end{aligned} \tag{13.29}$$

where $\rho = R_\rho(u)$ and $\varphi = R_\varphi(u)$ and $\frac{\partial^2 h}{\partial u^2} = -\frac{p^2}{h^3}$. In the previous chapter, it was explained that $\delta\varphi$ and $\delta\rho$ satisfy linearised versions of the solution operators and we showed how to find their PDEs. Since R_ρ is defined by demanding that $\mathcal{L}'_\varphi$ vanishes, the linearisation of the solution operator can be calculated by finding the Gâteaux derivative of $\mathcal{L}'_\varphi$. The same goes, of course, for R_φ, which

is defined in terms of $\mathcal{L}'_\rho$ vanishing. More details can be found in Section 12.3.1. Here, we simply quote the result of this procedure, i.e.,

$$\delta\rho_z + \nabla\cdot(\delta\rho\boldsymbol{S}\nabla h) = \left\{\frac{u}{h}\delta u, \rho\right\}, \quad 0 < z \leq Z, \tag{13.30}$$

with $\delta\rho(0) = 0$, while $\delta\varphi$ is defined as the solution to

$$\delta\varphi_z + (\boldsymbol{S}\nabla h)\cdot\nabla\delta\varphi = \left\{\varphi, \frac{u}{h}\delta u\right\}, \quad Z > z \geq 0. \tag{13.31}$$

We've used here the property $\{g, f\} = -\{f, g\}$ to remove any minus-signs on the right-hand side. The terminal condition of φ is given by (12.35), which reads

$$\int_{\mathcal{P}} \delta\varphi(Z)\,\tau\rho(Z)\,\mathrm{d}y = \Psi''[\rho(Z), \delta\rho(Z), \tau\rho(Z)]. \tag{13.32}$$

Hence, $\delta\varphi(Z)$ is the unique function we can associate with $\Psi''[\rho(Z), \delta\rho(Z), \cdot]$ by the Riesz Representation Theorem [152]. See Section 12.4 for a discussion on two possible choices for Ψ and their implies terminal conditions for φ and $\delta\varphi$.

The linear system we wish to solve is represented by $j''[u, \delta u, \tau u] = j'[u, \tau u]$ for all variations τu. The Riesz Representation Theorem ensures us that there exists a unique function we can associate j'' with, which we denote by $A(u, \delta u)$ and can be read off from (13.29). Hence, we wish to solve for δu the linear equation

$$A(u, \delta u) = -\frac{\delta\mathcal{L}}{\delta u}. \tag{13.33}$$

Clearly, $A(u, \cdot)$ is a bounded linear operator, taking one function and turning it into another. Unfortunately, it's impossible to extract a matrix from A, as it's defined in terms of solutions to PDEs (13.30) and (13.31). Hence, calculating the effect of the linear operator requires two PDEs to be solved, which remains true in a discrete setting. Therefore, we require a matrix-free linear solver. Furthermore, we cannot conclude that $A(u, \cdot)$ is self-adjoint, or equivalently that the matrix is Hermitian. To solve the linear equation (13.33), we'll use a matrix-free version of the generalised minimal residual method (GMRES) that's formulated on the continuum PDE level. The L^2-inner products are discretised by the DG method, while matrix-vector products should be replaced with the application of $A(u, \cdot)$.

We stipulate that computing the effect of the linear operator $A(u, \cdot)$ requires two solutions of Liouville's equation with a source term, namely $\delta\rho$ from (12.32)

and $\delta\varphi$ from (12.34). Since GMRES is a Krylov subspace method, every iteration requires another application of A. Furthermore, computing the L^2-gradient of j also requires two solutions to Liouville's equation, ρ and φ. Therefore, every iteration of the linear solver will be equally expensive as a simple gradient descent step with a fixed step size. Suffice it to say that a balance has to be struck between the descent steps and the effort put into solving for the Newton update.

GMRES, as is well known, minimises the residual over some search space. If only a single search direction is chosen, the gradient, this would simply result in a gradient descent with a second-order estimate of the optimal step size. Therefore, a single iteration of GMRES is already better than a simple gradient descent step. This means we can afford to use a relatively high tolerance and a low number of iterations.

Another matter of note is that the output distribution is very sensitive to changes in the GRIN profile. For small changes, the response is linear, which is more or less what the calculus of variations reveals. However, ray momentum also depends on the gradient of the refractive index field, which is the second of Hamilton's equations (1.33), i.e. $\frac{\mathrm{d}p}{\mathrm{d}z} = -\frac{u}{h}u'$. Thus, even though changes may be small, they can have large gradients, resulting in large responses. Variations should therefore have small gradients, as well as small values. To counter this sensitivity, we smoothen the profile after each iteration by applying a small amount of diffusion. In particular, we solve the heat equation using a fictitious coordinate t, i.e.,

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial q^2}. \tag{13.34}$$

This PDE, the heat equation, is integrated for a short interval $[0, T]$ using the updated profile as an initial condition and keeping the boundary values fixed. Famously, the heat equation tends to reduce gradients [71]. Therefore, it will stabilise the optimisation process, ensuring that responses remain linear. Naturally, any other method of reducing gradients can be used, e.g. a filter. Of course, a small amount of diffusion equation can itself be interpreted as a filter, since the heat equation (13.34) can be solved in terms of Fourier modes. Modes with shorter wavelengths decay exponentially faster than longer wavelengths.

13.2.3 Results

As mentioned earlier, we've used both the L^2-distance and the difference-of-moments squared to measure the optimality of the output distribution. Both

forms of Ψ were discussed in the previous chapter. As for numerical details, the PDEs were solved on a 20×20 mesh of curved elements as outlined above with $\tilde{p}_{\max} = 0.8$. We used the DG-SEM with a polynomial order of 5 in conjunction with the standard RK4 integrator. As a convergence criterion, the optimisation process is stopped when the relative decrease in the objective is smaller than $5 \cdot 10^{-4}$. This because we have control constraints and the gradient of the Lagrangian may not vanish in the optimum. The coefficient α was set to 10^{-4}. The number of z-steps was fixed by computing the maximum velocity in the problem and choosing a step size so that the CFL condition for DG (7.37) is satisfied, see Section 7.5. The initial profile and outcome are displayed in Figure 13.2.

As the GRIN profile changes each iteration of the optimisation process, the number of z-steps changes as well. The matrix-free GMRES solver was set to a tolerance of 0.1 and a maximum number of 10 iterations. Finally, the heat equation (13.34) was applied for an integration time of $\Delta t = 5 \cdot 10^{-3}$ and a diffusion number of $\frac{1}{40}$ using the forward Euler method and continuous Galerkin spatial discretisation. As an initial condition, we use a Gaußian distribution, i.e.,

$$\rho_0(q,p) = \frac{1}{2\pi\lambda_q\lambda_p} \exp\left(-\frac{q^2}{2\lambda_q^2}\right) \exp\left(-\frac{p^2}{2\lambda_p^2}\right). \tag{13.35}$$

The target distribution is given by $\rho^\star(q,p) = \rho_0(p,-q)$.

As a stopping criterion, we look at the relative update of the Lagrangian, as there are control constraints present. If the relative update is smaller than $5 \cdot 10^{-3}$, we stop the optimal control iteration. Therefore, the norm of the reduced gradient $\|j'\|$ may not go to zero in the optimum.

L^2-distance minimisation

In this instance, we look for a minimum of the reduced functional j, which has the same value as the Lagrangian $\mathcal{L}$ (13.21) with ρ and φ both solutions to Liouville's equation.

The optimised profile is shown in Figure 13.3 and, as expected, it resembles the elliptic guide quite closely near the centre. It flares upward a bit more than the elliptic guide when going out from the centre, after which it comes down hard. Interestingly, the control constraints are only activated near the boundary, where the refractive index would otherwise becomes less than 1. Near the centre, the optimal profile is slightly less than the original 1.4.

In Figure 13.4, we've shown the target and final distributions. Compared to Figure 13.2, the output distribution seems to be much closer to the target,

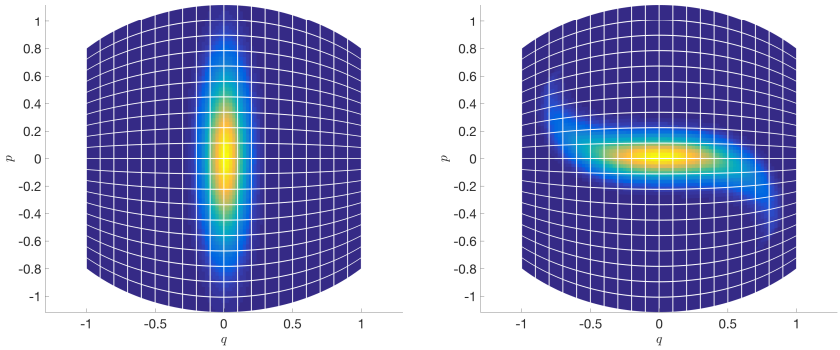


Figure 13.2: The initial brightness distribution (left) and the output distribution of the elliptic guide (right).

with thinner tails. However, observe that the tails are also longer. Even so, the L^2-distance has been reduced by over 75% compared to the elliptic guide.

Finally, the convergence history is presented in Figure 13.5. The normalised values of the reduced function j and the norm of its gradient $\|j'\|$ are plotted. The convergence criterion, a change of $5 \cdot 10^{-4}$ relative to the initial objective, is attained at iteration 51. Convergence is smooth and monotonous for j, while there is a slight increase in $\|j'\|$ around 35 iterations.

Difference-of-moments minimisation

To show the effect of a different objective function, we'll also use the difference-of-moments squared discussed in Section 12.4. In particular, we'll use the first set of moments up to and including the third-order moments, thus the moments generated by integrating over phase space ρq, ρp, ρq^2, ρqp, ρp^2, ρq^3, $\rho q^2 p$, ρqp^2 and ρp^3. There's some connection between monomials and aberration coefficients, see for instance Wolf [16], but we're not sure if the moments of ρ somehow represent aberrations.

The minimisation of the difference-of-moments is another kettle of fish entirely. Minimising the moments focusses much more on the global nature of the distribution. In particular, the tails inherent in the elliptic wave guide carry a much greater weight than in the previous example of the L^2-distance. At the same time, the accuracy in the centre carries much less weight compared to the L^2-distance. Hence, the centre is sacrificed to ensure that the behaviour near

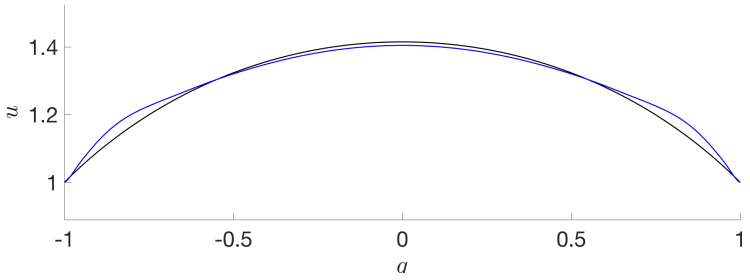


Figure 13.3: The L^2-distance optimised GRIN profile (blue) together with the initial guess of the elliptical guide (black).

the tails is correct.

The resulting GRIN profile after the optimisation process is shown in Figure 13.6. Note that the maximum refractive index is less than $\sqrt{2}$ while the minimum is greater than 1. Therefore, the control constraints aren't active anywhere, contrary to the L^2-distance minimisation. Furthermore, the profile features a central bump, deviating further from the elliptical guide.

Figure 13.7 displays the output distribution after the optimisation process[2]. One thing that particularly springs to the eye is that the central mass seems to be rotated by $\frac{\pi}{4}$ with respect to the target distribution. At the same time, the tails are straightened out somewhat, so that the distribution is more closely centred around the $p = 0$ axis compared to the previous example.

Finally, we present the convergence history in Figure 13.8. Also here we find major differences with the previous example where the L^2-distance was minimised. In particular, the convergence criterion is attained much quicker, requiring only 10 iterations. The objective j does seem to converge monotonously, but the convergence of the norm of the reduced gradient is very rough. In the L^2-distance example, we also saw an increase of $\|j'\|$, but it was much more gradual and less severe. Nonetheless, the reduction in the value j is over 70%.

13.3 Final remarks on GRIN optimisation

We've presented some details on how to employ optimal control on GRIN optics by interpreting an optic as an open loop controller on Liouville's equation.

[2]The shape looks a bit like an Indonesian kris dagger to me.

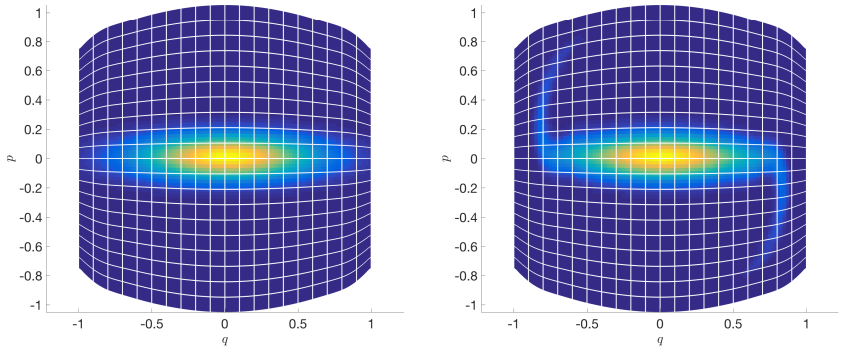


Figure 13.4: The target distribution (left) and the output distribution after minimising the L^2-distance objective (right).

Our numerical example consisted of finding a GRIN profile that emulates an ideal lens for a particular input distribution. We've shown some results for the two metrics presented in the previous chapter, namely the L^2-distance and the difference-of-moments-squared. The results are encouraging, confirming that the optimal control framework can indeed be successfully applied to Liouville's equation and GRIN optics. We hasten to add that this positive result has been made possible by the fast and accurate solvers we've developed in the previous part.

The numerical examples show that choosing a different objective functional can have a great impact on the result. If we were to judge the final output distributions by eye, then we'd conclude that the L^2-distance minimisation gives a more desirable result. The tails of the distribution resulting from our starting guess of an elliptic guide, compare Figures 13.2 and 13.4, have been diminished, though they are somewhat longer. By minimising the difference-of-moments, the tails are straightened out compared to Figure 13.3, but the centre of the distribution looks a bit off-kilter. Yet, both processes result in a reduction of the objective of more than 70%. Perhaps the eyeball-norm of the second example can be improved by including more moments. Another possibility would be to use the result of the difference-of-moments minimisation as an initial guess for the L^2-distance minimisation.

In the next chapter, we'll turn to the optimisation of refractive surfaces. Although the basic ideas remain the same, the control is now *impulsive*, rather than gradual. Whereas the GRIN profile exerts a continuous and steady pressure

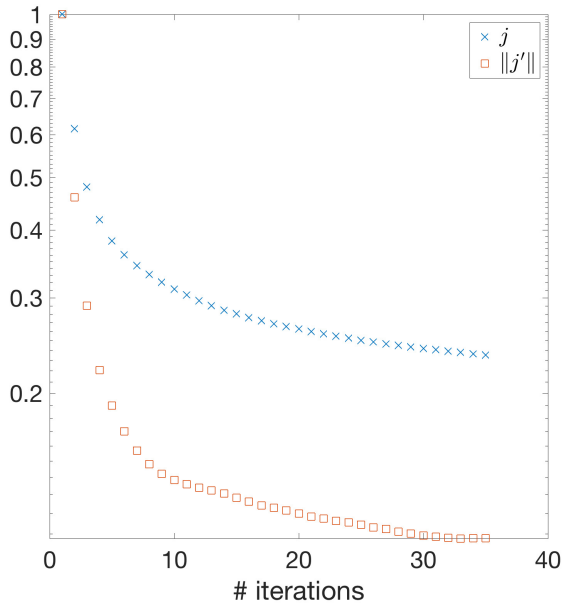


Figure 13.5: The convergence history of the reduced functional j using the L^2-distance and the norm of its gradient j', both normalised.

on the distribution, a refractive surface does mostly nothing. The distribution is undergoing free propagation for the major part of the optic. In fact, the brightness distribution undergoes free propagation almost everywhere, since the interface is a null set. Only in transitioning from one refractive index to another will the distribution be influenced. Mathematically, this isn't more or less hard to handle, but conceptionally it's rather contrasting.

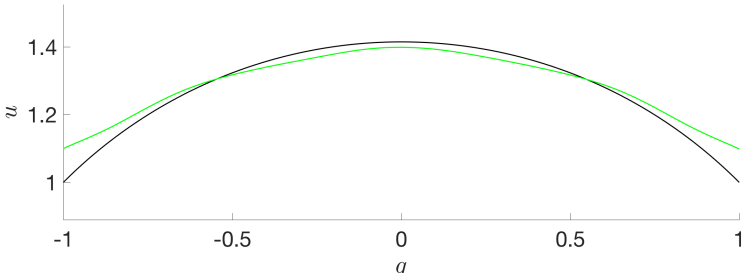


Figure 13.6: The difference-of-moments optimised GRIN profile (green) together with the initial guess of the elliptical guide (black).

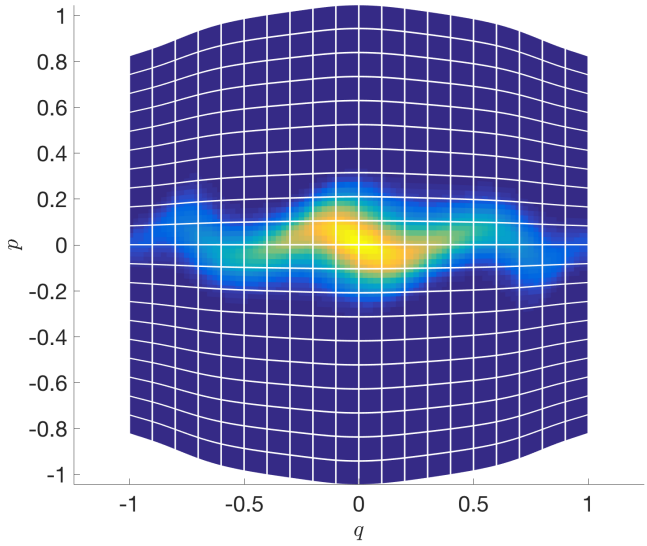


Figure 13.7: The output distribution after minimising the difference-of-moments.

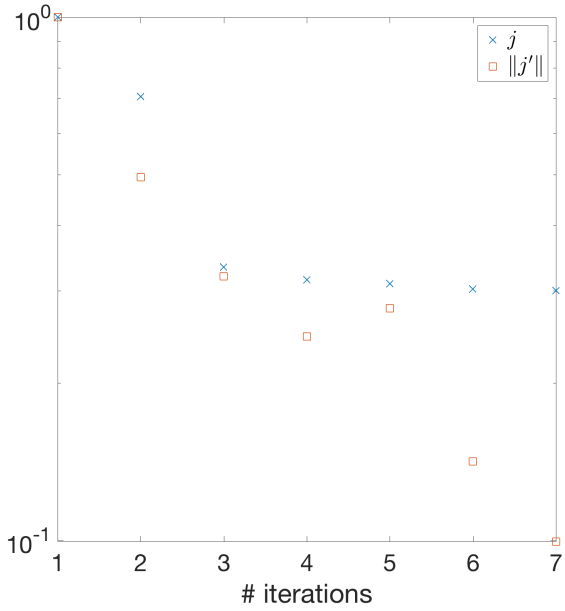


Figure 13.8: The convergence history of the reduced functional j using the difference-of-moments and the norm of its gradient j', both normalised.

Chapter 14

Optimising freeform optical surfaces

> Sculpting is easy, just remove
> anything that's not a sculpture.
>
> Michelangelo Buonarotti

In this chapter we apply optimal control theory to freeform surfaces, resulting in the optimisation of optically active surfaces. This process is also known as tailoring or customising optics. Hence, in this case we assume that the refractive index field is piecewise constant, with the interfaces given by smooth surfaces. This is of course also a practical choice compared to the previous chapter on GRIN optics, as shaped surfaces are currently much easier to manufacture than arbitrary smooth index fields.

Optimising freeform surfaces is an integral part of optical engineering. One could argue that Archimedes is, according to legend, one of the first to use freeform optimisation. Allegedly, he used a bunch of soldiers holding mirrors to form a death ray with which to burn ships [160, 161]. If this tale is true, he would arguably be the first to have used adaptive optics as well. According to Lucian, Archimedes used his awesome burning glass to protect his beloved city of Syracuse from the invading Romans [162].

In more recent times, there has been success with a formulation of the freeform surface problem in terms of a second-order nonlinear PDE known as the Monge-Ampère equation [163]. It can be derived by using conservation of

energy and applying the basic laws of optics. The Monge-Ampère equation appears commonly in optimal transport problems [164]. It's particularly difficult to solve and requires specialised numerical methods, see for instance Kochengin and Oliker, who provided the first provably stable solver [165–167]. Other solvers for the reflector problem followed in quick succession [168, 169], as well as those specifically aimed at the two-reflector problem [170]. Naturally, these solvers can be applied to refractive freeform optics problems as well [171, 172]. However, all of the solvers mentioned above assume zero étendue. This means either point sources or perfectly collimated beams are assumed as input light sources. We've found only one publication that tackles the nonzero-étendue problem, which is the method of Hirst et al. [173]. However, their solver is based on solving an implicit integral equation, rather than the Monge-Ampère equation. Moreover, all the works summarised here solve problems involving either illuminance or intensity, none of them are able to solve the problem on the full phase space.

Our approach is radically different from the aforementioned literature. Instead of deriving a PDE for the surface and trying to solve it, our line of attack is much more circumlocutory. The relation between the work presented in this chapter and those concerning the Monge-Ampère equation is somewhat reminiscent of the Benamou-Brenier formula [174]. It establishes the equivalence between the Wasserstein distance and a continuity equation with a velocity field that morphs one distribution into another. The Wasserstein distance is defined as the cost of optimal transport. The Benamou-Brenier formula says the velocity field is such that optimal transport incurs minimal total kinetic energy. Thus, the Benamou-Brenier formula provides a relation between physical transport and the Monge-Ampère equation. Likewise, the works mentioned above have focussed mostly on the Monge-Ampère equation, while we focus on the physical transport of light. To make the connection even more explicit, Liouville's equation is a type of continuity equation.

In theory, our approach can handle both zero- and nonzero-étendue problems. Here, we present some results for nonzero-étendue problems. Moreover, our method is essentially defined on phase space, hence we present our method in its native environment of full phase space optimisation.

Recall that the PDE we wish to solve, Liouville's equation, is given by

$$\frac{\partial \rho}{\partial z} + \nabla \cdot \left(\rho \boldsymbol{S} \nabla h \right) = 0, \tag{14.1}$$

with initial condition $\rho(0, q, p) = \rho_0(q, p)$. In two-dimensional optics, the gradi-

ent operator is given by $\nabla = (\partial_q, \partial_p)^T$ and the matrix $\boldsymbol{S}$ is given by

$$\boldsymbol{S} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}. \tag{14.2}$$

Finally, the Hamiltonian h is given by

$$h(z, q, p) = -\sqrt{n(z, q)^2 - p^2}, \tag{14.3}$$

where n is now assumed to be piecewise constant. When traversing an interface where the refractive index changes discontinuously from n_1 to n_2, rays obey Snell's law, given by

$$\mathcal{S}\left(p; n_1, n_2, \nu\right) := \begin{cases} p - \left(\psi + \operatorname{sgn}\left(n_2\right)\sqrt{\delta}\right)\nu & \text{if } \delta \geq 0, \\ p - 2\psi\nu & \text{if } \delta < 0, \end{cases} \tag{14.4a}$$

where

$$\delta := n_2^2 - n_1^2 + \psi^2 \quad \text{and} \quad \psi := p\nu \pm \sqrt{n_1^2 - p^2}\sqrt{1 - \nu^2}, \tag{14.4b}$$

where the sign is to be taken such that $\psi \leq 0$, which follows from the angle convention of Snell's law, see Section 1.3. This is the two-dimensional form, so that one component of the normal provides sufficient information, hence $\nu \in \mathbb{R}$ such that $|\nu| \leq 1$. The solution to Liouville's equation is continuous along rays for continuous initial conditions, implying

$$\rho\big(z^+, q(z^+), p(z^+)\big) = \rho\big(z^-, q(z^-), \mathcal{S}(p(z^-); n_1, n_2, \nu))\big), \tag{14.5}$$

where pluses and minuses denote one-sided limits toward the interface.

At the same time, we wish to minimise some objective function, e.g., closely approaching some desired output state. Snell's law gives us a way to couple the various regions where n is constant. Note that both mirrors and lenses can be captured in this formulation with the trick that $n_2 = -n_1$ represents a mirror, see Section 1.3.

For optically active surfaces, we have to treat the regions delineated by the interface separately. For instance, for a lens surface given by $z = \zeta(q)$, there's a region with $z > \zeta(q)$, which has a different refractive index compared to the region where $z < \zeta(q)$. Therefore, both regions have to be treated separately, with Snell's law giving us the connection between the two regions.

14.1 Choice of control

Regardless of whether we have a lens or mirror, we have to somehow parametrise the surface and choose a control input, denoted u. Clearly, u must represent the surface in some way, but it's worth giving it some careful thought[1]. A first naive attempt could be to describe the surface directly, i.e., setting u equal to ζ in the case of a lens. However, Snell's law doesn't care about the position of the intersection with a ray, it depends on the local surface normal. The normal is in turn related to the gradient of the surface. If we were to pursue the naive choice, we'd have to perform an extra integration by parts in the augmented functional, much like in the previous chapter, since the surface normal would then be related to the gradient of u. The issue is that Snell's law is quite nonlinear in its dependence on the normal, resulting in a highly complicated expression.

In general, it 's easier if the system depends directly, albeit nonlinearly, on u instead of its derivatives. As such, we should use the gradient of the surface as control input. This way we avoid both the pesky inversion of Snell's law and the partial integration. The price we have to pay is that we then need to solve for the surface as well. Suppose we're optimising a freeform lens where ζ is fixed on the boundary. Our choice of u allows us to differentiate the control relation with respect to q, from which we find that ζ satisfies Poisson's equation in one dimension, i.e.,

$$\zeta'' = u' \tag{14.6}$$

Thus, our slightly more informed choice of control leads to a more simple calculation for the gradient of the augmented functional. This simplification comes at the price of solving Poisson's equation to find the surface itself. Naturally, we need to know the surface to sufficient accuracy, which is to say with the same order as the numerical solution to Liouville's equation. Hence, if we solve Liouville's equation with a spectral element method, we also need to solve Poisson's equation with a spectral element method.

Finally, there's a modest advantage of choosing u to be the gradient of the surface, rather than the normal. Snell's law depends directly on the normal and not on the gradient, so why not go all the way? The reasons has to do

[1] There's a, probably apocryphal, story about Michelangelo where he stared at a block of marble for ages without lifting his hammer and chisel once. Then, suddenly, the David appeared as if through magic. When asked about it Michelangelo said he'd been studying the marble so excruciatingly that he knew exactly where to strike. If nothing else, this story taught me to think carefully beforehand in the hope of making the right choices resulting in less work later on.

with control constraints. Suppose we use as our control the q-component of the normal, ν in (14.4). Then our control is immediately constrained by the fact that $|\nu| \leq 1$. Using the gradient results in an unconstrained control, which is slightly easier to deal with, see Chapter 12. Choosing the gradient of the surface as the control input results in the normal being given by

$$\nu(u) = \frac{u}{\sqrt{1+u^2}}, \tag{14.7}$$

which will always lie within the constraints $|\nu| \leq 1$.

14.2 Assembling the augmented functional

To find the proper first-order necessary conditions for optimality, i.e., a set of PDEs that characterise an optimal surface, we first have to construct the augmented functional. As per usual the augmented functional, or Lagrangian, contains the objective function as well Liouville's equation carried by a Lagrange multiplier. Snell's law provides the transition across interfaces, so that it must be enforced using another Lagrange multiplier. As an example, we consider a freeform lens described by $z = \zeta(q)$ with a transition from n_1 to n_2, with $n_1 > n_2$. We assume the optimality of the output $\rho(Z)$, which is shorthand for $\rho(Z, \cdot)$, is measured by a Gâteaux differentiable functional Ψ, see 12.4. Furthermore, we assume the objective is given by

$$\Psi[\rho(Z)] + \int_{\mathcal{P}} \int_{0}^{Z} L(z, \mathbf{y}, \rho, u)\, \mathrm{d}z\, \mathrm{d}y,$$

where $\mathcal{P}$ is phase space. In each separate region where the refractive index is constant, phase space is a product space given by $Q \times P$. Note the order of integration in the fourth term, which actually represented two integrals, one of each region where the refractive index is constant. The z-limits for each region depends on q, which is why this is the correct order. The Lagrangian is therefore

given by

$$\begin{aligned}\mathcal{L}[\rho,u,\varphi,\mu] =&\Psi[\rho(Z)] + \int_{\mathcal{P}}\int_0^Z L(z,\mathbf{y},\rho,u)\,\mathrm{d}z\,\mathrm{d}y + \tfrac{\alpha}{2}\int_Q u^2\,\mathrm{d}q \\ &- \int_{\mathcal{P}}\int_0^Z \varphi\big(\rho_z + \nabla\cdot(\rho\boldsymbol{S}\nabla h)\big)\,\mathrm{d}z\,\mathrm{d}y \\ &- \int_{\mathcal{P}} \mu\Big(\rho\big(\zeta(q)^+,q,p\big) - \rho\Big(\zeta(q)^-,q,S\big(p;n_1,n_2,\nu(u)\big)\Big)\Big)\,\mathrm{d}y,\end{aligned} \tag{14.8}$$

where, the first three terms represent the objective and regularisation, the fourth term is the PDE constraint and the last term represents the enforcement of Snell's law. We'll also abbreviate $\rho^+ = \rho\big(\zeta(q)^+,q,p\big)$ and $\rho^- = \rho\Big(\zeta(q)^-,q,S\big(p;n_1,n_2,\nu(u)\big)\Big)$.

In the following, we'll apply the machinery of calculus of variations to derive a system of first-order necessary conditions. In particular, this means calculating the Gâteaux derivative of the Lagrangian and demanding that it vanishes for all variations. We refer the reader to Section 12.2 for more details. Here, we'll mostly quote the results from that section and only deal with the novelty of the interface and Snell's law. As per usual in constrained optimisation problems, variation with respect to the Lagrange multipliers φ and μ leads to the constraint equations. Variation with respect to ρ leads to the adjoint or costate equation for φ, together with a terminal condition. Variation of ρ on both sides of the interface leads to a jump condition for the costate φ, which rather unsurprisingly will also be Snell's law.

First, we quote the results from Chapter 12 that remain unaltered. Naturally, variation with respect to φ and μ, i.e., setting $\mathcal{L}'_\varphi = 0$ and $\mathcal{L}'_\mu = 0$ leads to Liouville's equation (14.1) and Snell's law (14.4), respectively. The adjoint equation is given, on both sides of the interface, by

$$\frac{\partial\varphi}{\partial z} + (\boldsymbol{S}\nabla h)\cdot\nabla\varphi = -\frac{\partial L}{\partial\rho}, \tag{14.9a}$$

with the terminal condition

$$\varphi(T) = \frac{\delta\Psi}{\delta\rho}\big(\rho(Z)\big). \tag{14.9b}$$

Here $\frac{\delta\Psi}{\delta\rho}$ is the L^2-gradient of Ψ, which is extensively discussed in Section 12.4. The jump condition for φ is derived by varying ρ on the interface. To facilitate this, we first split the fourth integral in (14.8) into the two regions separated by the interface. Integrating by parts each region, we obtain that

$$-\int_{\mathcal{P}}\int_{0}^{Z}\varphi\rho_z\,\mathrm{d}z\,\mathrm{d}y=\int_{\mathcal{P}}\int_{0}^{Z}\rho\varphi_z\,\mathrm{d}z\,\mathrm{d}y-\int_{\mathcal{P}}\rho\varphi\big|_0^{\zeta(q)}\,\mathrm{d}y-\int_{\mathcal{P}}\rho\varphi\big|_{\zeta(q)}^{Z}\,\mathrm{d}y. \tag{14.10}$$

Variations with respect to ρ will therefore pick out the boundary terms over the optical surface, combining with the terms involving μ. As such, the Gâteaux derivative of $\mathcal{L}$ with respect to ρ is given by

$$\begin{aligned}\mathcal{L}'_\rho[\rho,u,\varphi,\mu,\delta\rho]&=\int_{\mathcal{P}}\int_{0}^{Z}\left(\frac{\partial L}{\partial\rho}+\varphi_z+(\boldsymbol{S}\nabla h)\cdot\nabla\varphi\right)\delta\rho\,\mathrm{d}z\,\mathrm{d}y\\&+\int_{\mathcal{P}}\left(\frac{\delta\Psi}{\delta\rho}-\varphi(Z)\right)\delta\rho(Z)\,\mathrm{d}y+\int_{\mathcal{P}}(\mu-\varphi^-)\delta\rho^--(\mu-\varphi^+)\delta\rho^+\,\mathrm{d}y.\end{aligned} \tag{14.11}$$

Thus, aside from the ordinary result (14.9), we also have that $\mu=\varphi^+$ and while $\mu=\varphi^-$. In particular, this tells us that $\varphi^+=\varphi^-$, so that φ also satisfies Snell's law.

Finally, we'll calculate the variation of $\mathcal{L}$ with respect to u. Applying the definition of the Gâteaux derivative, see Section 12.1, we find that

$$\mathcal{L}'_u[\rho,u,\varphi,\mu,\delta u]=\int_{Q}\alpha u\,\delta u\,\mathrm{d}q+\int_{\mathcal{P}}\int_{0}^{Z}\frac{\partial L}{\partial u}\delta u\,\mathrm{d}z\,\mathrm{d}y+\int_{\mathcal{P}}\mu\frac{\partial\rho^-}{\partial u}\delta u\,\mathrm{d}y. \tag{14.12}$$

We've already found that $\mu=\varphi^+$, so the only thing left to do is to calculate the derivative of ρ^- with respect to u. Careful usage of the chain rule yields

$$\frac{\partial\rho^-}{\partial u}=\nu'(u)\frac{\partial S}{\partial\nu}\frac{\partial\rho^-}{\partial p}. \tag{14.13}$$

Note that it's also possible to state continuity of the brightness along rays in terms of the backward rays, leading to an alternative expression for the gradient in terms of ρ^+. Of course, these two options should provide the same answer.

14.2.1 Newton minimisation

As has become a habit in this part, we'd like to minimise $\mathcal{L}$ using Newton's method. Therefore, we'll also need to calculate the second Gâteaux derivative. We first write the reduced functional $j[u] = \mathcal{L}[R_\rho(u), u, R_\varphi(u), R_\mu(u)]$, where R_ρ is the solution operator for (14.1), likewise R_φ provides the solution φ of (14.9a) and $R_\mu(u) = \varphi^+$. Like before, we have

$$j'[u, \delta u] = \mathcal{L}'_u[R_\rho(u), u, R_\varphi(u), R_\mu(u), \delta u] = \int_Q \delta u \left(\alpha u + \int_P \mu \frac{\partial \rho^-}{\partial u} \,\mathrm{d}p \right) \mathrm{d}q, \tag{14.14}$$

where the solutions $\rho = R_\rho(u)$ and $\mu = R_\mu(u)$ should be used. The second variation of j is therefore given by

$$j''[u, \delta u, \tau u] = \int_Q \tau u \left(\alpha \delta u + \int_P \delta\mu \frac{\partial \rho^-}{\partial u} + \mu \frac{\partial \delta\rho^-}{\partial u} + \mu \frac{\partial^2 \rho^-}{\partial u^2} \delta u \,\mathrm{d}p \right) \mathrm{d}q. \tag{14.15}$$

Again, some careful application of the chain and product rules for differentiation gives that

$$\frac{\partial^2 \rho^-}{\partial u^2} = \nu''(u) \frac{\partial S}{\partial \nu} \frac{\partial \rho^-}{\partial p} + \big(\nu'(u)\big)^2 \frac{\partial^2 S}{\partial \nu^2} \frac{\partial \rho^-}{\partial p} + \left(\nu'(u) \frac{\partial S}{\partial \nu} \right)^2 \frac{\partial^2 \rho^-}{\partial p^2}. \tag{14.16}$$

In Section 12.3.1, we've explained that $\delta\rho$ and $\delta\mu$ are now defined by linearisation of the solution operators. The solution operators are defined by demanding that certain variations of $\mathcal{L}$ vanish, for instance, R_ρ is defined by the relation $\mathcal{L}'_\varphi = 0$. The PDE for $\delta\rho$ can therefore be found by calculating the variation of $\mathcal{L}'_\varphi$ and demanding that the result vanishes as well. The governing equations for $\delta\varphi$ and $\delta\mu$ can be found in the same way. Here, we quote the result from Section 12.3.1, which states that $\delta\rho$ satisfies Liouville's equation, i.e.,

$$\delta\rho_z + \nabla \cdot (\delta\rho \boldsymbol{S} \nabla h) = 0. \tag{14.17}$$

The initial condition for $\delta\rho$ follows from the fact that ρ itself has specified initial conditions. Therefore, the variation must satisfy $\delta\rho(0) = 0$. This implies that $\delta\rho$ is identically zero, at least up to the surface, so that also

$$\delta\rho^- = 0. \tag{14.18}$$

To know how $\delta\rho$ behaves after encountering the interface, we must look for a jump condition for $\delta\rho$. Similarly to the earlier argument, the jump condition

for ρ is defined as demanding that $\mathcal{L}'_\mu = 0$. To find the jump condition on $\delta\rho$, we therefore have to look at the variation of $\mathcal{L}'_\mu$ and demand that the result vanishes. This procedure yields

$$\int_{\mathcal{P}} \delta\mu \left(\delta\rho^- + \frac{\partial \rho^-}{\partial u}\delta u - \delta\rho^+ \right) \mathrm{d}y = 0. \tag{14.19}$$

This relation must hold for all variations $\delta\mu$, so that

$$\delta\rho^+ = \delta\rho^- + \frac{\partial \rho^-}{\partial u}\delta u = \frac{\partial \rho^-}{\partial u}\delta u, \tag{14.20}$$

where we've applied (14.18). The equation for φ is given by variation of (14.11), yielding

$$\delta\varphi_z + (\boldsymbol{S}\nabla h)\cdot\nabla\delta\varphi = -\frac{\partial^2 L}{\partial \rho^2}\delta\rho, \tag{14.21}$$

with terminal condition given by

$$\int_{\mathcal{P}} \delta\varphi(Z)\,\tau\rho(Z)\,\mathrm{d}y = \Psi''[\rho(Z), \delta\rho(Z), \tau\rho(Z)], \tag{14.22}$$

see Section 12.4 for explicit expressions and a discussion on how to choose Ψ. Finally, the jump condition $\delta\varphi^+ = \delta\varphi^-$ must be observed, i.e., $\delta\varphi$ satisfies Snell's law over the optical surface.

14.2.2 Solution strategy

Compared to the basic recipe outlined in Chapter 12, we only need to add a single step, which is to actually solve for the surface. We must still make an initial guess for the surface, compute the solution to Liouville's equation forward in z and the costate equation backwards in z. The control input is also determined either by a gradient descent or a Newton step. The difference with earlier strategies is that after the update to the control input is made, we now also have to compute the surface shape itself by solving Poisson's equation (14.6). This is done using a continuous Galerkin spectral element method, see e.g., Kopriva [121].

14.3 Example: ideal lens

As an example of optimal control for optical surfaces, we again use the example of mimicking ideal lens behaviour, this time by finding the optimal shape of a refractive surface. Thus, for a specific input distribution, we use as a target distribution the input distribution rotated by $\frac{\pi}{2}$, i.e., $\rho^\star(q,p) = \rho_0(p,-q)$. In terms of (14.8), we use $L = 0$, as we're only interested in achieving the target distribution. We'll use both choices suggested in Section 12.4, namely the L^2-distance and the difference-of-moments-squared. The Lagrangian that we'll use is therefore given by

$$\begin{aligned}\mathcal{L}[\rho,u,\varphi,\mu] =& \Psi[\rho(Z)] + \tfrac{\alpha}{2}\int_{\mathcal{Q}} u^2\,\mathrm{d}q - \int_0^Z\int_{\mathcal{P}} \varphi\big(\rho_z + \nabla\cdot(\rho\boldsymbol{S}\nabla h)\big)\,\mathrm{d}y\,\mathrm{d}z \\ &- \int_{\mathcal{P}} \mu\big(\rho^+ - \rho^-\big)\,\mathrm{d}y,\end{aligned} \tag{14.23}$$

where the last term implements Snell's law. The parameter α is set to 10^{-3}.

We compute the numerical solution to Liouville's equation using the discontinuous Galerkin method detailed in Chapter 10. The solution is computed on a regular mesh of 20×20 elements with a polynomial degree of $N = 4$ and a maximum momentum set to $\frac{3}{4}$. This is not an essential constraint of the system, it simply limits the advection velocity. In particular, on the edge of Descartes' disc, the advection velocity diverges, resulting in extremely small z-steps. Choosing some maximum momentum limits the velocity and results in fewer z-steps. The z-integration is performed using the procedure specified in Section 10.3.

As an initial guess for the surface we use the spherical lens also discussed in Section 10.4.2, as spherical lenses provide a crude approximation to the ideal. The lens is a plano-convex with the curved side having a radius of 1.8 units and a refractive index of 1.6, resulting in a focal length of 3 units, see Figure 14.1. The lens has a height of 2 units and no light can get around it. The source distribution is simply chosen to be a Gaußian, given by

$$\rho_0(q,p) = \frac{1}{2\pi\sigma_q\sigma_p}\exp\left(-\frac{q^2}{2\sigma_q^2}\right)\exp\left(-\frac{p^2}{2\sigma_p^2}\right), \tag{14.24}$$

where $\sigma_q = \frac{2}{10}$ and $\sigma_p = \frac{1}{10}$. In Figure 14.2

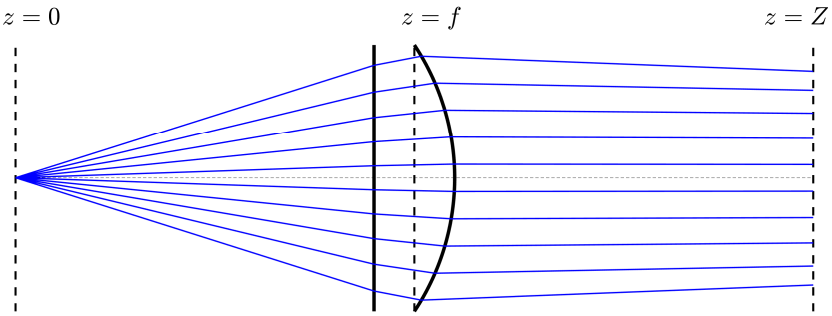


Figure 14.1: Sketch of the situation for the spherical lens case. The initial condition is specified on the $z = 0$ focal plane. Several rays (blue) are sketched as well.

See Figure 14.2 for a plot of the initial condition and the output distribution for the spherical lens. We stress that the optimised surface will only mimic ideal lens behaviour for this specific input. To obtain a true ideal lens, we'd need to optimise the surface using all possible initial conditions as well.

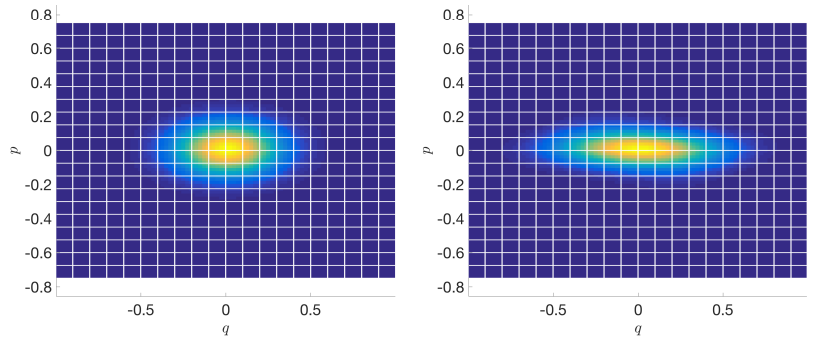


Figure 14.2: Source (left) and output (right) distribution for the spherical lens initial guess to u.

One thing that catches the eye in the output distribution of the spherical lens is the amount of stretching in the q-direction. An ideal lens would rotate the left distribution in Figure 14.2 by $\frac{\pi}{2}$, however, it looks more like it stretched

the distribution slightly. This is due to all sorts of aberrations that spherical lenses are prone to.

14.3.1 L^2-distance minimisation

When Ψ is chosen as the L^2-distance, the optimality system and its computation are slightly simplified. However, this also provides some room for instabilities to occur in the optimisation process. To stabilise the process, we add a term $\frac{\gamma}{2}\int_Q (u')^2 \,\mathrm{d}q$ to the objective, resulting in a term added to the derivative, i.e.,

$$\frac{\delta \mathcal{L}}{\delta u} = \alpha u - \gamma u'' + \int_P \mu \frac{\partial \rho^-}{\partial u} \,\mathrm{d}p. \tag{14.25}$$

The second derivative will also receive an additional term. The reasoning behind the addition term will become clear soon. For now, it's enough to note that, since u is the gradient of the surface, regions with high curvature are penalised.

In Section 12.4, we've seen that the both $\delta\rho$ and $\delta\varphi$ satisfy Liouville's equation. These are necessary to compute the second Gâteaux derivative of $\mathcal{L}$. At the same time the terminal condition is simply given by (14.22):

$$\delta\varphi(Z) = \delta\rho(Z), \tag{14.26}$$

see also (12.41). This, together with the fact that $\delta\varphi$ and $\delta\rho$ both satisfy Liouville's equation, implies $\delta\varphi = \delta\rho$ everywhere. Hence, the second Gâteaux derivative of the reduced functional j is simplified greatly, i.e.,

$$j''[u, \delta u, \tau u] = \int_Q \tau u \, \delta u \left[\alpha + \int_P \left(\frac{\partial \rho^-}{\partial u} \right)^2 + \varphi^+ \frac{\partial^2 \rho^-}{\partial u^2} \,\mathrm{d}p \right] - \gamma \, \tau u \, \delta u'' \,\mathrm{d}q, \tag{14.27}$$

which is (14.15) with $\delta\rho^- = 0$ and $\mu = \delta u \frac{\partial \rho^-}{\partial u}$. Note that this reduced gradient doesn't contain any dependence on other variations. Thus, the Newton step can be solved explicitly in this case, which greatly speeds up the optimisation procedure. Contrast this with the previous chapter, where a matrix-free GMRES linear solver had to be used and each matrix product required another two more instances of Liouville's equation to be solved. The simple form of the second derivative of $\mathcal{L}$ is a double-edged sword, however, as it also means the gradient of u isn't regulated in any kind of way when $\gamma = 0$. This can cause

instabilities in the optimisation process. The additional regulator term will tend to smoothen out δu. In particular, the update δu will satisfy an elliptical PDE with homogeneous Neumann boundary conditions. The elliptical equation for δu will also be solved using a continuous Galerkin spectral element method. The parameter γ is chosen to be 10^{-2}.

Additionally to the simple form of the second derivative of j, the costate on the surface is easier to evaluate. This has again to do with the fact that both ρ and φ satisfy Liouville's equation. The terminal condition for the costate is given by

$$\varphi(Z) = \rho(Z) - \rho^\star. \tag{14.28}$$

Since Liouville's equation is a linear advection equation, we can therefore integrate ρ only up to the surface and $\rho^\star$ backwards from the target screen to the surface and take the difference there. This will save roughly half the computational effort.

The optimised surface itself is plotted in Figure 14.3, together with the initial guess of a spherical surface. The thing that catches the eye here is that the optimised surface is completely above the initial spherical surface. Furthermore, the optimised surface has a higher curvature near the centre, while it's a bit less curved near the edges, though the difference is small.

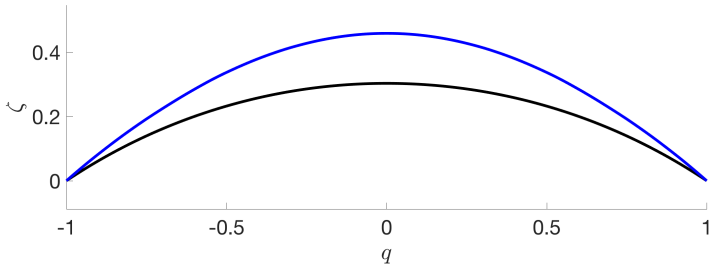


Figure 14.3: Initial lens surface (black) and final lens surface(blue).

The final distribution, after the optimisation process, is shown in Figure 14.4. There are several interesting things to point out about the final state. First, by eye it's hard to judge which distribution is closer compared to the right plot in Figure 14.2. The optimised distribution looks more like a spherical Gaußian, although there are some small tails sticking out at around and angle of $\frac{\pi}{4}$. However, compared to the final state of the initial spherical surface, it's much

closer in the norm we're using. To provide some numbers, the reduction in the objective value j is almost 80%. Most of this reduction is in the L^2-distance, with the regularisation term contributing roughly $5 \cdot 10^{-3}$ relative to the whole.

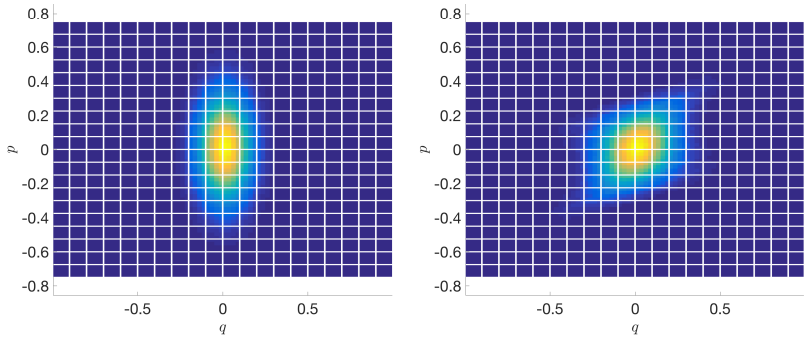


Figure 14.4: Target (left) and final (right) distributions for the ideal lens example using the L^2-distance as objective.

Finally, for completeness we show the convergence history of the optimisation process in Figure 14.5. The convergence is fairly quick in the first 15 or so iterations and slows down afterwards. The convergence criterion of a relative change of $5 \cdot 10^{-3}$ to the objective is attained there and the final norm of the gradient is about $2 \cdot 10^{-2}$ compared to the initial norm of the gradient.

14.3.2 Difference-of-moments minimisation

As suggested in Section 12.4, an alternative objective function for the final state is to look at the difference-of-moments squared. The logic being that two distributions with equal moments are themselves equal, provided they have compact support or bounded domain. We look at the first couple of moments up to the third-order ones, i.e., q, p, q^2, qp, p^2, q^3, q^2p, qp^2 and p^3, all multiplied with the distribution and integrated over the domain. As discussed in the aforementioned section, choosing a small number of moments results in a very smooth costate φ, which is often a benefit. The downside is that the full apparatus of matrix-free GMRES, also discussed in Section 13.2.2, is needed to perform a Newton step. Each optimisation step will therefore be more expensive compared to the L^2-distance minimisation. Some additional numerical details: we don't need a stabilising term here, hence $\gamma = 0$, while we use a tolerance of $\frac{1}{10}$ on the GMRES solver with a maximum number of iterations of 10.

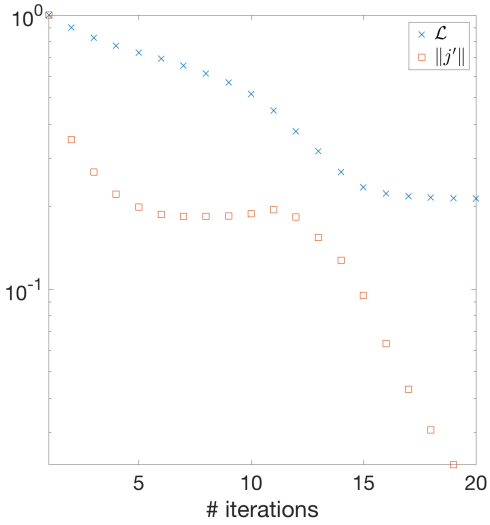


Figure 14.5: Convergence history for the Lagrangian (blue) and the norm of the reduced gradient (red).

The optimised surface is presented in Figure 14.6 and the first thing to notice is that it looks a lot like the previous optimised surface in Figure 14.3. The differences are minute and mostly noticeable near the edges. Again, the surface is entirely above the initial spherical surface and has a higher radius of curvature near the centre.

If we look at the final output distribution, presented in Figure 14.7, we're again struck by a feeling of familiarity. Here also, the final distribution looks a lot like the one shown on the right of Figure 14.4. Although, careful examination would seem that the distribution is comparatively slightly more stretched in the diagonal direction and the tails are somewhat more pronounced.

Finally, the convergence history of this optimisation process is shown in Figure 14.8. The immediate stunner here is that it only took four iterations to reach this state, about three times fewer than the L^2-distance minimisation. However, each iteration is more expensive as it involves running GMRES. All in all, the moments optimisation is decidedly faster. In terms of computation time, the difference-of-moments minimisation took about half of the time required by the previous example. At the same time, also here, a reduction of roughly 80%

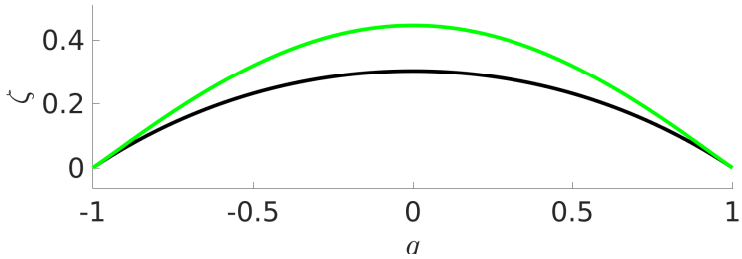


Figure 14.6: Initial lens surface (black) and optimised lens surface(green) using difference-of-moments objective.

of the initial objective function is attained.

14.4 Final remarks on freeform optimisation

In this chapter, we've looked at the optimisation of freeform surfaces by reinterpreting the gradient of the surface as an impulsive control to Liouville's equation. In this reinterpretation, the shape of an optical surface acts like an open loop controller on the light distribution, or alternatively, a large set of rays. Our example consisted of emulating an ideal lens for a particular input distribution, which was chosen a Gaußian for convenience. The most remarkable result of this chapter is that the minimisation of the difference-of-moments seems to be the quicker choice by far. The results compared with L^2-distance minimisation are strikingly similar, yet those results are obtained with fewer iterations. More research is needed to see if this is not just a fluke.

We've focussed on freeform refractive surfaces, although the theory is of course equally applicable to reflector design. For instance, if the reflector design problem were to be solved with the discontinuous Galerkin method, a moving mesh would need to be used, see Section 7.6. The objective function would need to be slightly altered as well, in particular the term that enforces Snell's law in (14.8) would need to be replaced with a term that enforces the law of specular reflection along the mirror.

Another point of note is that the theory we've shown here and in Chapter 12 is formulated on the exact level, i.e., in terms of PDEs. Potentially, therefore,

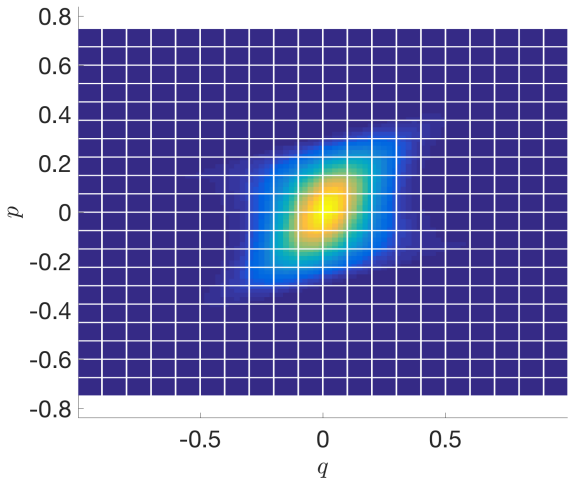


Figure 14.7: Final distribution for the ideal lens example using the difference-of-moments objective.

any solver could be used to solve Liouville's equation, even ray tracing[2]. As mentioned in Chapter 10, the discontinuous Galerkin method works well for smooth solutions, which implies nonzero étendue distributions. Although in practice all light sources have nonzero étendue, some approximate point sources very well indeed. Thus, our theory could be adapted to solve zero étendue problems by using ray tracing instead of the Liouville solver. Another possibility is that the theory can also deal with irradiance or intensity optimisation. This is, in principle, easily achieved by using a properly defined objective.

Philosophically, the method exhibited here is quite appealing, as it's closer to what an optical engineer tends to do in comparison with, for instance, Monge-Ampère based methods. Indeed, to put it bluntly, an optical engineer uses a trial-and-error approach, where the corrections are (highly) educated guesses. Of course, some intuition and deeper understanding comes into play, but at a coarse grain at least, our method does the same, only with mathematically precise corrections. However, optical engineers will always be necessary to guide the process. Just consider that our method looks, rather blindly, for local minima of the objective. An engineer could use it as an enlightening tool to find

[2]Imagine a slight jeering chuckle.

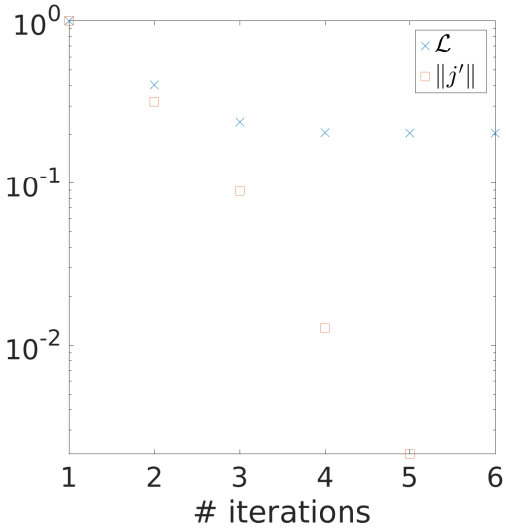


Figure 14.8: Convergence history for the Lagrangian (blue) and the norm of the reduced gradient (red) using difference-of-moments objective.

global minima.

Part V

Final remarks

Chapter 15

Suggestions for future research

> Segui il tuo corso, e lascia dir le genti.
>
> Dante Alighieri

Throughout this work, we've dropped various hints and made numerous suggestions for future research. However, not all ideas fit naturally into the flow of the story told so far. In this chapter, therefore, we discuss some ideas for future investigations. As always with research, there was too little time to try it all.

15.1 General discontinuous power redistribution

Throughout this work, we've used Snell's law as the rule to redistributing power across an interface. Of course, this device may be applied with any redistribution rule as long as it uniquely defines the energy flow.

Fresnel's equations give a physically accurate expression for the power transmission and reflection according to angle and polarisation of the light. For all but some special angles, such as the Brewster angle, a ray splits up into a reflected and transmitted ray. Both carry off some of the incident energy. This could easily be implemented in our Liouville solvers. Instead of drawing en-

ergy flux from one momentum in phase space, the flux would be drawn from two different momenta, each energy contribution weighted by Fresnel's expressions. Each polarisation direction would require its own brightness distribution independently satisfying Liouville's equation.

Another example of discontinuous power redistribution is scattering phenomena, such as Mie scattering or scattering due to surface roughness. These examples are more complicated than the Fresnel equations, but the same principle holds: energy flux is drawn from a range of momenta, each weighted accordingly. Provided the redistribution rule is known, this can be incorporated into a Liouville solver.

15.2 Dealing with the Fresnel tree

As mentioned in the previous section, a physical ray splits into two each time an interface is encountered. The transmitted and reflected rays each carry off some of the energy of the incident ray. For accurate ray tracing, of course, this implies that both rays have to be followed through the optic. However, this splitting happens at each interface, resulting in an exponential growth of the number of rays that need to be traced. This has the structure of a binary tree, dubbed the Fresnel tree. There are two ways to deal with this structure, either set a depth limit or an energy limit. The first is straightforward, the second needs to keep track of the energy carried by a ray, disregarding it when the energy falls below a certain threshold.

In the context of Liouville's equation, such complications appear as well, though in a slightly more manageable form. Think, for instance, of a series of lens-like surfaces. At each interface, part of the brightness is reflected and part of it is transmitted. However, unlike in ray tracing, the exponential growth of the number of cases can be avoided by exploiting the linearity of Liouville's equation. At each interface, we simply have to store the reflected brightness distribution. We can then sweep the optical system several times, at each interface adding the brightness that came from a reflection. By linearity of Liouville's equation, this leads to the correct solution. Like this, the Fresnel tree can be explored much faster by sweeping forward and backward along the optical axis several times. Similar to the ray tracing approach, a stopping criterion is needed to decide on the depth of the exploration. By analogy, this can be done either by setting a fixed number of sweeps or an energy threshold.

15.3 Tensor meshes

Throughout this work, we've looked at two-dimensional optics, which provides a simplified setting. The full problem is of course three-dimensional optics, which exhibits a four-dimensional phase space. In theory, the discontinuous Galerkin method is easily adapted since extra dimensions are added by dyadic products on the elements. This same strategy can be applied to construct four-dimensional meshes. This makes sense, since whenever the refractive index field is constant, phase space has a product structure as well. In particular, the spatial part is somewhat free while the momentum part is then always Descartes' disc. Naturally, when the refractive index field is piecewise constant, this construction can be applied to each piece. In the GRIN case, the curved mesh technique presented in Chapter 13 can be used.

15.4 Light meshes

As mentioned in Chapter 10, the one flaw of spectral methods is that they only work really well for smooth solutions. Once discontinuities are present in the solution, their uses are limited. Since Liouville's equation is linear, the only way that a discontinuity can appear that isn't aligned with the mesh is when the initial conditions aren't smooth. A possible way to deal with this is to let parts of the mesh move exactly along with the solution itself, much like front tracking, tying some nodes to rays, as it were. This, of course, leads to complications near optical interfaces, but the possibility is not to be dismissed.

15.5 Modelling choices

For simplicity, we've used some tacit assumptions that don't necessarily hold in practice. For instance, we've assumed monochromatic light and completely transmissive media. These assumptions are easily fixed, but for our purposes they would have only complicated matters.

15.5.1 Colour

Colour can be added in several ways, depending on the colour spectrum of the light source. Some sources have a rather discrete spectrum of several sharp peaks. For such sources, one might reasonably solve Liouville's equation for each of these colour peaks separately. If the spectrum is continuous, the alternative

is to introduce another dimension over which is no energy transmission occurs, barring the presence of nonlinear optical effects. When only reflective optics are used, which is quite common in lighting applications, each light source can be modelled by a separate brightness. This is due to the fact that reflections are independent of the frequency of light.

15.5.2 Diffusive and dark materials

Diffusive optical media are fairly common in lighting, just look at your ceiling fixture, there's a good chance it has a dome with a frosted glass surface. In diffusive materials, rays are scattered many times, so that their paths are effectively described by random walks [175]. Combined with the Feynman-Kac formula, which states that random walks are the characteristics of a diffusion equation, this leads to a diffusion term in Liouville's equation [176]. Materials can also be partly absorptive, like sunglasses or stained glass windows. In linear media, attenuation is linear to the incident power [6]. Of course, both phenomena depend on the travelled path length instead of the length down the optical axis. Therefore, if we wish to include these effects into Liouville's equation, we end up with something like

$$\frac{\partial \rho}{\partial z} + \nabla \cdot (\rho \boldsymbol{S} \nabla h) = \frac{\mathrm{d}s}{\mathrm{d}z}\big(\nabla \cdot (\boldsymbol{D} \nabla \rho) - \mu \rho\big), \tag{15.1}$$

where μ is the attenuation rate, $\boldsymbol{D}$ the diffusion tensor and s is the arc length. The relation between s and z is discussed in Section 1.2.4, and it was found that $\frac{\mathrm{d}s}{\mathrm{d}z} = \frac{n}{\sqrt{n^2 - \|\mathbf{p}\|^2}}$ for forward propagating rays.

15.6 Extending active flux to higher order

The active flux scheme as presented in Chapter 6 supplies third-order accuracy. It is, at the moment, unclear how to extend the method to higher-order consistently. Attempts have been made by adding more boundary points, but the order of approximation doesn't line up with the order of the flux [112]. Our suggestion would be to add internal nodes to each element. The natural extension, we believe, to the active flux scheme uses the Fekete points on the triangle [177–179].

The Fekete points are closely related to polynomial interpolation. Indeed, one way to define them is to maximise the Vandermonde determinant induced by the set of points. Fekete points are commonly labelled by the degree of

polynomial that they support. Fekete points on the triangle can be efficiently computed. Furthermore, the Fekete points of degree d have $d+1$ points on each boundary segment which are the Gauß-Lobatto points. The points used in Chapter 6 would be labelled $d = 2$.

For any polynomial degree higher than 2, the Fekete points also include points interior of the triangle. For instance, the $d = 3$ Fekete points on an equilateral triangle have four Gauß-Lobatto points on each side with the centre point of the triangle included, see Figure 15.1.

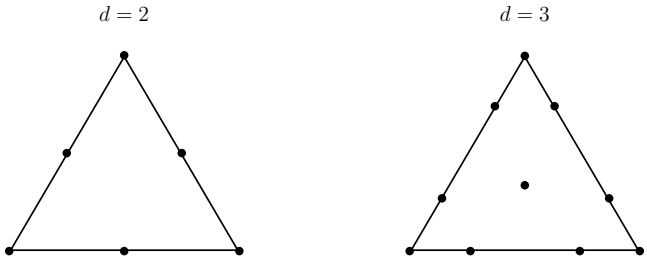


Figure 15.1: Fekete points of order 2 and 3.

For any degree d, the number of Fekete points corresponds to the number of coefficients of a two-dimensional polynomial, which is $\frac{1}{2}(d+1)(d+2)$. Note that in the third-order active flux scheme, we have 6 coefficients, fixing a quadratic, while the average value has to be accommodated some way. In general, given the Fekete points of degree d, a polynomial of the same degree is fixed, which does not necessarily have an average value equal to some given average value. Therefore, a higher-order bubble function is needed, for instance

$$\varphi = C_d(b_1 b_2 b_3)^k, \tag{15.2}$$

with $3k > d$, where b_i, $i = 1, 2, 3$ are the barycentric coordinates, which can all three be represented by degree one polynomials, making the proposed bubble function (15.2) of degree $3k$.

15.7 Zero-étendue freeform optimisation

As already mentioned in Chapter 14, it may well be possible that our method for optimising freeform surfaces can be adapted to zero-étendue problems. This has, in part, to do with the fact that we've used the optimise-then-discretise

approach to finding the optimality system. Thus, the optimality system can be solved using any numerical method of our choice. The methods we've presented in Part III all work assuming nonzero étendue. However, ray tracing engages with individual rays and as such works rather well for zero-étendue problems. This reasoning suggests we use ray tracing to solve Liouville's equation when dealing with zero-étendue freeform optimisation problems. In all likeliness, it will be necessary to reinterpret ρ as the density of a measure and apply some measure theory in concordance with functional minimisation.

15.7.1 The French connection

In Chapter 14, we've also briefly alluded to the correspondence between optimal transport and our work through the Benamou-Brenier formula [174]. Although we've only used it loosely to draw an analogy, we believe it can be made more rigorous. However, our suspicion is that this rigorous correspondence between optimal transport and our work only holds for zero-étendue problems. And even then quite possibly only in even more restrictive special cases. Our belief is based on the observation that when dealing with extended sources, at any one point on the surface there's a whole range of angles coming in. As a consequence, it's in general not possible to attain the target distribution, which is a requirement in optimal transport theory. For zero-étendue problems there is in fact a one-to-one mapping between rays coming from the source and positions on the freeform surface. This one-to-one mapping means the freeform design problem reduces to an assignment problem, which is probably equivalent to an optimal transport problem.

15.8 Solving the eikonal equation

One of the most pernicious objections raised against phase space methods is that they are useless by default. Engineers and physicists use the lower-dimensional distributions intensity and illuminance anyway, so why bother? Hopefully, we've managed to argue throughout this work that phase space methods are quite useful and interesting indeed. But, if the objections persist, other methods can be explored. Our suggestion would be to look at numerical methods for the eikonal equation, given by (1.9), i.e.,

$$\|\nabla S\| = n. \tag{15.3}$$

Rays, the pathways of luminous power, can simply defined as the integral curves of ∇S and the spatial part of the Poynting vector $\vec{P}$ can be written as $\|\vec{P}\|\nabla S$. A bit of mathematics using Maxwell's equations reveals that

$$\nabla \cdot \Big(\|\vec{P}\| \nabla S \Big) = 0, \tag{15.4}$$

see e.g. Rubinstein and Wolansky [180]. Thus, the eikonal equation together with (15.4) provide a system of equations to determine the energy distribution of a light source in real space. The benefit compared with Liouville's equation is that it doesn't operate on phase space at all. The disadvantage is that the eikonal equation is a nonlinear PDE, which is as a rule of thumb harder to solve.

15.9 Cautionary advise

Most of the ideas presented here are rather straightforward extensions of the work in this thesis, although surprises always lurk around the corner. Some things can be foreseen as easy modifications, such as implementing the Fresnel equations[1] or dealing with different modelling choices. Other suggestions may be harder to implement or require additional work, the tensor mesh seems to be of such a kind. The cautionary advise therefore is not to think too lightly of undertaking any of the suggested projects, but approach it as a fresh problem.

[1] I've actually implemented Fresnel's equations already in the upwind solver, but the results didn't add anything to those already presented in Chapter 8.

Chapter 16

Conclusion

> You see, a guy called William Shakesman once said: "Brevity is the soul of wit." This just means don't waste my time.
>
> Mr. Plinkett

In this thesis, we tried to gain some basic understanding of a new field we dubbed *computational illumination optics*. We delved into the very foundations of optics in electromagnetism, looked at ray tracing techniques and introduced a new paradigm by shifting to Liouville's equation. A rather wide range of topics has come by, which we'll now review.

In the first part of this work, we discussed the classical approach to illumination optics. We've focussed on the Hamiltonian formulation throughout this work. Hamilton's equations provide the governing equations of a light ray in terms of a first-order system of ODEs. This is achieved by introducing a higher-dimensional space called phase space, consisting of all positions and all momenta. Phase space provides a full description of geometric optics, a single point is enough to determine an entire ray. Conversely, the brightness distribution, the luminous power per unit area per solid angle, is the power distribution on phase space and therefore contains everything there is to know about a luminaire geometrically. This as opposed to the illuminance, power per unit area, or the luminous intensity distribution, power per solid angle. Even if both illuminance and intensity are known, this does not provide complete information on a luminaire. Phase space therefore furnishes us with a tidy and compact

description of geometric optics.

After this brief introduction to Hamiltonian optics, we explored ray tracing techniques. Ray tracers, commercial or home-brewed, often have problems with finding the correct intersection point when the rays skim very closely to the surface. We advanced two main ideas for ray tracers: a new way of constructing bounding boxes that divides the surface into convex and concave pieces, and the philosophy of first finding a bracket on which there's certainly an intersection point before finding the root. Our contribution in this part was to identify several types of rays and propose algorithms designed to deal with them in the context of rotationally symmetric optics. The amalgamation of the proposed algorithms forms what we named the Magic Bullet algorithm. We furthermore proved that the Magic Bullet is guaranteed to find the correct intersection point in two-dimensional optics. For three-dimensional optics, the result is slightly weaker, but we identified rather precisely the rays for which the Magic Bullet fails. Our numerical results were very encouraging and show a significant increase in reliability compared to a more naive ray tracer.

Next, we investigated another essential component of any ray tracer, namely the root-finder. Finding roots of nonlinear functions is a problem that crops up everywhere in science and engineering. Grau-Sanchez et al. have introduced a clever reformulation of the root-finding problem as an ODE, after which any ODE solver can be converted to a root-finder [41–43]. Our main theoretical result here is a theorem that provides some fundamental barriers for root-finders based on linear multistep methods. Our practical contribution was to introduce *full* LMMs as root-finders. We discovered that methods that ordinarily don't work as ODE solvers do in fact work as root-finders. In particular, we showed that full LMMs, which are not zero-stable in general, produce excellent root-finders. Compared to Newton iteration, our method is significantly faster and more efficient. We also showed some pathological examples where Newton iteration doesn't converge for a certain range of starting guesses. In one case, Newton iteration diverges for all initial guesses except the exact root itself. In these pathological examples, our LMM-based methods extended the range of starting guesses and does indeed provide a finite range for the second problem. We also proposed a bracketed version based on the LMM root-finders that's guaranteed to converge given a suitable starting interval. This method can be seen as extending Brent's method using derivatives.

After this brief foray into the world of ray tracers, we decided to break away from the paradigm entirely. Instead, we proposed using an Eulerian description of illumination optics, realising that ray tracing can be interpreted as the Lagrangian picture. This, we believe, is the most important new concept in our

work. Hence, from ray tracing we came to Liouville's equation, a hyperbolic PDE, as the central object of our fascination. Thereafter, we devoted a part of this work to the dissemination of basic discretisation schemes for hyperbolic equations in their native environment. In particular, we discussed the upwind and WENO methods, the active flux scheme and the discontinuous Galerkin spectral element method. Along the way, we managed to introduce a new type of WENO method that we called embedded WENO. The novel idea here is to improve control over the numerical approximation when discontinuities are present in the solution. The result is a new variant of WENO that produces equally good or better results while using the same amount of computational effort. Our contribution to active flux schemes was to formulate them on a moving mesh in a different way than was previously known. The existing technique is to use the active flux scheme as a space-time discretisation technique, where our novel method is a semi-discretisation, leaving time continuous. This can have some advantages as any numerical integrator can be applied. Our intention was to adapt the aforementioned numerical discretisation techniques to solve Liouville's equation. However, WENO didn't make the cut, as it turned out to be too complicated to adapt.

Consequently, we discussed at length on how to convert these basic schemes to Liouville solvers. We stress that the entire concept of solving Liouville's equation in geometrical optics is novel. Each solver comes with advantages and disadvantages. For instance, the upwind method is highly robust, but turned out to be rather hard to adapt. To do so required us to find an extra jump condition on the gradient of the brightness distribution. The upwind solver we proposed uses this jump condition to find a correct finite difference across an optical interface. The active flux scheme is a third-order method that uses local ray tracing. We showed that it makes an excellent Liouville solver and demonstrated the correctness of our proposed method for moving meshes. DG spectral element methods are quick and efficient and they provide superior performance for smooth solutions. Both the active flux scheme and the DG spectral element method were relatively easy to adapt, as we discovered that all that's really needed is a proper alignment of the mesh with any interfaces. We also introduced a technique to solve Liouville's equation in the presence of lenses and other freeform refractive surfaces. All the Liouville solvers were compared to Monte Carlo ray tracing and we saw a general trend that Liouville solvers can attain much higher accuracies in similar or shorter times. For the simple test cases we looked at, they're without a doubt much more efficient solvers for geometric optics problems. Their only downside is perhaps that they're harder to implement. However, this is rather subjective, we didn't encounter

any difficulties in implementation.

Finally, we come to the justification and vengeance of the Eulerian methods with the introduction of optimal control theory to geometric optics. The speed and precision of the DG method were put to good use. The novelty we introduced is to reinterpret the optical design problem as a feed-forward control problem acting on Liouville's equation, or alternatively, on an ensemble of rays. Then we applied the theory of optimal control on PDEs to produce a method that computes locally optimal optics. We demonstrated that both GRIN and freeform surface optimisation can be handled. Moreover, our method optimises on phase space and works for nonzero étendue problems. This as opposed to, e.g., Monge-Ampère based methods that only work with the lower-dimensional distributions and for zero étendue. Our numerical experiments involved optimising the optics in such a way as to produce an output close to a desired output. The closeness was measured using both the L^2-distance and the square of the difference of moments. All examples showed a decrease in the objective value of over 75%. However, they also show that the optimised optic is somewhat sensitive to which objective is used. We believe the framework we proposed has an awesome potential, as in principle any design constraint can be handled and any number of control inputs can be used. It will be up to the mathematicians and engineers down the road to explore the full range of possibilities.

Appendix A

Critique of symplectic integrators in geometric optics

Symplectic integrators are methods that are especially designed to numerically solve Hamilton's equations in a way that preserves structure on phase space [31]. One property of such integrators is that an approximate Hamiltonian is exactly preserved, provided the original Hamiltonian is preserved. It's a natural and sound idea to preserve as many properties of the physical system as possible.

In high-dimensional physical systems, such as many-particle systems, a constant Hamiltonian defines a hyperplane in phase space [12]. Often though, such systems are also chaotic, for instance in astronomy and molecular dynamics. The hyperplane defined by the constant Hamiltonian therefore defines the physical states that the system can reach from the initial condition.

Especially when integration happens over relatively long periods, it's useful to be able to constrain the system to physically relevant solutions. A simple example would be a satellite in orbit: the orbital energy is conserved when using a symplectic integrator, so at least the integrator keeps the satellite in roughly the correct orbit. Using other integrators would result in nonphysical energy loss or gain.

Thus, symplectic integrators are typically useful under two conditions:

1. The Hamiltonian is preserved in the original system.

2. Time integration is required over extensive periods.

However, in numerical analysis, like many other fields, there's no free lunch [181]. If your fancy integrator exactly preserves something, it will be worse in other ways. In terms of Fourier analysis, symplectic integrators exhibit no dissipation, but provide a large phase error [35]. Even in astronomy or molecular dynamics, whenever one of the above conditions isn't met, other integrators are used that are more well-rounded in terms of performance, such as the standard-issue RK4 method [182].

Typical problems in illumination optics satisfy neither condition. The Hamiltonian is usually not preserved in the illumination setting, see Section 1.2.4. This is only true when the refractive index doesn't depend on z, a rather rare situation. Typical illumination systems have curved mirrors or freeform surfaces, hence there's no axis along which the refractive index is constant. Finally, integration lengths are typically short. The aspect ratio of optics used in illumination systems is usually smaller than ten or so. Nowhere near the giga-year evolution problems astronomers would use a symplectic integrator for [183].

Using a symplectic integrator for illumination optics therefore only makes sense in one specific case: guides. Guides are defined as optics with an axis along which the refractive index profile is constant, e.g., fibre optics. Even then, some guides are quite short, such as the elliptical guide we've used as a lens in Chapter 13. Guides are sometimes employed in fibre optics communication, in which case we'd want to know how the light behaves over long distances. Hence, for fibre optics it would indeed make sense to use symplectic integrators. For other types of optics, however, using a symplectic integrator is simply unnecessary.

Appendix B

Matrix-free GMRES

We discuss here briefly the generalised minimal residual method (GMRES) and its matrix-free version for those who aren't intimately familiar with numerical linear algebra. For a detailed discussion of that topic, we refer the reader to the excellent textbook of Saad [155] and his original paper on GMRES [184].

GMRES is an iterative method for solving linear systems, i.e.,

$$A\mathbf{x} = \mathbf{b}, \tag{B.1}$$

where $A \in \mathbb{R}^{m\times m}$ is a large sparse matrix, the right-hand side $\mathbf{b} \in \mathbb{R}^m$ is a given vector and m is the dimension. Our aim is to iteratively find a vector $\mathbf{x} \in \mathbb{R}^m$ that satisfies the system (B.1). GMRES is a Krylov subspace method, which means a basis of a subspace is built by consecutive multiplications with A, so that at the nth iteration we have

$$K_n = \{\mathbf{b}, A\mathbf{b}, A^2\mathbf{b}, \ldots, A^{n-1}\mathbf{b}\}. \tag{B.2}$$

If we'd naively choose $\mathbf{b}, A\mathbf{b}, A^2\mathbf{b}, \ldots$ as our basis, it would quickly become close to linearly dependent, as the vectors converge to the eigenvector belonging to the largest eigenvalue in magnitude. To counter this, we orthogonalise each new vector that's added to the basis with respect to the ones we already have. This results in what's known as Arnoldi iteration, described in Algorithm 11.

Arnoldi iteration can be interpreted as the numerically stable Gram-Schmidt procedure where each new direction is obtained by a matrix multiplication with A. The projection coefficients $h_{j,k}$ are stored in an upper Hessenberg matrix $\tilde{H}_{n-1} \in \mathbb{R}^{n\times(n-1)}$. The basis vectors are arranged in a matrix $Q_n = [\mathbf{q}_1, \ldots, \mathbf{q}_n]$,

Algorithm 11 Arnoldi iteration

$\mathbf{q}_1 \leftarrow \frac{\mathbf{b}}{\|\mathbf{b}\|}$
for $k = 2, 3, \ldots, n$ **do**
  $\mathbf{q}_k \leftarrow A\mathbf{q}_{k-1}$
  for $j = 1$ to $k-1$ **do**
    $h_{j,k-1} \leftarrow \mathbf{q}_j^T \mathbf{q}_k$
    $\mathbf{q}_k \leftarrow \mathbf{q}_k - h_{j,k-1}\mathbf{q}_j$
  end for
  $h_{k,k-1} \leftarrow \|\mathbf{q}_k\|$
  $\mathbf{q}_k \leftarrow \frac{\mathbf{q}_k}{h_{k,k-1}}$
end for

so that $Q_n \in \mathbb{R}^{m\times n}$. By construction, the matrix Q_n is orthogonal, so that $Q_n^T Q_n = I_m$, where I_m is the $m \times m$ identity matrix. Moreover, from the Arnoldi iteration, we see that matrix Q_n satisfies the relation

$$AQ_n = Q_{n+1}\tilde{H}_n. \tag{B.3}$$

This observation together with the fact that Q_n is orthogonal form the foundation of GMRES.

As the name suggests, GMRES minimises the residual $\|A\mathbf{x} - \mathbf{b}\|$, where the minimisation is performed over the Krylov subspace K_n. Thus, as an approximation to $\mathbf{x}$, we set

$$\mathbf{x}_n = Q_n \mathbf{y}_n, \tag{B.4}$$

where $\mathbf{y}_n \in \mathbb{R}^n$. Clearly, as n approaches m, the Krylov subspace will grow until eventually we have $K_m = \mathbb{R}^m$ and the minimisation results in the exact solution. However, for $n < m$, we aim to find a $\mathbf{y}_n$ that minimises

$$\|A\mathbf{x}_n - \mathbf{b}\| = \|AQ_n\mathbf{y}_n - \mathbf{b}\| = \|Q_{n+1}\tilde{H}_n\mathbf{y}_n - \mathbf{b}\| = \|\tilde{H}_n\mathbf{y}_n - \beta\mathbf{e}_1\|, \tag{B.5}$$

where we've applied (B.3) and the fact that Q_{n+1} is orthogonal. We've also defined the constant $\beta = \|\mathbf{b}\|$ and $\mathbf{e}_1 \in \mathbb{R}^{n+1}$ is the vector where the first component is 1 and all others are 0. Thus, minimising the residual in the Krylov subspace reduces to the following minimisation problem,

$$\min_{\mathbf{y}_n \in \mathbb{R}^n} \|\tilde{H}_n\mathbf{y}_n - \beta\mathbf{e}_1\|. \tag{B.6}$$

This problem can be efficiently solved by keeping track of a QR-decomposition on $\tilde{H}_n$ during the Arnoldi iteration by means of Givens rotations. We'll not go

into those details here, but we urge the reader to pick up the aforementioned textbook if their interest is peaked.

B.1 A matrix-free version

Suppose now that, for whatever reason, we don't have access to the matrix A, but we know its effect on a vector input. In Chapters 13 and 14, we saw an example of this, where a linear operator was defined by the solution to a PDE. After some discretisation technique, the numerical solution can be represented as a vector, while the discretised linear operator remains linear. Hence, we're given a function $F : \mathbb{R}^m \to \mathbb{R}^m$, where F is linear so that $F(c\mathbf{v}) = cF(\mathbf{v})$ and $F(\mathbf{u} + \mathbf{v}) = F(\mathbf{u}) + F(\mathbf{v})$, for $c \in \mathbb{R}$ and $\mathbf{u}, \mathbf{v} \in \mathbb{R}^m$.

Now, we consider the linear system represented by

$$F(\mathbf{x}) = \mathbf{b}, \tag{B.7}$$

and we ask the question of how to solve it. Any method that calls the transpose matrix A^T won't work, since we don't have access to it. Therefore, methods like BiCGSTAB cannot solve (B.7). However, we note that GMRES only references A itself and at that only requires products of A with some vector. Hence, we can simply replace the matrix-vector product that's used in Arnoldi iteration to obtain a matrix-free version, resulting in Algorithm 12.

Algorithm 12 Matrix-free Arnoldi iteration

$\mathbf{q}_1 \leftarrow \frac{\mathbf{b}}{\|\mathbf{b}\|}$
for $k = 2, 3, \ldots, n$ **do**
  $\mathbf{q}_k \leftarrow F(\mathbf{q}_{k-1})$
  for $j = 1$ to $k - 1$ **do**
    $h_{j,k-1} \leftarrow \mathbf{q}_j^T \mathbf{q}_k$
    $\mathbf{q}_k \leftarrow \mathbf{q}_k - h_{j,k-1}\mathbf{q}_j$
  end for
  $h_{k,k-1} \leftarrow \|\mathbf{q}_k\|$
  $\mathbf{q}_k \leftarrow \frac{\mathbf{q}_k}{h_{k,k-1}}$
end for

The rest of matrix-free GMRES is exactly the same as the above version. In particular, Algorithm 12 still produces an orthogonal matrix Q_n and an upper Hessenberg matrix $\tilde{H}_{n-1}$. The relation between iterations (B.3) now translates

to

$$F(Q_n) = Q_{n+1}\tilde{H}_n, \tag{B.8}$$

provided we interpret $F(Q_n)$ as applying F column-wise to Q_n. Minimising the residual in the Krylov subspace K_n still entails (B.4), while (B.5) is slightly altered to read

$$\|F(\mathbf{x}_n) - \mathbf{b}\| = \|F(Q_n\mathbf{y}_n) - \mathbf{b}\| = \|F(Q_n)\mathbf{y_n} - \mathbf{b}\| = \|Q_{n+1}\tilde{H}_n\mathbf{y}_n - \mathbf{b}\|,$$

from which we again find that $\mathbf{y}_n$ should satisfy (B.6).

Summary

The research explores the applications of Hamiltonian optics on phase space. The focus is on nonimaging and illumination optics, i.e., the design and analysis of optical systems for illumination. The outset was to find a viable alternative to ray tracing methods, which are the current industry standard. The alternative examined in the work is aimed on Liouville's equation, a hyperbolic PDE. Ray tracing and Liouville's equation can be viewed as two sides of the same coin. Ray tracing represents the Lagrangian side of light while Liouville's equation is the corresponding Eulerian description. This may be interpreted providing local, Lagrangian, information versus global, Eulerian, information. In illumination optics, global information is usually desired, for instance the light distribution in a room.

The Eulerian approach is argued to be faster, though it sacrifices precise path information of individual rays. The catch with Liouville's equation, however, is that typical optical systems have discontinuous refractive index fields, e.g. lenses and mirrors. Hence, standard numerical solvers for hyperbolic PDEs will typically fail. In this work, several solvers are constructed to deal with these difficulties. The results show that if local information is not required, high-order solvers for Liouville's equation dramatically outperform ray tracing methods. They are both faster and more accurate by orders of magnitude. The Liouville solvers are able to provide a level of accuracy in seconds where ray tracing methods would take hours.

The upshot of the increase in speed and accuracy are new possibilities for applications. For example, return times for optical designers are shorter, allowing more iterations in the design process. The last part of the research looks into the design of optical systems by applying optimal control theory to Liouville's equation. The refractive index field or the shape of an optical interface is interpreted as an open-loop control input. This provides a broad paradigm for customisation of the full phase space distribution of a lighting system.

Acknowledgements

> What would you think if I sang out of tune? Would you stand up and walk out on me?
>
> Ringo Starr

Thanks everyone. That's all I can really say to all the people who supported me over the past four years. There are too many people to name, so allow me to provide a nonexhaustive list.

Jan and Wilbert, my supervisors, thank you for giving me as much freedom and support as you did. I always enjoyed our collaboration.

Jan and Max, my brothers, you were always there to help me along. And of course let's not forget our awesome roadtrip in Australia. Your girlfriends, Margot and Naomi, have become like sisters to me. They would always cheer me up.

Esther and Jan, my parents, thank you for all your support. I hope I've made you proud.

My friends and coworkers at CASA, thank you for all the helpful discussions. And thanks for all the fun, particularly at the pub quizzes.

My buddies at Tuna Ciudad de Luz, you've always provided the necessary distraction.

Finally, my lovely wife Lily, without you I wouldn't have been able to do this.

Curriculum Vitae

Bart van Lith was born on the 21st of August 1989 in Berlicum, the Netherlands. After finishing his VWO degree in 2006 at the Rodenborch College in Rosmalen, the Netherlands, he studied Physics and Mathematics at Eindhoven University of Technology. In 2013 he graduated within CASA and TPS on self-assembly of filaments. From December 2013 he started a PhD project at CASA of which the results are presented in this dissertation.

Bibliography

[1] R. Winston, J. C. Miñano, and P. Benítez, *Nonimaging Optics*. Elsevier, 2005.

[2] J. Chaves, *Introduction to non-imaging optics*. CRC Press, 2008.

[3] NASA, "International Space Station." `https://www.nasa.gov/mission_pages/station/main/index.html`. Online; accessed 23-03-2017.

[4] "NASA Earth Observatory." `https://www.earthobservatory.nasa.gov`. Online; accessed 21-9-2017.

[5] E. Hecht, *Optics*. Addison Wesley, 2002.

[6] D. J. Griffiths, *Introduction to electrodynamics*. Prentice Hall, 1999.

[7] J. C. Maxwell, "A dynamical theory of the electromagnetic field," *Philosophical Transactions of the Royal Society of London*, vol. 155, pp. 459 – 512, 1865.

[8] I. Tolstoy, *James Clerk Maxwell, a biography*. University of Chicago press, 1983.

[9] M. Born and E. Wolf, *Principles of Optics*. Pergamon Press, 4th ed., 1970.

[10] J. A. Sethian, "A fast marching level set method for monotonically advancing fronts," *Proc. Natl. Acad. Sci. USA*, vol. 93, pp. 1591 – 1595, 1996.

[11] W. R. Hamilton, "On a general method in dynamics," *Philosophical Transactions of the Royal Society*, pp. 247 – 308, part II for 1834.

[12] V. I. Arnold, *Mathematical Methods of Classical Mechanics.* Springer-Verlag, 1978.

[13] M. Giaquinta and S. Hildebrandt, *Calculus of Variations I.* Springer, 2004.

[14] M. Giaquinta and S. Hildebrandt, *Calculus of Variations II.* Springer, 2004.

[15] T. Petrila and D. Trif, *Basics of fluid mechanics and introduction to computational fluid dynamics.* Springer, 2005.

[16] K. B. Wolf, *Geometric Optics on Phase Space.* Springer, 2004.

[17] Abu Sa'd al-'Ala' Ibn Sahl, *On Burning Mirrors and Lenses.* 984.

[18] R. Rashed, "A pioneer in anaclastics: Ibn Sahl on burning mirrors and lenses," *Isis*, vol. 81, no. 3, pp. 464 – 491, 1990.

[19] K. Iizuka, *Engineering Optics.* Springer-Verlag, 1983.

[20] R. S. Fishman, "Perish, then publish: Thomas Harriot and the sine law of refraction," *Arch Ophthalmol.*, vol. 118, no. 3, pp. 405 – 409, 2000.

[21] J. A. Volgraff, "Snellius' notes on the reflection and refraction of rays," *Osiris*, vol. 1, pp. 718 – 725, 1936.

[22] R. Descartes, *La dioptrique.* 1637.

[23] A. S. Glassner, *An Introduction to ray tracing.* London: Academic Press, 1991.

[24] P. Shirley, *Realistic ray tracing.* A.K. Peteres, 2000.

[25] C. Filosa, J. H. M. ten Thije Boonkkamp, and W. L. IJzerman, "Ray tracing method in phase space for two-dimensional optical systems," *Applied Optics*, vol. 55, no. 13, pp. 3599 – 3606, 2016.

[26] J. Saarinen and R. van de Vrie, "Printed optics usher in new era of manufacturing," *EuroPhotonics*, 2014.

[27] P. J. Wang, J. A. Yeh, W. Y. Hsu, Y. C. Cheng, W. Lee, N. H. Wu, and C. Y. Wu, "Study of 3d printing method for GRIN micro-optics devices," *Proc. SPIE*, vol. 9759, pp. 9759 – 9759, 2016.

[28] A. I. Hernandez-Serrano, M. Weidenbach, S. F. Busch, M. Koch, and E. Castro-Camus, "Terahertz GRIN lenses fabricated by 3d-printing," in *2016 41st International Conference on Infrared, Millimeter, and Terahertz waves (IRMMW-THz)*, pp. 1–2, 2016.

[29] D. T. Nguyen, C. Meyers, T. D. Yee, N. A. Dudukovic, J. F. Destino, C. Zhu, E. B. Duoss, T. F. Baumann, T. Suratwala, J. E. Smay, and R. Dylla-Spears, "3d-printed transparent glass," *Advanced Materials*, vol. 29, no. 26, p. 1701181, 2017.

[30] D. Schut, C. Dupuy, and J. Harmon, "Inks for 3d printing gradient refractive index (GRIN) optical components," Nov. 6 2014. WO Patent App. PCT/US2014/036,660.

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

[32] H. Yoshida, "Construction of higher order symplectic integrators," *Phys. Lett. A*, vol. 150, no. 5,6,7, pp. 262–268, 1990.

[33] J. M. Sanz-Serna, "Runge–Kutta schemes for Hamiltonian systems," *BIT*, vol. 28, pp. 877–833, 1988.

[34] R. Chan, H. Liu, and G. Sun, "Efficient symplectic Runge–Kutta methods," *Applied Mathematics and Computation*, vol. 172, pp. 908 – 924, 2006.

[35] J. C. Butcher, *The Numerical Analysis of Ordinary Differential Equations*. Wiley, 1987.

[36] A. J. Dragt, "Lie algebraic theory of geometrical optics and optical aberrations," *J. Opt. Soc. Am.*, vol. 72, no. 3, 1982.

[37] A. J. Dragt and J. M. Finn, "Lie series and invariant functions for analytic symplectic maps," *J. Math. Phys.*, vol. 17, no. 12, p. 2215, 1976.

[38] B. Bolzano, *Rein analytischer Beweis des Lehrsatzes, daß zwischen je zwei Werthen, die ein entgegengesetztes Resultat gewähren, wenigstens eine reelle Wurzel der Gleichung liege*. Gottlieb Haase, 1817.

[39] S. B. Russ, "A translation of Bolzano's paper on the intermediate value theorem," *Hist. Math.*, vol. 7, pp. 156 – 185, 1980.

[40] W. Gautschi, *Numerical Analysis*. Birkhäuser, Boston, 2012.

[41] M. Grau-Sánchez, M. Noguera, and J. L. Díaz-Barrero, "Adams-like techniques for zero-finder methods," *Applied Mathematics and Computation*, vol. 211, pp. 130–136, 2009.

[42] M. Grau-Sánchez and J. L. Díaz-Barrero, "Zero-finder methods derived using Runge–Kutta techniques," *Applied Mathematics and Computation*, vol. 217, pp. 5366–5376, 2011.

[43] M. Grau-Sánchez and J. M. Gutiérrez, "Zero-finder methods derived from Obreshkov's techniques," *Applied Mathematics and Computation*, vol. 215, no. 8, pp. 2992 – 3001, 2009.

[44] R. P. Brent, *Algorithms for minimization without derivatives*. Prentice-Hall, 1973.

[45] I. Newton, *Methodus fluxionum et serierum infinitarum, cum eisudem applicatione ad curvarum geometriam*. 1744.

[46] E. Halley, "Methodus nova accurata & facilis inveniendi radices aequationum quarumcumque genereliter, sine praevia reductionei," *Philosophical Transactions of the Royal Society of London*, vol. 18, pp. 136 – 148, 1694.

[47] H. T. Kung and J. F. Traub, "Optimal order of one-point and multipoint iteration," *J. Assoc. Comput. Math.*, vol. 21, pp. 643 – 651, 1974.

[48] P. Jarratt, "Some efficient fourth order multipoint methods for solving equations," *BIT Numerical Mathematics*, vol. 9, pp. 119–124, Jun 1969.

[49] N. Osada, "An optimal multiple root-finding method of order three," *Journal of Computational and Applied Mathematics*, vol. 51, no. 1, pp. 131 – 133, 1994.

[50] W. Bi, Q. Wu, and H. Ren, "A new family of eighth-order iterative methods for solving nonlinear equations," *Applied Mathematics and Computation*, vol. 214, no. 1, pp. 236 – 245, 2009.

[51] L. Shengguo, L. Xiangke, and C. Lizhi, "A new fourth-order iterative method for finding multiple roots of nonlinear equations," *Applied Mathematics and Computation*, vol. 215, no. 3, pp. 1288 – 1292, 2009.

[52] F. Soleymani, "Regarding the accuracy of optimal eighth-order methods," *Mathematical and Computer Modelling*, vol. 53, no. 5, pp. 1351 – 1357, 2011.

[53] M. Sharifi, D. Babajee, and F. Soleymani, "Finding the solution of nonlinear equations by a class of optimal methods," *Computers & Mathematics with Applications*, vol. 63, no. 4, pp. 764 – 774, 2012.

[54] R. Behl, A. Cordero, S. S. Motsa, and J. R. Torregrosa, "Construction of fourth-order optimal families of iterative methods and their dynamics," *Applied Mathematics and Computation*, vol. 271, pp. 89 – 101, 2015.

[55] T. Lotfi, S. Sharifi, M. Salimi, and S. Siegmund, "A new class of three-point methods with optimal convergence order eight and its dynamics," *Numerical Algorithms*, vol. 68, pp. 261–288, Feb 2015.

[56] Y. I. Kim, R. Behl, and S. S. Motsa, "An optimal family of eighth-order iterative methods with an inverse interpolatory rational function error corrector for nonlinear equations," *Mathematical Modelling and Analysis*, vol. 22, no. 3, pp. 321–336, 2017.

[57] A. Quarteroni, R. Sacco, and F. Saleri, *Numerical Mathematics*. Springer, 2007.

[58] J. Raphson, *Analysis Aequationum Universalis*. 1690.

[59] T. J. Ypma, "Historical development of the Newton–Raphson method," *SIAM Review*, vol. 37, no. 4, pp. 531–551, 1995.

[60] T. J. Dekker and W. Hoffmann, *ALGOL 60 procedures in numerical algebra*. Mathematisch Centrum Amsterdam, 1968.

[61] R. A. Adams, *Calculus: a complete course*. Pearson, 2013.

[62] Y. H. Geum and Y. I. Kim, "A family of optimal sixteenth-order multipoint methods with a linear fraction plus a trivariate polynomial as the fourth-step weighting function," *Computers & Mathematics with Applications*, vol. 61, no. 11, pp. 3278 – 3287, 2011.

[63] G. C. Donovan, A. R. Miller, and T. J. Moreland, "Pathological functions for Newton's method," *The American Mathematical Monthly*, vol. 100, no. 1, pp. 53–58, 1993.

[64] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*. Springer, 2004.

[65] M. Lentine, J. T. Grétarsson, and R. Fedkiw, "An unconditionally stable fully conservative semi-Lagrangian method," *J. Comput. Phys.*, vol. 230, no. 8, pp. 2857 – 2879, 2011.

[66] J. W. Gibbs, "On the fundamental formula of statistical mechanics, with applications to astronomy and thermodynamics," *Proceedings of the American Association for the Advancement of Science*, vol. 33, pp. 57–58, 1884.

[67] J. Liouville, "Note sur la théorie de la variation des constantes arbitraires," *Journal de Mathematiques Pures et Appliquées*, vol. 3, pp. 342 – 349, 1838.

[68] D. Rausch and A. M. Herkommer, "Phase space transformations - a different way to understand free-form optical systems," in *Renewable Energy and the Environment Congress*, 2013.

[69] D. Rausch and A. M. Herkommer, "Phase space approach to the use of integrator rods and optical arrays in illumination systems," *Advanced Optical Technologies*, vol. 1, no. 1-2, pp. 69 – 78, 2012.

[70] R. M. M. Mattheij, S. W. Rienstra, and J. H. M. ten Thije Boonkkamp, *Partial Differential Equations: Modeling, Analysis, Computation*. SIAM, 2005.

[71] L. C. Evans, *Partial Differential Equations*. American Mathematical Society, 2010.

[72] R. Courant, E. Isaacson, and M. Rees, "On the solution of nonlinear hyperbolic differential equations by finite differences," *Communications on Pure and Applied Mathematics*, vol. V, pp. 243 – 255, 1952.

[73] R. Courant, K. Friedrichs, and H. Lewy, "Über die partiellen Differenzengleichungen der mathematischen Physik," *Mathematische Annalen*, vol. 100, no. 1, pp. 32 – 74, 1928.

[74] S. K. Godunov, "A difference scheme for numerical computation of discontinuous solutions of the equations of hydrodynamics," *Math. Sbornik*, vol. 47, pp. 271 – 306, 1959.

[75] B. S. van Lith, J. H. M. ten Thije Boonkkamp, and W. L. IJzerman, "Embedded WENO: a design strategy to improve existing WENO schemes," *J. Comput. Phys.*, vol. 330, pp. 529 – 549, 2016.

[76] A. Harten and S. Osher, "Uniformly high-order accurate nonoscillatory schemes. I," *SIAM J. Numer. Anal.*, vol. 24, no. 2, pp. 279–309, 1987.

[77] X.-D. Liu, S. Osher, and T. Chan, "Weighted essentially non-oscillatory schemes," *J. Comput. Phys.*, vol. 115, no. 1, pp. 200 – 212, 1994.

[78] T. J. Barth and H. Deconinck, eds., *High-Order Methods for Computational Physics*, ch. High order ENO and WENO schemes for computational fluid dynamics by C.W. Shu, pp. 439 – 582. Springer, 1999.

[79] J.-M. Qiu, C.-W. Shu, L.-L. Feng, and L.-Z. Fang, "A WENO algorithm for the radiative transfer and ionized sphere at reionization," *New Astronomy*, vol. 12, no. 1, pp. 1–10, 2006.

[80] S. Amat, S. Busquier, and J. C. Trillo, "On multiresolution schemes using a stencil selection prodecure: Applications to ENO schemes," *Numer. Algorithms*, vol. 44, pp. 45 – 68, 2007.

[81] K. Siddiqi, B. B. Kimia, and C.-W. Shu, "Geometric shock-capturing ENO schemes for subpixel interpolation, computation and curve evolution," *Graphical Models and Image Processing*, vol. 59, pp. 278 – 301, 1997.

[82] G.-S. Jiang and C.-W. Shu, "Efficient implementation of weighted ENO schemes," *J. Comput. Phys.*, vol. 126, pp. 202 – 228, 1996.

[83] A. K. Henrick, T. D. Aslam, and J. M. Powers, "Mapped weighted essentially non-oscillatory schemes: achieving optimal order near critical points," *J. Comput. Phys.*, vol. 207, pp. 542 – 567, 2005.

[84] H. Feng, F. Hu, and R. Wang, "A new mapped weighted essentially non-oscillatory scheme," *J. Sci. Comput.*, vol. 51, no. 2, pp. 449 – 473, 2012.

[85] R. Borges, M. Carmona, B. Costa, and W.-S. Don, "An improved weighted essentially non-oscillatory scheme for hyperbolic conservation laws," *J. Comput. Phys.*, vol. 227, pp. 3191 – 3211, 2008.

[86] M. Castro, B. Costa, and W. S. Don, "High order weighted essentially non-oscillatory WENO-Z schemes for hyperbolic conservation laws," *J. Comput. Phys.*, vol. 230, no. 5, pp. 1766 – 1792, 2011.

[87] F. Aràndiga, M. Martí, and P. Mulet, "Weights design for maximal order WENO schemes," *J. Sci. Comput.*, vol. 60, pp. 641 – 659, 2014.

[88] Y. Ha, C. H. Kim, Y. J. Lee, and Y. Yoon, "An improved weighted essentially non-oscillatory scheme with a new smoothness indicator," *J. Comput. Phys.*, vol. 232, no. 1, pp. 68 – 86, 2013.

[89] S. Zhao, N. Nardjane, and I. Fedioun, "Comparison of improved finite-difference WENO schemes for the implicit large eddy simulation of turbulent non-reacting and reacting high-speed shear flows," *Computers & Fluids*, vol. 95, pp. 74 – 87, 2014.

[90] N. K. Yamaleev and M. H. Carpenter, "Third-order energy stable WENO scheme," *J. Comput. Phys.*, vol. 228, no. 8, pp. 3025 – 3047, 2009.

[91] N. K. Yamaleev and M. H. Carpenter, "A systematic methodology for constructing high-order energy stable WENO schemes," *J. Comput. Phys.*, vol. 228, no. 11, pp. 4248 – 4272, 2009.

[92] X. Hu, Q. Wang, and N. Adams, "An adaptive central-upwind weighted essentially non-oscillatory scheme," *J. Comput. Phys.*, vol. 229, no. 23, pp. 8952 – 8965, 2010.

[93] L. Fu, X. Y. Hu, and N. A. Adams, "A family of high-order targeted ENO schemes for compressible-fluid simulations," *J. Comput. Phys.*, vol. 305, pp. 333 – 359, 2016.

[94] F. Jia, Z. Gao, and W. S. Don, "A spectral study on the dissipation and dispersion of the WENO schemes," *J. Sci. Comput.*, vol. 63, no. 1, pp. 49 – 77, 2015.

[95] M. P. Martin, E. M. Taylor, M. Wu, and V. G. Weirs, "A bandwidth-optimized WENO scheme for the effective direct numerical simulation of compressible turbulence," *J. Comput. Phys.*, vol. 220, no. 1, pp. 270 – 289, 2005.

[96] P. D. Lax, "Weak solutions of nonlinear hyperbolic equations and their numerical approximation," *Comm. Pure Appl. Math.*, vol. 7, pp. 159 – 193, 1954.

[97] P. Fan, Y. Shen, B. Tian, and C. Yang, "A new smoothness indicator for improving the weighted essentially non-oscillatory scheme," *J. Comput. Phys.*, vol. 269, pp. 329 – 354, 2014.

[98] S. Zhang and C.-W. Shu, "A new smoothness indicator for the WENO schemes and its effect on the convergence to steady state solutions," *J. Sci. Comput.*, vol. 31, pp. 273 – 305, 2007.

[99] S. Gottlieb and C.-W. Shu, "Total variation diminishing Runge–Kutta schemes," *Mathematics of Compuation*, vol. 67, no. 221, pp. 73 – 85, 1998.

[100] S. Gottlieb, C.-W. Shu, and E. Tadmor, "Strong stability-preserving high-order time discretization methods," *SIAM Review*, vol. 43, no. 1, pp. 89 – 112, 2001.

[101] R. Wang and R. J. Spiteri, "Linear instability of the fifth-order WENO method," *SIAM J. Numer. Anal.*, vol. 45, no. 5, pp. 1871 – 1901, 2007.

[102] J. Banks, T. Aslam, and W. Rider, "On sub-linear convergence for linearly degenerate waves in capturing schemes," *J. Comput. Phys.*, vol. 227, no. 14, pp. 6985 – 7002, 2008.

[103] B. S. van Lith, J. H. M. ten Thije Boonkkamp, and W. L. IJzerman, "High-order embedded WENO schemes," in *Spectral and High Order Methods for Partial Differential Equations ICOSAHOM 2016*, Springer, 2016.

[104] C. Klingenberg and B. Plohr, *Multidimensional hyperbolic problems and computations*, vol. 29, ch. An introduction to front tracking. Springer, 1991.

[105] S. O. Unverdi and G. Tryggvason, "A front-tracking method for viscous, incompressible, multi-fluid flows," *J. Comput. Phys.*, vol. 100, pp. 25–37, May 1992.

[106] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y.-J. Jan, "A front-tracking method for the computations of multiphase flow," *J. Comput. Phys.*, vol. 169, pp. 708–759, May 2001.

[107] P. K. Devi and V. Naidu, "A new finite difference front tracking method for two phase 1-d moving boundary problems," *Procedia Engineering*, vol. 127, pp. 1034 – 1040, 2015.

[108] M. Irfan and M. Muradoglu, "A front tracking method for direct numerical simulation of evaporation process in a multiphase system," *Journal of Computational Physics*, vol. 337, pp. 132 – 153, 2017.

[109] W. Bangerth and R. Rannacher, *Adaptive Finite Element Methods for Differential Equations (Lectures in Mathematics. ETH Zürich)*. Birkhäuser, 2003.

[110] H. Weller, P. Browne, C. Budd, and M. Cullen, "Mesh adaptation on the sphere using optimal transport and the numerical solution of a Monge–Ampère type equation," *Journal of Computational Physics*, vol. 308, pp. 102 – 123, 2016.

[111] B. N. Granzow, M. S. Shephard, and A. A. Oberai, "Output-based error estimation and mesh adaptation for variational multiscale methods," *Computer Methods in Applied Mechanics and Engineering*, vol. 322, pp. 441 – 459, 2017.

[112] R. Akoh, S. Ii, and F. Xiao, "A multi-moment finite volume formulation for shallow water equations on unstructured mesh," *Journal of Computational Physics*, vol. 229, no. 12, pp. 4567 – 4590, 2010.

[113] H. Nishikawa and P. L. Roe, "Third-order active-flux scheme for advection diffusion: Hyperbolic diffusion, boundary condition, and Newton solver," *Computers & Fluids*, vol. 125, pp. 71 – 81, 2016.

[114] T. A. Eymann, *Active Flux Schemes*. PhD thesis, University of Michigan, 2013.

[115] J. Ruppert, "A Delaunay refinement algorithm for quality 2-dimensional mesh generation," *J. Algorithms*, vol. 18, pp. 548–585, May 1995.

[116] D. R. Durran, *Numerical methods for fluid dynamics: with applications to geophysics*. Springer, 2010.

[117] D. S. Richeson, *Euler's gem: the polyhedron formula and the birth of topology*. Princeton University Press, 2008.

[118] J. R. Shewchuk, "Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator," in *Applied Computational Geometry: Towards Geometric Engineering* (M. C. Lin and D. Manocha, eds.), vol. 1148 of *Lecture Notes in Computer Science*, pp. 203–222, Springer-Verlag, May

1996. From the First ACM Workshop on Applied Computational Geometry.

[119] K. Ding, K. J. Fidkowski, and P. L. Roe, "Continuous adjoint based error estimation and r-refinement for the active-flux method," *AIAA SciTech*, 2016.

[120] C. A. A. Minoli and D. A. Kopriva, "Discontinuous Galerkin spectral element approximations on moving meshes," *Journal of Computational Physics*, vol. 230, no. 5, pp. 1876 – 1902, 2011.

[121] D. A. Kopriva, *Implementing Spectral Methods for Partial Differential Equations*. Springer, 2009.

[122] J. S. Hesthaven and T. Warburton, *Nodal Discontinuous Galerkin methods*. Springer, 2008.

[123] C. Canuto, M. Y. Hussaini, A. Quarter, and T. A. Zang, *Spectral methods*. Springer-Verlag, 2007.

[124] R. Pasquetti and F. Rapetti, "Spectral element methods on triangles and quadrilaterals: comparisons and applications," *J. Comput. Phys.*, vol. 198, pp. 349 – 362, 2004.

[125] D. Wirasaet, S. Tanaka, E. J. Kubatko, J. J. Westerink, and C. Dawson, "A performance comparison of nodal discontinuous Galerkin methods on triangles and quadrilaterals," *International Journal for Numerical Methods in Fluids*, vol. 64, pp. 1336 – 1362, 2010.

[126] W. J. Gordon and C. A. Hall, "Construction of curvilinear co-ordinate systems and applications to mesh generation," *International Journal for Numerical Methods in Engineering*, vol. 7, pp. 461 – 477, 1973.

[127] A. M. Turing, "On computable numbers, with an application to the Entscheidungsproblem," *Proceedings of the London Mathematical Society*, vol. 42, no. 2, pp. 230 – 265, 1937.

[128] E. J. Barbeau, *Polynomials*. Springer-Verlag, 1989.

[129] G. Szegő, *Orthogonal polynomials*. Providence, 1939.

[130] L. Demkowicz, *Computing with hp-adaptive finite elements: volume 1, one and two dimensional elliptic and Maxwell problems*. Chapman and Hall/CRC, 2006.

[131] L. Demkowicz, *Computing with hp-adaptive finite elements: volume 2, Frontiers three dimensional elliptic and Maxwell problems with applications*. Chapman and Hall/CRC, 2007.

[132] P. Houston, B. Senior, and E. Süli, "hp-discontinuous Galerkin finite element methods for hyperbolic problems: error analysis and adaptivity," *International Journal for Numerical Methods in Fluids*, vol. 40, no. 1-2, pp. 153–169, 2002.

[133] R. Hartmann and P. Houston, "Adaptive discontinuous Galerkin finite element methods for nonlinear hyperbolic conservation laws," *SIAM Journal on Scientific Computing*, vol. 24, no. 3, pp. 979–1004, 2003.

[134] A. Anderson, X. Zheng, and V. Cristini, "Adaptive unstructured volume remeshing – I: The method," *Journal of Computational Physics*, vol. 208, no. 2, pp. 616 – 625, 2005.

[135] E. J. Kubatko, J. J. Westerink, and C. Dawson, "hp discontinuous Galerkin methods for advection dominated problems in shallow water flow," *Computer Methods in Applied Mechanics and Engineering*, vol. 196, no. 1, pp. 437 – 451, 2006.

[136] H. Zhu and J. Qiu, "Adaptive Runge–Kutta discontinuous Galerkin methods using different indicators: One-dimensional case," *Journal of Computational Physics*, vol. 228, no. 18, pp. 6957 – 6976, 2009.

[137] M. Ohlberger, "A review of a posteriori error control and adaptivity for approximations of non-linear conservation laws," *International Journal for Numerical Methods in Fluids*, vol. 59, no. 3, pp. 333–354, 2009.

[138] B. Cockburn and C.-W. Shu, "Runge–Kutta discontinuous Galerkin methods for convection-dominated problems," *Journal of Scientific Computing*, vol. 16, no. 3, pp. 173 – 261, 2001.

[139] A. R. Winters and D. A. Kopriva, "ALE-DGSEM approximation of wave reflection and transmission from a moving medium," *Journal of Computational Physics*, vol. 263, pp. 233 – 267, 2014.

[140] B. Cockburn and C.-W. Shu, "The Runge–Kutta local projection P^1-discontinuous Galerkin method for scalar conservation laws," *RAIRO modél Math. Anal. Numér.*, vol. 25, pp. 337 – 361, 1991.

[141] L. Krivodonova and R. Qin, "An analysis of the spectrum of the discontinuous Galerkin method," *Applied Numerical Mathematics*, vol. 64, pp. 1 – 18, 2013.

[142] D. Gottlieb and C.-W. Shu, "On the Gibbs phenomenon and its resolution," *SIAM Review*, vol. 39, no. 4, pp. 644–668, 1997.

[143] R. H. Goddard, "A method of reaching extreme altitudes," *Smithsonian Miscellaneous Collections*, vol. 72, no. 2, 1919.

[144] L. S. Pontryagin, V. L. Boltyanskii, R. V. Gamkrelidze, and K. N. Trirogoff, *The mathematical theory of optimal processes.* Interscience, 1962.

[145] R. Bellman, *Dynamic Programming.* Princeton University Press, 1957.

[146] D. E. Kirk, *Optimal Control Theory.* Prentice Hall, 1970.

[147] F. Tröltzsch and J. Sprekels, *Optimal control of partial differential equations: theory, methods and applications.* American Mathematical Society, 2010.

[148] J. L. Lagrange, *Mécanique Analytique.* Courcier, 1811.

[149] W. Karush, *Minima of functions of several variables with inequalities as side constraints.* PhD thesis, University of Chicago, 1939.

[150] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," *Proceedings of the Second Berkely Symposium on Mathematical Statistics and Probability*, pp. 481 – 492, 1951.

[151] D. G. Luenberger and Y. Ye, *Linear and nonlinear programming.* Springer International Publishing, 2016.

[152] E. Kreyszig, *Functional analysis with applications.* John Wiley & Sons, 1978.

[153] P. K. van V. L. Mehrmann, *Differential-algebraic equations: analysis and numerical solution.* European Mathematical Society, 2006.

[154] A. Borzì and V. Schulz, *Computational optimization of systems goverend by partial differential equations.* SIAM, 2012.

[155] Y. Saad, *Iterative methods for sparse linear systems.* SIAM, 2003.

[156] W. Feller, *An introduction to probability theory and its applications*. Wiley, 1971.

[157] J. P. Romano and A. F. Siegel, *Counterexamples in Probability and Statistics*. Chapman and Hall, 1986.

[158] J. R. Leigh, *Control Theory: A Guided Tour*. The Institution of Engineering and Technology, 2012.

[159] J. R. Rumble, ed., *Handbook of Chemistry and Physics*. CRC Press, 98th ed., 2017.

[160] A. C. Claus, "On Archimedes' burning glass," *Appl. Opt.*, vol. 12, no. 10, pp. A14–A14, 1973.

[161] MIT, "Archimedes death ray: October, 2005."

[162] H. W. Fowler and F. G. Fowler, *The works of Lucian of Samosata*. Oxford at the Clarendon press, 1905.

[163] X.-J. Wang, "On the design of a reflector antenna," *Inverse Problems*, vol. 12, pp. 351 – 375, 1996.

[164] C. Villani, *Optimal transport: old and new*. Springer, 2009.

[165] S. A. Kochengin and V. I. Oliker, "Determination of reflector surfaces from near-field scattering data," *Inverse Problems*, vol. 13, no. 2, p. 363, 1997.

[166] S. A. Kochengin and V. I. Oliker, "Determination of reflector surfaces from near-field scattering data ii. numerical solution," *Numerische Mathematik*, vol. 79, no. 4, pp. 553–568, 1998.

[167] S. A. Kochengin, V. I. Oliker, and O. von Tempski, "On the design of reflectors with prespecified distribution of virtual sources and intensities," *Inverse Problems*, vol. 14, no. 3, pp. 661 – 678, 1998.

[168] H. Ries and J. Muschaweck, "Tailored freeform optical surfaces," *J. Opt. Soc. Am. A*, vol. 19, no. 3, pp. 590–595, 2002.

[169] C. R. Prins, J. H. M. ten Thije Boonkkamp, J. van Roosmalen, W. L. IJzerman, and T. W. Tukker, "A Monge–Ampère-solver for free-form reflector design," *SIAM Journal on Scientific Computing*, vol. 36, no. 3, pp. B640–B660, 2014.

[170] N. K. Yadav, J. H. M. ten Thije Boonkkamp, and W. L. IJzerman, "A least-squares method for the design of two-reflector optical systems," CASA-report 1619, Eindhoven University of Technology, 2016.

[171] C. R. Prins, R. Beltman, J. H. M. ten Thije Boonkkamp, W. L. IJzerman, and T. W. Tukker, "A least-squares method for optimal transport using the Monge–Ampère equation," *SIAM Journal on Scientific Computing*, vol. 37, no. 6, pp. B937–B961, 2015.

[172] Z. Feng, B. D. Froese, and R. Liang, "Freeform illumination optics construction following an optimal transport map," *Appl. Opt.*, vol. 55, no. 16, pp. 4301–4306, 2016.

[173] A. Hirst and J. A. Muschaweck, "Irradiance tailoring for extended sources in 3d by implicit integral equation solution," in *Imaging and Applied Optics 2015*, p. FT4B.2, Optical Society of America, 2015.

[174] J.-D. Benamou and Y. Brenier, "A computational fluid mechanics solution to the Monge-Kantorovich mass transfer problem," *Numerische Mathematik*, vol. 84, no. 3, pp. 375–393, 2000.

[175] A. Ishimaru, *Wave Propagation and Scattering in Random Media. Vol 1: Single Scattering and Transport Theory.* Academic Pr, 1978.

[176] B. Øksendal, *Stochastic Differential Equations: An Introduction with Applications (Universitext).* Springer, 2014.

[177] M. A. Taylor, B. A. Wingate, and R. E. Vincent, "An algorithm for computing Fekete points in the triangle," *SIAM J. Numer. Anal.*, vol. 38, no. 5, pp. 1707 – 1720, 2000.

[178] J. S. Hesthaven, "From electrostatics to almost optimal nodal sets for polynomial interpolation in a simplex," *SIAM J. Numer. Anal.*, vol. 35, no. 2, pp. 655 – 676, 1998.

[179] M. Briani, A. Sommariva, and M. Vianello, "Computing Fekete and Lebesgue points: simplex, square, disk," *Journal of Computational and Applied Mathematics*, vol. 236, pp. 2477 – 2486, 2012.

[180] J. Rubinstein and G. Wolansky, "A variational principle in optics," *J. Opt. Soc. Am. A*, vol. 21, no. 11, pp. 2164 – 2172, 2004.

[181] R. A. Heinlein, *The Moon is a harsh mistress.* G. P. Puntam's Sons, 1966.

[182] J. Binney and S. Tremaine, *Galactic Dynamics*. Princeton University Press, 1987.

[183] R. P. D. Sisto, X. S. Ramos, and C. Beaugé, "Giga-year evolution of Jupiter Trojans and the asymmetry problem," *Icarus*, vol. 243, pp. 287 – 295, 2014.

[184] Y. Saad and M. H. Schultz, "Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM Journal on Scientific and Statistical Computing*, vol. 7, no. 3, pp. 856–869, 1986.

Index