# The Monge-Ampère equation for optimal mass transport with applications in optical design

van Roosmalen, J.

*Award date:*
2013

Link to publication

# The Monge-Ampère equation for optimal mass transport with applications in optical design

Jarno van Roosmalen

Supervisor: Dr. ir. Jan ten Thije Boonkkamp

11th January 2013

## Abstract

For lighting applications it is important to control the direction of light beams using optical elements. Recent developments make it possible to design asymmetrical optical elements. To be able to calculate the shape of an optic that produces the desired light pattern, new design methods have to be developed. In this thesis the relation between optics and optimal mass transport will be explored. Using calculus of variations, a system of non linear PDEs is derived. We pose that an optical design is given by the solution of a Monge-Ampère type equation with a special boundary condition. Using a recently developed numerical algorithm by Oberman et al.[BFO12c], a computer program was written for solving this class of Monge-Ampère equations. Different examples show that the convergence behaviour is between first and second order depending on the problem set. The algorithm scales very well as the runtime grows with the 1.3rd power of the number of grid points. To show the practical usability, a free form lens is computed for a parallel beam source, with very promising results.

# Foreword

You are now reading my master thesis. I would not have been able to do this project and write this report without the help and support of many people. First I would like to thank my supervisor Jan ten Thije Boonkkamp for giving me this opportunity and introducing me to Philips Lighting. He has been a great help during this project, steering me in the right direction when needed. His feedback was often very detailed and helped me a lot for writing this thesis. At Philips Lighting I would like to thank Wilbert IJzerman and Teus Tukker for providing me with this great project. Wilbert's questions and feedback during the weekly meetings were very useful. Teus helped a lot in making me feel welcome, and our discussions were often enlightening. He can always tell an anecdote from his year at Philips Research. I'm grateful to Corien Prins for letting me work on this topic that is so interleaved with her PhD research. She gave some crucial input for my work and can hopefully use it in her further research.

Everyone at Philips was very nice and I really enjoyed the many conversations in the office and during the lunches. I would like to thank especially Siebe de Zwart and Alyona Ivanova for all the awesome discussions and brainstorm moments. Ferry Zijp let me join the NIO project meetings for which I am very grateful. It let me see and learn a lot about the everyday routine and inner workings at Philips.

Last but not least I would like to thank my family. They have always been very supporting throughout my study time in more ways than they know.

# Contents

# Chapter 1

# Introduction

Lighting plays a central role in modern life. We use lights everywhere, from our homes and offices to the streets. Its use can be utilitarian like in an office, or have a wider purpose by creating a certain mood like candles. Some lights are used to be able to see things, others like bike rear lights, are used to be seen by other people.

For many applications there is the wish to be able to direct the light from a lamp or light source. For example, car lights should illuminate the road and traffic signs without blinding any oncoming traffic. Recently, the steering of light has attracted much attention as it might have multiple benefits. First, it can help in producing exactly the lighting needed for the application. For example a bike light might send most of the light to the street surface, but some in other directions for your visibility. Second, directing light where it is needed reduces light pollution. Street lights send some light up into the sky. This obscures the stars for many people living in urban regions, and can even affect the health of people and animals, see [Wik12b], [Wik12a]. Third, it reduces cost and energy usage. For example a floodlight used to illuminate a building sends on average half its light over the roof. The light that arrives at the building hits the windows and is waisted. So, only a fraction of the light is used as intended. If you are able to send lamplight were it is useful you can do with a much smaller lamp and save money and electricity at the same time.

In a lighting shop you find that most lamps and/or luminaires use optical elements to direct the light. Often this element is a reflector. On closer inspection one would see that the vast majority of these elements are either rotationally symmetric (e.g. spotlights) or translationally symmetric (e.g. fluorescent tubes in offices). There are several reasons for using symmetrical

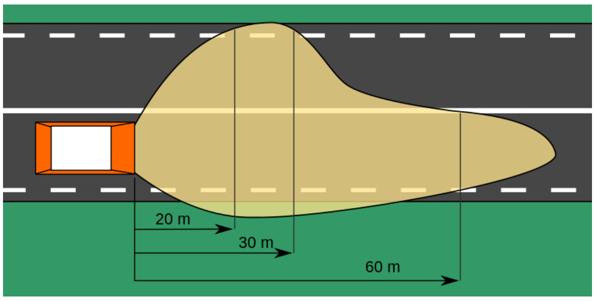


Figure 1.1: Light pattern as desired for car headlights. Source: Wikipedia

elements: a) it is much easier to design a symmetrical optic, b) until recently it was the only type of optic that could be easily produced in mass production. However, many applications demand asymmetrical optical components. For example in street lighting the area illuminated by a single pole is a big rectangle, which is not centred around the lamp, and consequently these are not rotational symmetric. Another example is the car head lights, see Figure 1.1, where you must illuminate the road without blinding oncoming traffic. Recent insights and advancements in technology make it feasible to design free form optics. There are several developments that come together to drive this.

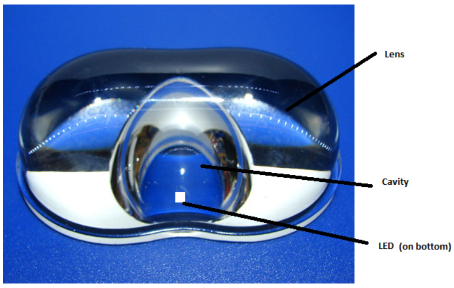


Figure 1.2: A ‘peanut‘ nonrotationally symmetric lens for a street light. The asymmetric cavity and lens steer the hemisphere of light into a rectangular pattern, source http://led22.ru/ledstat/power/powereng.html, accessed 04-12-2012

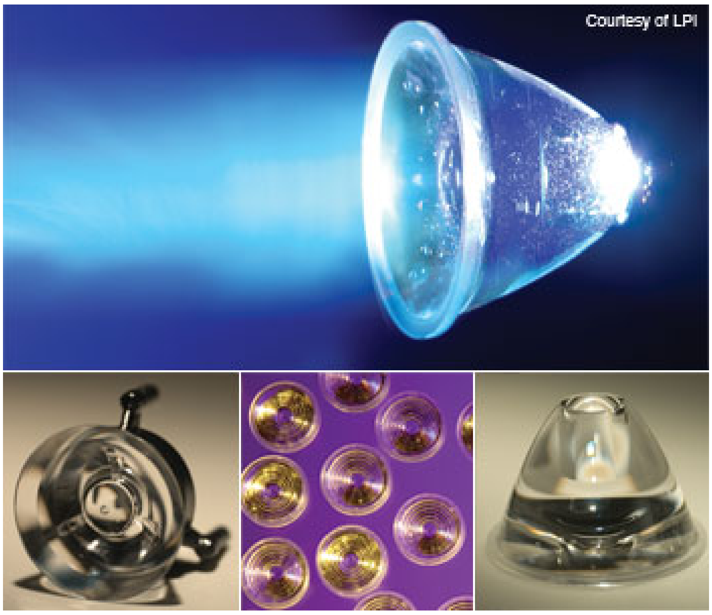


Figure 1.3: A set of rotationally symmetric collimators, source [BM07]

The introduction of LED lighting gives rise to a revolutionary change in lighting. First it increases the efficacy, a measure of the amount of visible light produced per unit of electrical power input, for many lighting applications. Secondly its thermal properties are different, allowing the use of different materials like plastics instead of glass. This allows more freedom in manufacturing of optical components. Thirdly, the total amount of light produced by LEDs is not yet at the same level as that of certain old technologies like sodium street lights. The implication is that a more efficient use of the light is needed to have the same results. For an example of an asymmetric lens, see Figure 1.2.

The global movement to a more sustainable society means there is a bigger focus by customers on efficiency. Lamps should provide the same illumination as they are used to, but not waste any by sending into the sky and they should consume much less electricity. There is a significant saving to be made as around 19% of all electricity world wide was for lighting in 2005 [Int06]. New production methods and materials allow more freedom in the design of optics. Examples are injection molding for glass and plastics, or advancements in technology of milling machinery, see for example Figure 1.3.

This all requires the design tools to catch up with these developments. Therefore new design methods have to be invented to help optical designers utilise these new opportunities. One would like to have tools that can be given a light source specification and a target intensity. These tools should then output the optical element that achieves this. That means we search for a so-called *inverse method*. This name comes as these problems are the inverse of the direct problem, which is to calculate the light output distribution given a light source and an optical element.

## 1.1 Current state

Currently there are only a very limited number of methods for solving these optical problems [BM07]. The first method is based on multi-parameter optimisation. The designer should create a quantitative merit function, and an analytically parameterised description of the free form surface. Then this is optimised with a generic optimisation algorithm. However this has many drawbacks, as the freedom in surface shape is restricted by the initial parameterised description, and without a good merit function it will not work.

A second method is the simultaneous multiple surface (SMS) method [BM07] [Wik12c]. It is a constructive method that slowly builds up an optical surface. The method is very complex, see [Wik12c] for an overview. It was one of the first methods that can handle extended sources (a line or surface emitting light). One disadvantage is that it only maps the edge rays of a source to a target without steering the distribution of light in between. A second disadvantage is that it is a very low order and unstable method, which is very hard to implement without a lot of user intervention. It is a method based around the full ‘phase space‘ which means that for fully free form optics, the mathematics takes place in a four-dimensional space.

A third category of methods is based on partial differential equations (PDEs) and optimal transport. According to Benitez [BM07] the use of PDEs in optical design is rare. However, it might offer many benefits. Before designing an optical element directly, we can take a step back for a moment. If we would know a suitable mapping that specifies for each ray coming from the source where on the target it should go, making an optic is relatively easy. For finding a mapping we are given the intensity distribution of the light source, and a target intensity distribution. The main physics principle involved is a conservation law, that states that the amount of light is conserved. This is now very similar to a big class of known problems called optimal mass transport. It turns out that for the mathematics it does not matter whether we are transporting sand piles, or light rays. As optimal transport is a very big field with a lot of applications, there exists a lot of research on analysis of these problems. In this thesis the link between optics and optimal transport is explored, and a numerical method is developed.

## 1.2 Outline of thesis

This thesis consists of the following chapters:

- In Chapter 2 a short overview of relevant concepts from geometrical optics and lighting is given.
- Chapter 3 explores the rotationally symmetric situation, as symmetry allows for an easier analysis.
- This analysis is expanded for asymmetric optics in Chapter 4. We derive a general set of equations, after which for a different problem a specific equation is shown.
- A numerical solver for the Monge-Ampère equation is described in Chapter 5, including a discussion of the boundary condition.
- The numerical solver is then used to solve a collection of examples in Chapter 6
- In Chapter 7 everything comes together for calculating the surfaces of a lens based on this theory.
- Finally we give in Chapter 8 the conclusions and discussion.

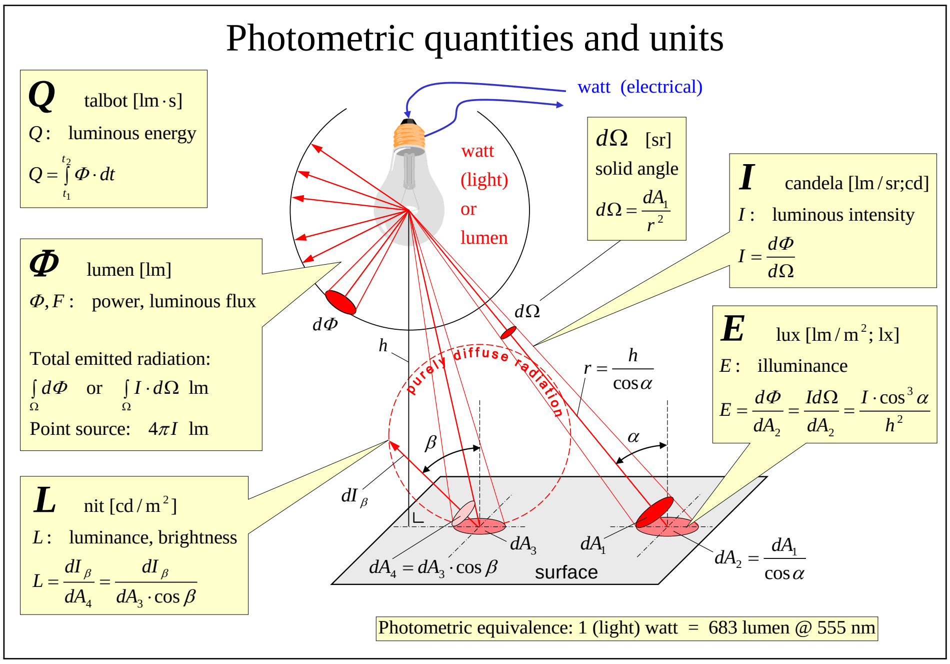
# Chapter 2

# Geometric Optics and Photometry

The aim of this chapter is to give a short introduction to the terminology and concepts of lighting and geometrical optics. For a more comprehensive treatment of these concepts the reader is referred to textbooks as [Hec02] and [PPP07].

## 2.1 Photometry

In illumination design a lot of physical quantities and units are used, cf. [Mae97, sec 2.2], see Figure 2.1. To understand them, imagine a single light bulb viewed from a large distance, so it resembles a point source. Then the following quantities are defined:



Photometric quantities and units
$Q$ talbot [lm·s]
$Q$: luminous energy
$Q = \int_{t_1}^{t_2} \Phi \cdot dt$
$\Phi$ lumen [lm]
$\Phi, F$: power, luminous flux
Total emitted radiation:
$\int_\Omega d\Phi$ or $\int_\Omega I \cdot d\Omega$ lm
Point source: $4\pi I$ lm
$L$ nit [cd/m$^2$]
$L$: luminance, brightness
$L = \frac{dI_\beta}{dA_4} = \frac{dI_\beta}{dA_3 \cdot \cos\beta}$
watt (electrical)
watt (light) or lumen
$d\Omega$ [sr]
solid angle
$d\Omega = \frac{dA_1}{r^2}$
$I$ candela [lm/sr;cd]
$I$: luminous intensity
$I = \frac{d\Phi}{d\Omega}$
$E$ lux [lm/m$^2$; lx]
$E$: illuminance
$E = \frac{d\Phi}{dA_2} = \frac{I d\Omega}{dA_2} = \frac{I \cdot \cos^3\alpha}{h^2}$
$d\Phi$
$d\Omega$
$h$
$r = \frac{h}{\cos\alpha}$
purely diffuse radiation
$\beta$
$\alpha$
$dI_\beta$
$dA_3$
$dA_1$
$dA_4 = dA_3 \cdot \cos\beta$
surface
$dA_2 = \frac{dA_1}{\cos\alpha}$
Photometric equivalence: 1 (light) watt = 683 lumen @ 555 nm

Figure 2.1: Various photometric quantities and their SI units. Image courtesy of Peter Nuyens.

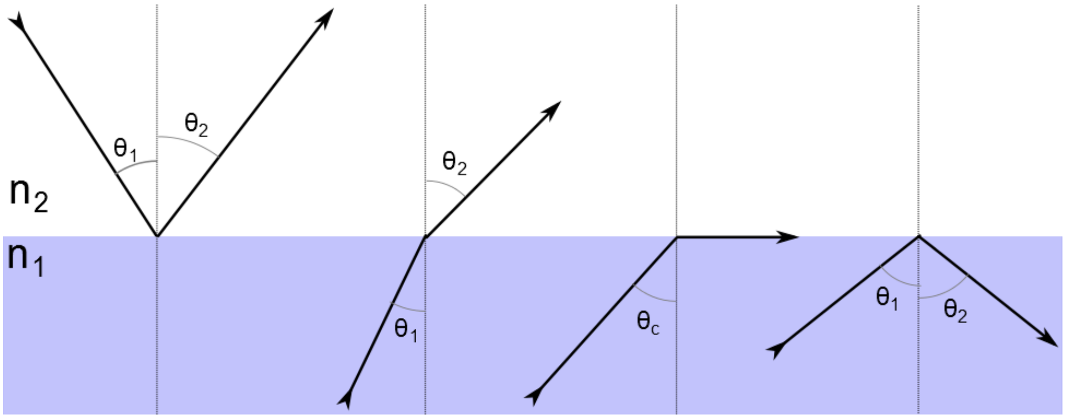


Figure 2.2: Diagram explaining the different optical effects. From left to right: Reflection, refraction, refraction at critical angle, total internal reflection

*Luminous flux*, also called luminous power, is the quantity describing the perceived amount of light. For example, the total amount emitted by a light bulb or falling on a surface. The amount of light is the radiated energy per unit time, adjusted for the sensitivity of the human eye to different colours. The unit of luminous flux is the *lumen* [lm].

*Luminous intensity* is the luminous flux per solid angle. The corresponding unit is lumen per steradian [lm/sr], which is called the *candela* [cd].

*Illuminance* is the luminous flux received by a unit area on a surface. The corresponding units is the *lux*, which is equal to lumen per square meter [lm/m$^2$]. If the surface emits the light it is called *emittance*

In the case a light source has a specified intensity profile $I$, given as function in spherical coordinates, then the total flux is easily calculated by integrating over the unit sphere. The light leaving our source can be manipulated using optics, the topic of the next section.

## 2.2 Geometrical optics

In this report we restrict ourselves to the domain of geometrical optics as this is enough for our purpose. This is an approximation where the wave character of light is ignored. This approximation is valid as long as the typical dimensions in the system are much larger then the wavelength of the light, which is the case for most illumination optics. In this approximation light is described as travelling in straight lines called *rays*. When light rays hit an optical surface two important phenomena can happen, reflection and refraction. The effect on the light ray depends on the geometry and material of the optical surface. The geometry of the surface is often described using the surface normals. In geometrical optics the angles of rays at an interface are always measured with respect to the surface normal.

### 2.2.1 Reflection

The simplest effect is when light rays are reflected by an optical surface, see Figure 2.2. This happens for example on a mirror or on a smooth water surface. The law of reflection states that the angle $\theta_1$ of the incident ray is equal to the angle $\theta_2$ of the reflected ray, i.e.,

$$\theta_1 = \theta_2, \qquad \text{for } 0 \leq \theta_1, \theta_2 \leq \tfrac{\pi}{2}. \tag{2.1}$$

In three dimensions the incident ray, the surface normal and the reflected ray all lie in a plane.

### 2.2.2 Refraction

If light rays hit the surface of another transparent medium they can be refracted (bend). An example is light entering water from the air. We have the following situation, a light ray hits the surface at an angle of incidence $\theta_1$ and after passing the interface it continues under angle $\theta_2$ with respect to the surface normal, see Figure 2.2. The refraction angle depends on a material property of the media involved, namely the index of refraction. Assume the first medium has index $n_1$ and the second $n_2$ then the refraction is governed by Snell's law

$$n_1 \sin\theta_1 = n_2 \sin\theta_2. \tag{2.2}$$

If $n_1 < n_2$ the light travels into a medium with a higher reflective index and deflects toward the normal, an example is light entering water from air. The reverse is true when $n_1 > n_2$ light enters a medium with a smaller index of refraction and the light is bent away from the normal. An example is light leaving glass and entering air.

In this last case something interesting can happen. As the rays are deflected away from the normal we have $\theta_2 > \theta_1$. This means that there exists an angle $\theta_c$ for the incident ray called the critical angle where $\theta_2 = 90°$. The refracted ray now travels parallel to the surface. One might wonder what happens if $\theta_1 > \theta_c$. Then the ray is completely reflected back into the medium it came from, according to the law of reflection. This effect is called total internal reflection (TIR) is illustrated in Figure 2.2.

# Chapter 3

# Rotationally Symmetric Optics

In this chapter a description of a rotationally symmetric optical system be given. The use of a minimisation formulation is shown for some simple examples.

The problem is the setup as seen in Figure 3.1. We have a point source at the origin sending light along directions in the source set $X \subset [0, \frac{\pi}{2}]$, with intensity distribution (density) $f : X \to \mathbb{R}^+$. An angle at the source is usually denoted with $t \in X$, and is defined as the angle between the positive $x$-axis and the ray, counter clockwise. The $x$-axis is the rotation axis for the rotational symmetry case. Using some optical component like a lens or reflector the light is directed in a target direction. The target is the so-called *far field*, i.e., you look from a large distance so that the whole system can be regarded as a point source. Therefore, the target is described only by an angular coordinate representing the direction the light is send to. This target interval is called $Y \subset [0, \frac{\pi}{2}]$, and the prescribed target intensity is denoted by $g : Y \to \mathbb{R}^+$. Target rays are denoted using an angle $\theta$ defined as the angle between the

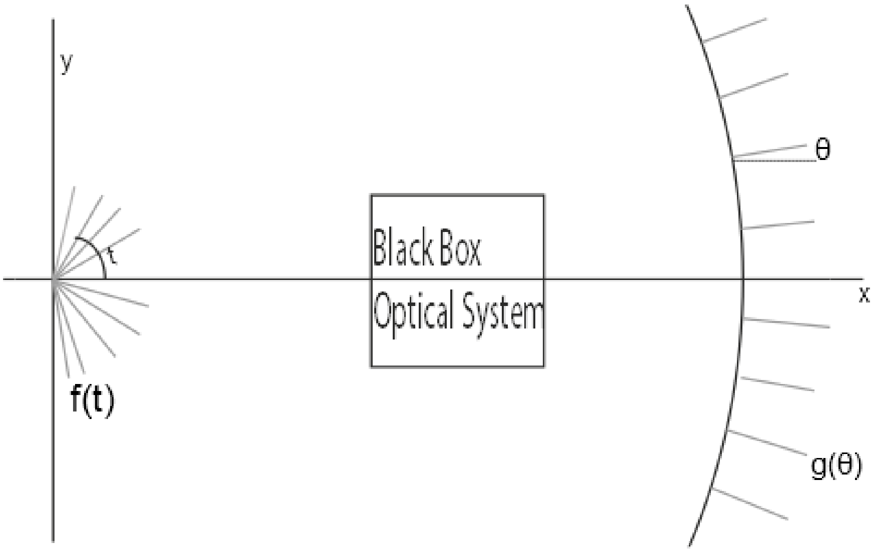


Figure 3.1: The setup of the optical system. On the left is a point source with intensity distribution $f(t)$. On the right, in the far field we have the target distribution $g(\theta)$. In the middle is an unknown optical system.

positive $x$-axis and the ray.

What we would like to know is the mapping function $s : X \to Y$ which maps $t \mapsto \theta(t)$. This function tells us exactly which light ray from the source is directed to which direction at the target. So the mapping assigns to each source angle $t$ a target angle $\theta$, so for brevity $\theta(t)$. One of the most fundamental laws of physics is the law of conservation of energy. Which states that in this case the amount of luminous flux in the source should equal the amount in the target. Therefore our problem must be such that the conservation law

$$\int_{\hat{X}} f(t)\,\mathrm{d}t = \int_{s(\hat{X})} g(\theta)\,\mathrm{d}\theta, \tag{3.1}$$

is obeyed for all closed subsets $\hat{X} \subset X$. To compute a mapping it is often easier to formulate (3.1) as differential equation, i.e.,

$$f(t) = g(\theta(t))|\dot{\theta}(t)|. \tag{3.2}$$

Here $\dot{\theta}(t)$ denotes the $t$-derivative of $\theta(t)$. For solutions to this equation to exist, a few conditions are imposed on the source and target densities:

1. $f(t) > 0$ for all $t \in X$ with $f(t) \geq 0$ for $t$ at the boundary of $X$, and $f$ must be continuous,
2. $g(\theta) > 0$ for all $\theta \in Y$ as we have only positive densities, and $g$ is continuous,
3. $\int_X f(t)\,\mathrm{d}t = \int_Y g(\theta)\,\mathrm{d}\theta < \infty$, both the source and target densities are required to have a finite energy.

In some of the papers, e.g. [Eva01], $f$ and $g$ are usually treated as measures which have these properties automatically. The most interesting part of (3.2) is the absolute value of $\dot{\theta}(t)$. This allows for two different solutions. If $\operatorname{sgn}(\dot{\theta}(t)) > 0$ then we have a diverging solution where the light rays will never cross. The condition $\operatorname{sgn}(\dot{\theta}(t)) < 0$ leads to a converging solution where all the rays will cross. Note that a corollary of (3.2) is that $\dot{\theta}(t)$ can never be equal to 0 as both $f$ and $g$ are strictly positive in the interior.

To solve the differential equation a boundary condition must be specified. We want the mapping to be surjective, so the most natural condition is $s(X) = Y$. A sufficient condition is $s(\partial X) = \partial Y$ [Fro12]. Obviously, the boundary of an interval consists of only two points. Let $X = [t_0, t_1], Y = [\theta_0, \theta_1]$, then there are two possible boundary conditions, either

$$\theta(t_0) = \theta_0 \text{ and } \theta(t_1) = \theta_1, \tag{3.3a}$$

or

$$\theta(t_0) = \theta_1 \text{ and } \theta(t_1) = \theta_0. \tag{3.3b}$$

These options only lead to consistent problems if they are in agreement with the sign chosen for $\dot{\theta}(t)$. So, to solve a problem, we take (3.2) together with a sign for $\dot{\theta}(t)$ and one of the boundary conditions, either (3.3a) or (3.3b). It does not matter whether you choose the first or the second BC, as $\theta_0$ and $\theta_1$ are not independent but related through (3.1). Global energy conservation from (3.1) makes sure that at the other side the boundary condition is also met. Note that $t_0$ and $t_1$ are the bounds of the set $X$, which is the support of the function $f$. The same is true for $\theta_0$ and $\theta_1$ as bounds of $Y$ which supports $g$. This means that these parameters can not be chosen freely but are part of the problem formulation, and relate directly to condition 3 above.

## 3.1 Segmentation

For design reasons it is often needed to create more freedom, e.g., to be able to optimise with respect to some merit function. This freedom can be created by dividing $X$ into different segments. Each of these segments corresponds to a subset of the source set $X$, and has its own target. Each segment is allowed to have a different sign for $\dot{\theta}(t)$. Then one can create a parameterised family of solutions by letting the location of the interfaces be parameters, or alternatively the division of the target density over the different segments. Solving the optical problem then generates a solution containing the parameters. One can then minimise some cost function over the solution with this parameter. To make this reasoning more precise, an exact formulation of this segmentation is required to know the conditions. This section is formulated as general as possible, without referring to the rotational symmetry,

Consider the domain $X$ and co-domain $Y$ of the mapping $s$. Assume $X$ and $Y$ are compact, convex sets. This domain can be partitioned into $n$ segments $X_1, X_2, \ldots, X_n$. This is done such that the following properties/conditions holds

1. Each $X_i$ is a closed connected set with a nonzero measure.
2. $\bigcup_{i=1}^{n} X_i = X$.
3. The measure of $X_i \cap X_j$ is zero if $i \neq j$.
4. We write $Y_i = s(X_i)$ the image of $X_i$ under the mapping $s$.
5. For the images we need $\bigcup_{i=1}^{n} Y_i = Y$. Note this is a covering not a partitioning, i.e., not necessarily disjoint.
6. We define on each $Y_i$ a $g^i \geq 0$ as the target distribution for source segment $X_i$.
7. We need $\sum_{i=1}^{n} g^i(\mathbf{y}) = g(\mathbf{y})$ for all $\mathbf{y} \in Y$. Energy conservation per segment requires $\int_{X_i} f(\mathbf{x})\,\mathrm{d}\mathbf{x} = \int_{Y_i} g^i(\mathbf{y})\,\mathrm{d}\mathbf{y}$.
8. The $g^i$ are continuous.

These requirements make sure we get independent problems on each segment. Furthermore, some nasty problems, e.g. zero measure sets, are excluded by requiring some regularity. The choice of the sets $Y_i$ and the accompanying functions $g^i$ introduces a new degree of freedom in the problem, as it is not fixed, as long as the requirements above are satisfied. It is up to an optical designer to determine the most appropriate way of segmenting her problem. The designer can use this freedom to help achieve other design goals/criteria. From here on, we return to the rotational symmetric situation. For each of the segments we now have exactly one of three cases.

1. $f = 0$ on $X_i$ and $g = 0$ on $Y_i$. That means there is no light in the segment so the mapping is meaningless (undefined). By allowing these type of segments we can relax the conditions for a well-posed problem. If $f(\mathbf{x}) = 0$ is in some convex closed subset of $X$ we can make this into a separate segment, and still solve the problem.
2. $f > 0$ and $g > 0$ on the interior of $X_i$. On the boundary $\partial X_i$ we can allow $f = 0$.
We choose $\dot{\theta}(t) > 0$ on $X_i$.
3. $f > 0$ and $g > 0$ on the interior of $X_i$. On the boundary $\partial X_i$ we can allow $f = 0$.
We choose $\dot{\theta}(t) < 0$ on $X_i$.

## 3.2 Formulation as minimisation problem

One way to use the freedom from segments is to optimise some cost function. The associated cost function determines in which way an *optimal* map is sought. A simple example is the following minimisation using a quadratic cost function comparing the target direction of a ray to the source direction.

$$\min_{\theta}\left\{\int_X (\theta(t)-t)^2 f(t)\,\mathrm{d}t \;\middle|\; |\dot\theta(t)| = \frac{f(t)}{g(\theta(t))} \text{ and } \theta(t_r)=\theta_r\right\}, \tag{3.4}$$

where $t_r$ and $\theta_r$ are reference positions from the boundary condition. Although we are only minimising over a set containing two solutions (one with $\dot\theta(t)$ positive, and one with $\dot\theta(t)$ negative) it is useful to do some analysis in the one-dimensional case. This simplifies the formulae so some inside can be gained.

The minimisation problem as posed in (3.4) is further analysed, cf. Section 2.2 from [?evans2001]. The constraint given by the differential equation can be incorporated into the minimisation functional using a *Lagrange multiplier* [CC03]. This means we look for an extremal for the following functional

$$J[\theta;\lambda] = \int_X F(t,\theta,\dot\theta)\,\mathrm{d}t = \int_X \Big((\theta(t)-t)^2 f(t) + \lambda(t)\Big(f(t) - |\dot\theta(t)|g(\theta(t))\Big)\Big)\,\mathrm{d}t, \tag{3.5}$$

for $\theta \in \{s \in C(X;Y) \mid s(t_r) = \theta_r\}$, where $C(X:Y)$ isthe set of continuous functions from $X$ to $Y$.

For a minimum to occur a necessary condition can be derived by calculating the first variation of $J[\theta;\lambda]$ or by using the Euler-Lagrange equation (derived by calculating the first variation in a general case). The first variation of $J[\theta;\lambda]$ with respect to $\theta(t)$ is given by

$$\begin{aligned}\tfrac{\mathrm{d}}{\mathrm{d}\epsilon}J[\theta+\epsilon\delta\theta;\lambda]\big|_{\epsilon=0} = \textstyle\int_{t_0}^{t_1}\delta\theta(t)\quad &\Big\{2(\theta(t)-t)f(t) - \lambda(t)|\dot\theta(t)|g'(\theta(t))\\ &+\tfrac{d}{dt}[\lambda(t)g(\theta(t))\operatorname{sgn}(\dot\theta(t))]\Big\}\,\mathrm{d}t\end{aligned} \tag{3.6}$$

for all possible variations $\delta\theta(t)$ continuously differentiable and subject to the boundary condition $\delta\theta(t_0) = \delta\theta(t_1) = 0$. This leads to the following equation which can also be calculated directly using the Euler-Lagrange equation (Euler's First equation)

$$2(\theta(t)-t)f(t) = \lambda(t)|\dot\theta(t)|g'(\theta(t)) - \frac{d}{dt}[\lambda(t)g(\theta(t))\operatorname{sgn}(\dot\theta(t))], \quad \forall t \in X. \tag{3.7}$$

Note that this corresponds to equation (2.13) in Evans [?evans2001]. We can also derive the variation with respect to $\lambda(t)$ which would yield the constraint equation. Using segmentation we can split a problem into subproblems, where we know or choose the sign of $\dot\theta(t)$. This way we can reformulate the functional (3.5). This split functional will no longer have absolute value bars, so the variations are easier. Suppose we have a problem that can be split into two segments such that $X = X^+ \bigcup X^-$. For the first segment $X^+$ we choose $\dot\theta(t) > 0$, the second segment $X^-$ we choose $\dot\theta(t) < 0$. In accordance with the requirements formulated before we also define the corresponding target sets $Y^+$ with density $g^+$, and $Y^-$, with density $g^-$. Then we can write for the functional

$$
\begin{aligned}
J[\theta;\lambda] = & \int_X \left[(\theta(t)-t)^2 f(t) + \lambda(t)\cdot\Big(f(t) - g(\theta(t))|\dot\theta(t)|\Big)\right]\,\mathrm{d}t \\
= & \int_{X^+} \left[(\theta(t)-t)^2 f(t) + \lambda(t)\cdot\Big(f(t) - g^+(\theta(t))\dot\theta(t)\Big)\right]\,\mathrm{d}t \\
& + \int_{X^-} \left[(\theta(t)-t)^2 f(t) + \lambda(t)\cdot\Big(f(t) + g^-(\theta(t))\dot\theta(t)\Big)\right]\,\mathrm{d}t. \\
=: & \; J^+[\theta;\lambda] + J^-[\theta;\lambda].
\end{aligned}
\tag{3.8}
$$

If we look carefully at (3.8) we see that the two integrals are defined on disjoint domains, except the interfaces. Now we have the sum of two positive functionals defined on disjoint domains that have to be minimised with respect to $\theta(t)$. The functions that have to be varied to find the extrema are completely independent for the two functionals. This means both functionals can be optimised independently. So we now look for extrema of both the functionals $J^+[\theta;\lambda]$ and $J^-[\theta;\lambda]$, and then stitch the solutions together to create a solution for the combined problem.

Looking at the first segment, the Euler-Lagrange equation with respect to $\theta$ (here $F$ denotes the integrand in the functional), i.e.,

$$
\frac{\partial F}{\partial\theta} - \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial F}{\partial\dot\theta}\right) = 0, \tag{3.9}
$$

gives together with the EL equation w.r.t. $\lambda$ that the extremal of $J^+$ is given by the solution of

$$
\begin{cases} 0 & = 2f(t)(\theta(t)-t) + \dot\lambda(t)(g^+)(\theta(t)) \\ 0 & = f(t) - g^+(\theta(t))\dot\theta(t) \end{cases} \qquad \forall t \in X^+ \tag{3.10}
$$

This can be done likewise for the other segment. The first equation in (3.10) is equivalent with (3.6) where 1 is substituted for $\operatorname{sgn}(\dot\theta(t))$, demonstrating the simplification created by choosing a sign for $\dot\theta(t)$.

### 3.2.1 Examples

To show how segmenting works, two examples is given. A simple quadratic cost function is used. A one-segment solution is compared to a two-segment solution. This is done by assuming $\theta(t)$ is smooth, i.e., at least continuously differentiable, on each segment and directly solving (3.2). Then the solution is put into the cost functional

$$
C[\theta] = \int_{t_0}^{t_2} (\theta(t)-t)^2 f(t)\,\mathrm{d}t, \tag{3.11}
$$

so we can compare different solutions with respect to this cost. $t_0$ and $t_2$ are the boundary points of the source set $X$ and $t_1$ is the location of the interface between the segments. To solve the individual segments, one can use the differential equation directly. Later the parameterised cost can be minimised directly, so no Euler-Lagrange equation is needed.

In this example the following situation is examined. We have a source with density $f(t) = 1$ for $-\pi/2 \le t \le \pi/2$. The target density is $g(\theta) = 1/2$ for $-\pi \le \theta \le \pi$. So we double the beam angle, and halve the intensity. It is obvious that the total intensity is the same for source and target. A one-segment and a two-segment solution are elaborated, the results of the other cases can be seen in Table 3.1.

### One increasing segment

A mapping with one increasing segment is proposed. This means we look for the solution of the following system

$$\begin{aligned} \dot{\theta}(t) &= \tfrac{f(t)}{g(\theta(t))} = \tfrac{1}{1/2} = 2, \qquad t \in (-\tfrac{\pi}{2}, \tfrac{\pi}{2}), \\ \theta(-\tfrac{\pi}{2}) &= -\pi. \end{aligned} \tag{3.12}$$

This is a very simple ODE, with solution $\theta(t) = 2t$. After some calculations based on (3.12) and (3.11) it can be shown that

$$C[\theta] = \frac{\pi^3}{12}. \tag{3.13}$$

There is no parameter so the cost is a constant.

### Two segments, first increasing, second decreasing

Now a solution is proposed consisting of two segments. Each segment illuminates a different segment of the target. $t_1 \in [t_0, t_2]$ is the boundary between the two segments. The first segment has an increasing solution, the second a decreasing solution. This gives the following system to be solved

$$\begin{aligned} &\left.\begin{aligned} \dot{\theta}(t) &= 2, \\ \theta(-\tfrac{\pi}{2}) &= -\pi \end{aligned}\right\} \quad -\tfrac{\pi}{2} \leq t \leq t_1, \\ &\left.\begin{aligned} \dot{\theta}(t) &= -2, \\ \theta(t_1) &= \pi \end{aligned}\right\} \quad t_1 \leq t \leq \tfrac{\pi}{2}. \end{aligned} \tag{3.14}$$

After solving we get the following solution

$$\theta(t) = \begin{cases} 2t & -\tfrac{\pi}{2} \leq t < t_1, \\ -2t + 2t_1 + \pi & t_1 \leq t \leq \tfrac{\pi}{2}, \end{cases} \tag{3.15}$$

resulting in a cost value of

$$C[\theta] = -\frac{2t_1^3}{3} + \pi t_1^2 - \frac{\pi^2 t_1}{2} + \frac{\pi^3}{6} \in (\frac{\pi^3}{12}, \frac{9\pi^3}{12}), \tag{3.16}$$

which is plotted in Figure 3.2. We see that the cost is minimal if $t_1 = \frac{\pi}{2}$. This corresponds with one segment with a positive sign for $\dot{\theta}(t)$.

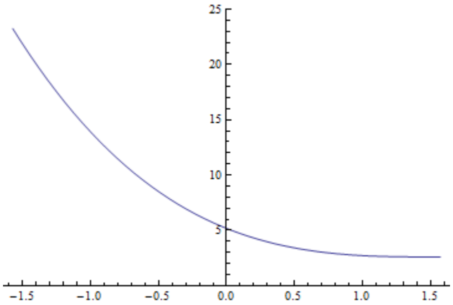


Figure 3.2: Plot of cost functional as function of $t_1$ for a two-segment solution, first segment increasing, second decreasing.

The results are shown in Table 3.1 (where + denotes a increasing segment and - denotes a decreasing segment). From this it can be deduced that a single rising segment is the best. For the configurations with two segments the best case is the limit where the separation point moves to the boundary and we have no crossing rays. This is as one would expect. A diverging solution means the rays are not refracted far, while for a converging solution all rays are sent into completely different directions.

Table 3.1: The costs for different segment configurations and the optimal $t_1$ under quadratic costs

| Configuration | Cost | best $t_1$ | best cost |
|---|---:|---:|---:|
| + | $\frac{\pi^3}{12}$ | | $\frac{\pi^3}{12}$ |
| - | $\frac{9\pi^3}{12}$ | | $\frac{9\pi^3}{12}$ |
| + - | $-\frac{2t_1^3}{3} + \pi t_1^2 - \frac{\pi^2 t_1}{2} + \frac{\pi^3}{6}$ | $t_1 = \frac{\pi}{2}$ | $\frac{\pi^3}{12}$ |
| - + | $\frac{2t_1^3}{3} + \pi t_1^2 + \frac{\pi^2 t_1}{2} + \frac{\pi^3}{6}$ | $t_1 = -\frac{\pi}{2}$ | $\frac{\pi^3}{12}$ |

Table 3.2: Table of costs depending on the boundary point, for different segment configurations.

| Configuration | Target | Cost | best $t_1$ | best cost |
|---|---|---:|---:|---:|
| + | | $\frac{\pi^3}{12}$ | | $\frac{\pi^3}{12}$ |
| - | | $\frac{9\pi^3}{12}$ | | $\frac{9\pi^3}{12}$ |
| + - | split | $\frac{5\pi^3-6\pi^2 t_1+12\pi t_1^2-8t_1^3}{48}$ | $t_1 = \frac{\pi}{2}$ | $\frac{\pi^3}{12}$ |
| - + | split | $\frac{5\pi^3+6\pi^2 t_1+12\pi t_1^2+8t_1^3}{48}$ | $t_1 = -\frac{\pi}{2}$ | $\frac{\pi^3}{12}$ |
| + + | shared | $\frac{3\pi^3-4\pi t_1^2}{24}$ | $t_1 = \pm\frac{\pi}{2}$ | $\frac{\pi^3}{12}$ |
| + - | shared | $\frac{\pi^3-\pi^2 t_1}{6}$ | $t_1 = \frac{\pi}{2}$ | $\frac{\pi^3}{12}$ |
| - + | shared | $\frac{\pi^3+\pi^2 t_1}{6}$ | $t_1 = -\frac{\pi}{2}$ | $\frac{\pi^3}{12}$ |
| - - | shared | $\frac{5\pi^3+4\pi t_1^2}{24}$ | $t_1 = 0$ | $\frac{5\pi^3}{24}$ |

### An example with different target splittings

In this example the source has again a constant distribution $f(t) = 1$ on $-\frac{\pi}{2} \leq t \leq \frac{\pi}{2}$. The target is $g(\theta) = 2$ on $\theta \in [0, \frac{\pi}{2}]$. This example is worked out for the one-segment and two-segment solutions. For the latter we consider two options from the infinite number of possibilities :

- Split targets: Both source and target are segmented. The upper segment is send to upper target, etc.
- Shared targets: Both source segments illuminate the whole target. The target distribution is modified according to the amount of light coming from each segment. Each target density is a scaled down version of the total density, proportional to the total light in the source segment.

For the one-segment solutions the problem statement is the same as in the previous example, with slightly different coefficients. For simplicity only the solutions are given. The results for this case are summarised in Table 3.2.

For illustration, the configuration with a shared target and with two segments, the first increasing and the second decreasing, is shown. This gives the following problem setup

$$\begin{aligned} &\left.\begin{aligned} f(t) &= 1 \\ g_1(\theta) &= \tfrac{2t_1+\pi}{\pi} \end{aligned}\right\} \quad -\tfrac{\pi}{2} \leq t \leq t_1, 0 \leq \theta \leq \tfrac{\pi}{2}, \\ &\left.\begin{aligned} f(t) &= 1 \\ g_2(\theta) &= \tfrac{\pi-2t_1}{\pi} \end{aligned}\right\} \quad t_1 \leq t \leq \tfrac{\pi}{2}, 0 \leq \theta \leq \tfrac{\pi}{2} \end{aligned} \tag{3.17}$$

where we note that $g_1(\theta) + g_2(\theta) = g(\theta)$ for all values of the parameter $t_1$. The targets are proportional to the size of the segment.

After solving (3.17) together with the conservation law and a boundary condition, we get the following solution

$$\theta(t) = \begin{cases} \frac{\pi}{2}\frac{\pi+2t}{\pi+2t_1}, & -\frac{\pi}{2} \leq t < t_1, \\ \frac{\pi}{2}\frac{\pi-2t}{\pi-2t_1}, & t_1 \leq t \leq \frac{\pi}{2} \end{cases} \tag{3.18}$$

resulting in a cost value of

$$C[\theta] = \frac{\pi^3 - \pi^2 t_1}{6}. \tag{3.19}$$

We see that the cost it minimal if $t_1 = \frac{\pi}{2}$ at the boundary of the domain. This corresponds once more with one segment with a positive sign for $\dot{\theta}(t)$.

We conclude that the introduction of an extra degree of freedom in the form of the boundary between two segments, allows us to formulate a minimisation problem. The optimum for these examples was obtained in the case that they reduce to a single segment, which is what one would expect. This shows that a minimisation formulation for these kind of optical problems can have merits. As can be seen in (3.10) there is no direct need in the one-dimensional case as the differential equation for energy conservation can be solved. In higher dimensions, as we see in the next chapter, this equation has multiple unknowns and the whole exercise gets some real use. Although there is no direct need for minimisation, there is a use case also in the one-dimensional situation. An optical designer can use the extra freedom introduced to optimise her design with respect to other requirements or goals like colour mixing.

# Chapter 4

# Free-form Optics

In this chapter the asymmetrical situation is discussed. We start with a point source and derive a set of Euler-Lagrange equations. Then a different system based on a parallel beam of light is discussed.

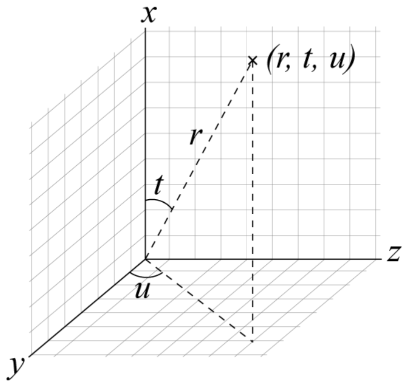


Figure 4.1: The coordinates system for the source distribution in 3D

## 4.1 Point source

The system we are looking at in general is very similar to the one described in Chapter 3 and Figure 3.1. However, as we have no symmetry we need two coordinates instead of one. The source rays can be represented as vectors on a subset of the unit sphere $X \subset S^2 \subset \mathbb{R}^3$. Using spherical coordinates a ray can be represented as $\mathbf{x} = (1, t, u) = \cos t \; \mathbf{i} + \sin t \cos u \; \mathbf{j} + \sin t \sin u \; \mathbf{k} \in X$, with $\mathbf{i}, \mathbf{j}$ and $\mathbf{k}$ the unit vectors of the Cartesian coordinate system and where $t$ is the angle between the ray and the positive $x$-axis, and $u$ is the azimuthal angle measured between the projection onto the $yz$-plane and the positive $y$-axis. For simplicity the radial coordinate is dropped, and any point on $S^2$ is identified by the pair $(t, u)$. This system is show in Figure 4.1. On this source we have a density function $f : X \to \mathbb{R}^+$, for brevity we often write $f = f(t, u)$. At the target we have a target set $Y \subset S^2$ and density function $g : Y \to \mathbb{R}^+$. The coordinates on the target are denoted by $(1, \theta, \phi) \in Y$. We are looking for a mapping $\mathbf{s} : X \to Y$ which is a vector field which maps $(t, u) \mapsto (\theta(t, u), \phi(t, u))$.

From here on $\mathbf{s}$ is used to denote the abstract mapping function, while $\theta$ and $\phi$ are used for the components. For readability the dependency on $t$ and $u$ are assumed for all appropriate functions from here on, and not written out explicitly, e.g., read $g(\theta, \phi)$ as $g\left(\theta(t,u), \phi(t,u)\right)$.

The governing principle is conservation of luminous flux (as special form of cosnervation of energy), as described by

$$\iint_{\hat{X}} f(\mathbf{x})\, \mathrm{d}S = \iint_{\mathbf{s}(\hat{X})} g(\mathbf{y})\, \mathrm{d}S, \tag{4.1}$$

for all connected closed subsets $\hat{X} \subset X$, and $\mathrm{d}S$ a surface element on $S^2$. In differential form this reads

$$f \sin t = g(\theta, \phi) \sin\theta \cdot |\theta_t \phi_u - \theta_u \phi_t| \qquad \forall (t,u) \in X, \tag{4.2}$$

where $\theta_t$ denotes the partial derivative of $\theta$ w.r.t. $t$ etc. For this problem to be well-posed it is required that $f$ and $g$ are positive continuous functions on $X$ and $Y$, respectively, and have finite and equal energy, i.e.,

$$\iint_X f(t,u) \sin t \,\mathrm{d}t\, \mathrm{d}u = \iint_Y g(\theta, \phi) \sin\theta \,\mathrm{d}\theta\, \mathrm{d}\phi < \infty. \tag{4.3}$$

The boundary condition needed for (4.2) follows from the requirement that we are looking for a mapping. This means we require

$$\mathbf{s}(X) = Y. \tag{4.4}$$

Unfortunately the system described here has no unique solution, because we have two unknown functions $\theta$ and $\phi$ with only one equation (4.2). There are many mappings that are able to transport the light in the correct way. For example imagine that we have two source rays with the same intensity, one could then swap their targets; although continuity would usually prevent this. This ambiguity is resolved by looking for the specific mapping among the ones that solve (4.2) that minimises

$$\iint_X c(\mathbf{x}, \mathbf{s}(\mathbf{x})) f(\mathbf{x})\, \mathrm{d}S, \tag{4.5}$$

for some cost function $c : X \times Y \to \mathbb{R}$. This formulation is called the Monge transport problem. Depending on the cost function, there exist theorems about existence and uniqueness of solutions. In the case of optics it is favourable to minimise the deflection of each light ray. In mathematical terms this means that the angle between a source ray, and the corresponding target ray should be as small as possible. This would lead to the cost function

$$\begin{aligned} c(t,u,\theta,\phi) &= \arccos\left[(\cos t\ \mathbf{i} + \sin t \cos u\ \mathbf{j} + \sin t \sin u\ \mathbf{k}) \cdot (\cos\theta\ \mathbf{i} + \sin\theta\cos\phi\ \mathbf{j} + \sin\theta \sin\phi\ \mathbf{k})\right] \\ &= \arccos\left[\cos t \cos(\theta) + \cos(u - \phi) \sin t \sin(\theta)\right]. \end{aligned} \tag{4.6}$$

### 4.1.1 Euler-Lagrange equations

In this section we derive a system of equations whith a solution that solves the minimisation problem formulated at the beginning of the chapter. For simplicity the problem is restricted to the case with one single segment with a positive sign for the Jacobian. First combine (4.2), (4.5) and (4.6) in a more formal way. Define a set of admissible functions

$$F_{\text{adm}} = \left\{ \mathbf{s} = (\theta, \phi) : X \to Y, \;\middle|\; f \sin t = g(\theta, \phi) \cdot |\theta_t \phi_u - \theta_u \phi_t| \sin \theta, \quad \mathbf{s} \text{ is continuous} \right\}. \tag{4.7}$$

Then we look for

$$\min_{\mathbf{s} \in F_{\text{adm}}} \iint_X c(t, u, \theta, \phi) f(t, u) \sin t \, \mathrm{d}t \, \mathrm{d}u. \tag{4.8}$$

This can be combined using a Lagrange multiplier $\lambda(t, u)$. This means we look for a function $\mathbf{s} : X \to Y$ that is an extremal for

$$J[\theta, \phi; \lambda] = \iint_X \big( c(t, u, \theta, \phi) f(t, u) \sin t + \lambda(t, u) \cdot (f \sin t - g(\theta, \phi) \sin \theta \cdot |\theta_t \phi_u - \theta_u \phi_t|) \big) \, \mathrm{d}t \, \mathrm{d}u. \tag{4.9}$$

To determine equations whose solution is the extremal of $J[\theta, \phi; \lambda]$ the first variation with respect to $\theta$, $\phi$ and $\lambda$ has to be calculated.

An extremal of this functional can be found by solving a set of PDEs, which can be derived using the Euler-Lagrange equation (or by calculating the First Variation). The Euler-Lagrange equation with respect to $\theta$ is given by

$$\frac{\partial F(\theta, \phi, \lambda)}{\partial \theta} - \frac{\partial}{\partial t} \left[ \frac{\partial F(\theta, \phi, \lambda)}{\partial \theta_t} \right] - \frac{\partial}{\partial u} \left[ \frac{\partial F(\theta, \phi, \lambda)}{\partial \theta_u} \right] = 0, \tag{4.10}$$

where $F$ is the integrand of the integral in (4.9) and equivalent equations can be written for $\phi$ and $\lambda$. The derivation is analog to Section 3.2. Just as in the rotational symmetric system (see Section 3.1), we can define segments to enable us to get rid of the absolute value of the determinant of the Jacobi matrix. For now it is assumed the sign of the Jacobian is positive. This results in the PDEs

$$\begin{cases} 0 = & g(\theta, \phi) \left( \phi_u \lambda_t - \lambda_u \phi_t \right) \sin \theta + f c_\theta(t, u, \theta, \phi) \sin t, \\ 0 = & g(\theta, \phi) \left( \lambda_u \theta_t - \theta_u \lambda_t \right) \sin \theta + f c_\phi(t, u, \theta, \phi) \sin t, \\ 0 = & f \sin t - g(\theta, \phi) \left( \theta_t \phi_u - \theta_u \phi_t \right) \sin \theta, \end{cases} \tag{4.11}$$

where $c_\theta$ is the derivative of $c$ w.r.t. $\theta$, and $c_\phi$ w.r.t. $\phi$. Substituting (4.6) in (4.11) results in

$$\begin{cases} 0 = & g(\theta, \phi) \left( \phi_u \lambda_t - \lambda_u \phi_t \right) \sin \theta - f \frac{\cos \theta \cos(u - \phi) \sin t - \cos t \sin \theta}{\sqrt{1 - (\cos t \cos \theta + \cos(u - \phi) \sin t \sin \theta)^2}} \sin t, \\ 0 = & g(\theta, \phi) \left( \lambda_u \theta_t - \theta_u \lambda_t \right) \sin \theta + f \frac{\sin t \sin \theta \sin(u - \phi)}{\sqrt{1 - (\cos t \cos \theta + \cos(u - \phi) \sin t \sin \theta)^2}} \sin t, \\ 0 = & f \sin t - g(\theta, \phi) \left( \theta_t \phi_u - \theta_u \phi_t \right) \sin \theta. \end{cases} \tag{4.12}$$

Unfortunately, nor (4.11), nor (4.12) does not bring us much further in finding the mapping. This is a system of three coupled nonlinear PDEs. Moreover it is not really clear what boundary conditions should be applied or how they should be treated. However, it is known in optimal mass transport [?evans2001] that for a quadratic cost function, the mapping can be written as the gradient of some convex potential $v(t, u)$. So we have $\mathbf{s} = \nabla v$ or in components, $\theta = v_t$ and $\phi = v_u$. This would clearly simplify the equations. The energy conservation equation (the third in (4.11)) would become

$$\det(D^2 v) = \frac{f \sin t}{g(\nabla v) \sin \theta}. \tag{4.13}$$

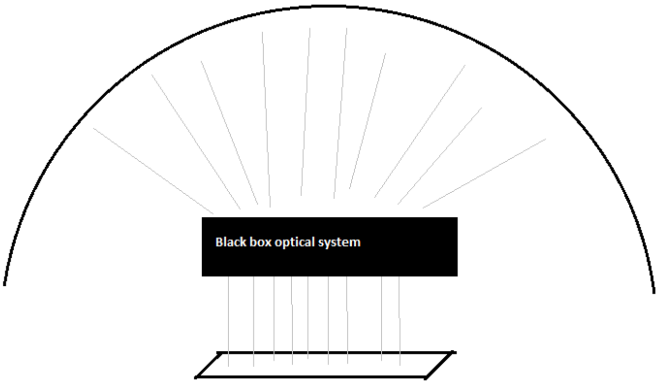


Figure 4.2: Sketch of the set-up for parallel beam optical system

This is an equation of the Monge-Ampère (MA) type. The main motivation for this analysis is that we know that for certain optical problems the relevant equation is of this MA type. Furthermore, we have the physical analogy between the transport of light and the displacement of, say, a pile of sand. Together these two facts make a compelling case to look for the relation between optimal mass transport and optical design. Unfortunately an explanation is not available (yet). In the next section the optical system for which we have an MA type equation are discussed.

## 4.2 Parallel beam

In this section a different optical setup is used, see Figure 4.2. The source is a set $X \subset \mathbb{R}^2$ in the $xy$-plane that emits a parallel beam in the $z$-direction. On this source the emittance is given by $f(x, y)$ in [lm/m$^2$], using Cartesian coordinates. The light will then hit an optical surface that is either a reflector or a refractive surface. The light then creates a certain intensity distribution in the far field. The goal is to determine the surface needed such that a certain prescribed intensity pattern $\hat{G}(\theta, \phi)$, with units [lm/sr], is created in the far field. Here $\theta$ and $\phi$ are the spherical coordinates as defined in the beginning of this chapter, and we assume $\hat{G}$ is defined on a domain $\hat{Y} \subset S^2$.

Just as for the point source we have the conservation of luminous flux and can therefore write

$$\iint_{\tilde{X}} f(\mathbf{x})\,\mathrm{d}S = \iint_{\mathbf{s}(\tilde{X})} \hat{G}(\mathbf{y})\,\mathrm{d}S, \tag{4.14}$$

for all $\tilde{X} \subset X$, or equivalently in coordinates

$$\iint_{\hat{X}} f(x, y)\,\mathrm{d}x\,\mathrm{d}y = \iint_{\mathbf{s}(\hat{X})} \hat{G}(\theta, \phi) \sin\theta\,\mathrm{d}\theta\,\mathrm{d}\phi. \tag{4.15}$$

It turns out that for certain optical problems, as we see later, a Monge-Ampère type equation can be derived. These equations are all of a similar form and therefore we can write a general problem description, and hopefully build a general solver. For now let $v : X \to \mathbb{R}$ be a function describing the unknown optical surface, and let $Y \subset \mathbb{R}^2$ be the target set on which a target density $g : Y \to \mathbb{R}$ is defined. This set $Y$ is not $\hat{Y}$ and are defined later. $\hat{Y}$

is the physical space on which $\hat{G}$ is defined, while $Y$ is a parameter space on which $G$ and $g$ are functions. For each optical problem we have a luminous intensity $\hat{G}$ on a set $\hat{Y}$. Using a transformation we are able to rewrite the problem such that a new density $g : Y \to \mathbb{R}$ is created. The general problem then becomes, using the notation $D^2v$ for the Hessian matrix of $v$. Find $v$ convex such that

$$\begin{aligned} &\det(D^2 v(\mathbf{x})) = f(\mathbf{x})/g(\nabla v(\mathbf{x})) \quad \text{for } \mathbf{x} \in X, \\ &\nabla v(X) = Y, \end{aligned} \tag{4.16}$$

In this setting $v$ is a potential function that is closely related to the optical surface and whose gradient $\mathbf{s} = \nabla v$ is exactly the mapping we were looking for.

During current, ongoing and as of now unpublished research, Prins [Pri12] has derived a set of equations, whose solution describes the surface of a reflector and a lens. To my understanding the derivation is based on the physical laws of reflection and refraction in vector form, together with a relation between the location of the surface and the surface normals.

### 4.2.1 Reflector

For a reflector described as $v : X \to \mathbb{R}$ Prins [Pri12] has shown that the solution is given by

$$\left|\det(D^2 v(x,y))\right| = \frac{f(x,y)(v_x^2 + v_y^2 + 1)^2}{4G(v_x, v_y)}, \tag{4.17}$$

and

$$G(v_x, v_y) = \hat{G}\left(\arccos\left(1 - \frac{2}{v_x^2 + v_y^2 + 1}\right), \; \arctan\left(\frac{v_y}{v_x}\right)\right). \tag{4.18}$$

Introducing the function

$$g(v_x, b_y) = \frac{4G(v_x, v_y)}{(v_x^2 + v_y^2 + 1)^2}, \tag{4.19}$$

and choosing the positive sign for the determinant (which can be done using the segmentation process as described in Section 3.1) the equation (4.17) can be written as the Monge-Ampère equation

$$\det(D^2 v(x,y)) = f(x,y)/g(\nabla v(x,y)), \tag{4.20}$$

where $f : X \to \mathbb{R}, g : Y \to \mathbb{R}$ and with the boundary condition $\nabla v(X) = Y$. Here $Y \subset \mathbb{R}^2$ is defined as the set $Y = \left\{(y_1, y_2) \in \mathbb{R}^2 \middle| \left(\arccos\left(1 - \frac{2}{y_1^2+y_2^2+1}\right), \arctan\left(\frac{y_2}{y_1}\right)\right) \in \hat{Y}\right\} = \mathbf{s}^{-1}(\hat{Y})$.

### 4.2.2 Lens

For a refractive surface Prins, again defines a new transformed function

$$G(v_x, v_y) = \hat{G}\left(\arccos\left(\frac{n_l(v_x^2 + v_y^2) + \sqrt{1 + (1 - n_l^2)(v_x^2 + v_y^2)}}{v_x^2 + v_y^2 + 1}\right), \; \arctan\left(\frac{v_y}{v_x}\right)\right), \tag{4.21}$$

and the corresponding Monge-Ampère equation is given by

$$|\det(D^2 v(x,y))| = \frac{(v_x^2 + v_y^2 + 1)^2\sqrt{1 + (1 - n_l^2)(v_x^2 + v_y^2)}}{\left(\sqrt{1 + (1 - n_l^2)(v_x^2 + v_y^2)} - n_l\right)^2}\frac{f(x,y)}{G(v_x, v_y)}. \tag{4.22}$$

Here $n_l = n_1/n_2$ with $n_1$ the refractive index of the material before the surface and $n_2$ the index of refraction after the optical surface. In a similar way as above this can be rewritten in the form of (4.20). As usual in optics one can see that the reflector equation is the limit case of the lens by taking $n_l = -1$.

# Chapter 5

# Numerical Solution of the Monge-Ampère Equation

The goal of this chapter is to find a luminous flux conserving mapping $\mathbf{s} = \nabla u$ that transports the energy with density $f(\mathbf{x})$, defined on a set $X \subset \mathbb{R}^d$ to a density $g(\mathbf{y})$ on a set $Y \subset \mathbb{R}^d$. The goal is to find a numerical solution for the potential $u : X \to \mathbb{R}$ satisfying the following Monge-Ampère (MA) problem

$$\begin{aligned}
&\det(D^2u(\mathbf{x})) = f(\mathbf{x})/g(\nabla u(\mathbf{x})) + c_r u(\mathbf{x}_r) \quad \text{for } \mathbf{x} \in X \\
&\nabla u(X) = Y, \\
&u \text{ is convex.}
\end{aligned} \tag{5.1}$$

In this system $c_r > 0$ is a weighting factor, and $\mathbf{x}_r$ is some reference point $\mathbf{x}_r \in X$. This addition is necessary to guarantee the uniqueness of the discrete solution, as otherwise $u$ is only determined up to a constant. The expression $\nabla u(X) = Y$ is called the transport boundary condition (TBC). Although we introduced the potential in the previous chapter with the letter $v$, it is usually denoted with $u$ in the literature. For that reason we use $u$ from here on.

In this chapter a discretisation of te Monge-Ampère equation (MA) is discussed. In Section 5.1 earlier literature is discussed. Section 5.2 describes the general outline of the numerical algorithm, with the discretisation scheme described in Section 5.3. Two schemes are discussed in Section 5.4 and Section 5.5 for the stable and the accurate sub schemes. The boundary treatment is considered in Section 5.6. Lastly, some implementation details are left to Section 5.7.

## 5.1 Literature on solving Monge-Ampère

When discussing the earlier numerical work on solving the MA equation, a distinction has to be made between different classes of problems that are solved. Most papers only consider the MA equation combined with Dirichlet boundary conditions. Among those are the early works of Froese and Oberman [BFO10, FO11a, FO11b, FO12], who developed a sophisticated discretisation based on finite difference operators. Other methods are by Dean and Glowinsky[DG06], which are restricted to regular smooth data, and gradient descent methods by Chartrand et al. [CVWB09]. An important, and often mentioned approach as it was the first, is the method based on the computational fluid mechanical approach by Benamou and Brenier

[BB00]. The biggest disadvantage of their method is the introduction of another dimension which increases the computational cost dramatically. This is also relevant for most of the gradient descent methods mentioned before. A method based on a Lagrangian formulation is by Haber, Rehman and Tannenbaum [HRT10]. A finite element approach is proposed by Brenner and Neilan [BN12]. For periodic boundary conditions an algorithm was proposed by Saumier [SAK10]

Solving the MA equation combined with the transport boundary condition, as relevant for our problems, has not received much attention yet. There is a paper by Sulman et al. [SWR11] which is based on finding a fixed point of a parabolic version of the MA equation on a rectangular domain. Recently, Froese has developed two methods for solving the MA eq. with the transport boundary condition. Her first approach [Fro12] is based on an iterative scheme, where in each iteration the MA equation is solved with Neumann BCs which are updated during the iteration. The second approach [BFO12c] is the one that is used in this report. This approach is based on creating a signed distance function that defines the location of the boundary for the target set. It builds on the earlier work of Oberman and Froese for the interior with some improvements. The paper describing the algorithm [BFO12c], was later split in a theoretical paper [BFO12b] and a numerical paper [BFO12a].

### 5.1.1 Notes on regularity and convergence

Froese and Oberman comment on the existing literature on this topic [FO11b]. They claim that most of the other approaches can not handle singular solutions very well, while their method can. The following problems/difficulties for solving (5.1) are identified

**Weak solutions** For a lot of problems the solution needs to be found in a suitably weak sense. The numerical methods must be able to find singular solutions. In this case the notion used is the viscosity solution [FO11a, Section 2.2].

**Convexity** Uniqueness requires that $u$ is convex, otherwise $-u$ is also a solution (the one corresponding to crossing rays). One does not want a numerical method to switch to one of those randomly, or be stuck in the middle.

**Accuracy** For two-dimensional problems or higher the *directional resolution* of a discretisation scheme becomes important. However Froese et al. also note that for singular solutions the accuracy is always low, regardless of the discretisation method used.

**Computational cost** The computational cost of different techniques differs a lot. However, the cost of a method may depend on the singularity of the problem. This means there might not be a universal fastest solver. See also [FO12].

There are two important types of ‘singularities‘. The first is seen on the left in Figure 5.1. In this case the target intensity $g$ has very small values and so we are almost dividing by zero at the right hand sight of (5.1). This causes the mapping to smear out the light from the source. If this effect is large there is a big distance between the images of source grid points. The second type of singularity can be seen on the right of Figure 5.1. Here $f$ takes on very small values resulting in a mapping from many places to almost the same point in the target. This means the right hand side of (5.1) becomes almost zero. This of course means the potential will locally almost loose its convexity. The biggest problem is that the solution provides no longer an invertible mapping.

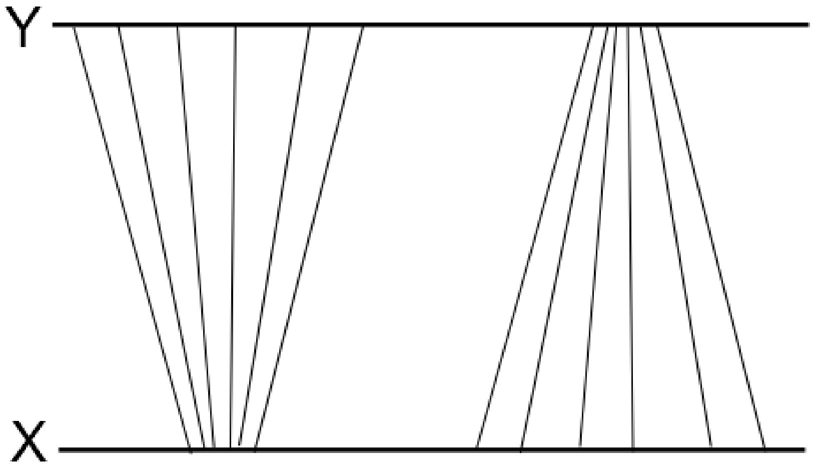


Figure 5.1: An example of solutions which are almost singular.

## 5.2 Numerical approach

The numerical method is explained in the following steps. First the most abstract structure of the solver is explained, including some basic definitions around the implementation. Secondly the two methods for the discretisation of the Monge-Ampère operator are described. Finally the handling of the boundary is shown.

Before continuing with the discretisation, the constraints on the problem should be specified. Froese et al. [BFO12c] have proved that her scheme converges under the following conditions

1. $f$ and $g$ are both $L^1$ with $\iint_X f(\mathbf{x})\,\mathrm{d}\mathbf{x} = \iint_Y g(\mathbf{y})\,\mathrm{d}\mathbf{y}$.
2. $f$ must be a nonnegative function.
3. $g$ must be Lipschitz continuous and strictly positive.
4. The target domain $Y$ *must* be convex.
5. The source domain $X$ should theoretically be convex, although in practice it turns out some nonconvex domains can still be handled.

In Chapter 6 it is shown that for certain nonconvex source domains, the algorithm still produces reasonable results. Note that the only restriction on $f$ is integrability, no continuity is assumed.

### 5.2.1 Grid points, domain and other technical details

We start with the discretisation of the computational domain $X$. For simplicity it is assumed that $X$ is a rectangular domain. This can always be constructed by extending $f$ with 0's. So we write $X = [a_1, b_1] \times [a_2, b_2] \subset \mathbb{R}^2$. We discretize this using $M_x$ and $M_y$ points for respectively the $x_1$ and $x_2$-direction, such that the grid size is constant, so $h = (b_1 - a_1)/(M_x - 1) = (b_2 - a_2)/(M_y - 1)$. This means that a grid point $\mathbf{x}_{i,j} \in X$ is given by

$$\mathbf{x}_{i,j} = \begin{bmatrix} a_1 + h(i-1) \\ a_2 + h(j-1) \end{bmatrix}, \tag{5.2}$$

for $i = 1, \ldots, M_x$ and $j = 1, \ldots, M_y$. For the solution $u$ we introduce the following approximation

$$u(\mathbf{x}_{i,j}) \approx u_{i,j}. \tag{5.3}$$

At several stages in the algorithm we need to create a vector where each element corresponds to a location in the two-dimensional grid. Therefore it is convenient to define a uniform way of doing this. For implementation reasons the indexing scheme chosen coincides with the memory layout of MATLAB matrices. The conversion is given by

$$k = (j-1)M_x + i, \qquad \text{for } i = 1, \ldots, M_x \quad j = 1, \ldots, M_y, \tag{5.4}$$

implying $k = 1, \ldots, M_x M_y$ and this means for example $\mathbf{x}_{i,j} = \mathbf{x}_k = \mathbf{x}_{(j-1)M_x+i}$.

### 5.2.2 Structure of the solver

The algorithm uses Newton's method (cf. [MRtTB05, sec 9.6]) to solve the discrete system. This is obtained after discretising the Monge-Ampère equation for the interior domain. At each interior grid point a discretised equation is formulated. Together with a discretised boundary condition at each boundary point this gives $M = M_x \cdot M_y$ coupled nonlinear equations in the $M$ unknowns at the grid points. This system is denoted as

$$\mathbf{N}(\mathbf{u}) = \mathbf{0}. \tag{5.5}$$

This nonlinear system can be solved using Newton iteration for which we need the Jacobi matrix of the system, defined as

$$\mathbf{J}_{i,j}(\mathbf{u}) = \frac{\partial N_i(\mathbf{u})}{\partial u_j}. \tag{5.6}$$

We can then create the following iterative procedure for solving. Start with some initial guess $\mathbf{u}^0$. Then solve in each iteration $\mathbf{J}(\mathbf{u}^n)\mathbf{w} = -\mathbf{N}(\mathbf{u}^n)$ and update the iterant according to $\mathbf{u}^{n+1} = \mathbf{u}^n + \beta\mathbf{w}$. Here $\beta \in (0, 1]$ is a damping factor, it is chosen such that $\|\mathbf{N}(\mathbf{u}^{n+1})\|_1 < \|\mathbf{N}(\mathbf{u}^n)\|_1$. This means we always have a contraction to a solution, or we stop for not converging at all. In practise we check if $\|\mathbf{u}^n + \beta\mathbf{w}\|_1 < \|\mathbf{N}(\mathbf{u}^n)\|_1$. If this is not the case $\beta$ is halfed and we try again until it is true and we can do the update. If $\beta$ becomes very small ($\leq 10^{-10}$) we stop anyway with a warning that we are not converging.

## 5.3 The hybrid or filtered scheme

A hybrid scheme is based on a stable monotone scheme, which exhibits convergence to the solution, although it is relatively inaccurate. This stable scheme can also handle singular solutions. This is then combined with a computationally cheaper and more accurate scheme. However this accurate scheme only works for regular smooth solutions. The trick is then to combine them in such a way that in parts of the domain where the solution is regular the more accurate scheme is used, and in the singular regions it reduces to the stable scheme. To this end a filter function is used for the selection of the scheme [FO12, Definition 1].

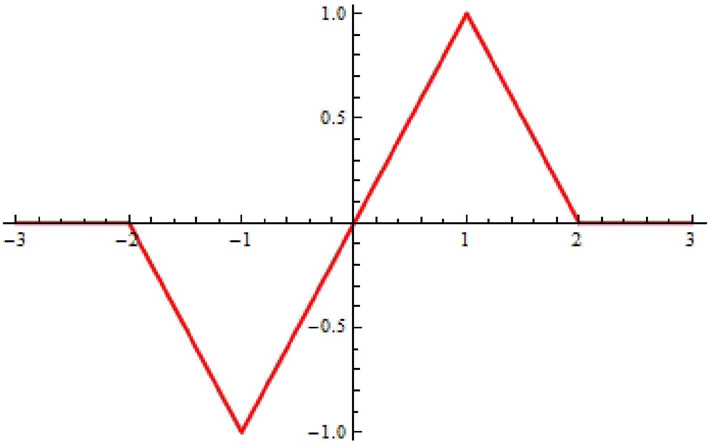


Figure 5.2: The filter function $S(x)$.

**Definition 5.3.1.** *(Filter function). A filter function is a continuous, bounded function $S$, which is equal to the identity function in a neighbourhood of the origin and vanishes for large arguments.*

As an example, also the filter function we use in the implementation, (see Figure 5.2) Benamou et al. [BFO12c] propose

$$S(z) = \begin{cases} z & |z| \leq 1, \\ 0 & |z| \geq 2, \\ -z+2 & 1 \leq z \leq 2, \\ -z-2 & -2 \leq z \leq -1. \end{cases} \tag{5.7}$$

This is used in general to create a scheme (in the interior domain) of the form

$$\mathbf{N}_k = \mathbf{N}_{M,k} + \epsilon(h,\, \mathrm{d}\alpha) S\left(\frac{\mathbf{N}_{A,k} - \mathbf{N}_{M,k}}{\epsilon(h,\, \mathrm{d}\alpha)}\right), \tag{5.8}$$

where $\mathbf{N}$ is the filtered scheme (to be combined with the boundary data), $\mathbf{N}_A$ is the accurate scheme, $\mathbf{N}_M$ is the stable monotone scheme, $S$ must be seen as point wise evaluation on the vector argument and $\epsilon(h,\, \mathrm{d}\alpha)$ is a user chosen parameter, such that $\epsilon(h,\, \mathrm{d}\alpha) \to 0$ when $h,\, \mathrm{d}\alpha \to 0$. $\mathrm{d}\alpha$ is the direction discretisation parameter defined in Section 5.4. One can note that the difference between the filtered scheme and the monotone scheme goes to zero as $h \to 0$, and therefore convergence can be proved; see [FO12]. In [BFO12c] a value of $\epsilon(h,\, \mathrm{d}\alpha) = \sqrt{h} + d\alpha/10$ is used, determined by testing different values. Note that if $\mathbf{N}_A$ and $\mathbf{N}_M$ are sufficiently close together, $S$ is the identity and $\mathbf{N} = \mathbf{N}_A$. Moreover, when the difference is large then $S = 0$ and the scheme reduces to the stable scheme.

## 5.4 Stable Discretisation

The stable discretisation is based on the observation that by looking in a special direction for taking the derivatives the Hessian can be diagonalised. The convexity constraint is incorporated in the MA-operator. This scheme is made to be able to find weak solutions to the problem [BFO12c].

Note that the Hessian matrix $D^2u(\mathbf{x})$ is a symmetric matrix with real eigenvalues $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_d$ where $d$ is the dimension. For our implementation $d = 2$. By definition $u$ is convex iff $\det(D^2u(\mathbf{x})) \geq 0$. So the $\det(D^2u(\mathbf{x}))$-operator and the convexity constrained can be combined by observing that the function $u$ is convex if the smallest eigenvalue $\lambda_1$ of $D^2u$ is nonnegative $\lambda_1 \geq 0$. This is combined into the operator by defining for any symmetric matrix $\mathbf{M}$ with eigenvalues $\lambda_1, \ldots, \lambda_d$

$$\det{}^+(\mathbf{M}) = \prod_{j=1}^{d} \lambda_j^+ \tag{5.9}$$

where $(\cdot)^+ := \max\{\cdot, 0\}$ and similarly $(\cdot)^- := \min\{\cdot, 0\}$. Note that although the convexity constraint is now absorbed into the operator, which is good, we lost the differentiability of the operator near singular matrices. In practice the 0 is replaced by a small parameter $\delta > 0$ to make sure it is strictly positive. A typical value is $\delta = 10^{-6}$. We use this $\det^+$ operator instead of det for determining numerical solutions of our problem.

The next step is to define a set $V$ of all orthonormal bases of $\mathbb{R}^d$ as

$$V = \left\{ (\boldsymbol{\nu}_1, \ldots, \boldsymbol{\nu}_d) \middle| \boldsymbol{\nu}_j \in \mathbb{R}^d, \boldsymbol{\nu}_i \perp \boldsymbol{\nu}_j \text{ if } i \neq j, \|\boldsymbol{\nu}_j\|_2 = 1 \right\}.$$

In [FO11a, Lemma 2] it is proved that for a symmetric positive definite matrix $\mathbf{M} \in \mathbb{R}^{d\times d}$ with eigenvalues $\lambda_j$

$$\det(\mathbf{M}) = \prod_{j=1}^{d} \lambda_j = \min_{(\boldsymbol{\nu}_1,\ldots,\boldsymbol{\nu}_d)\in V} \prod_{j=1}^{d} \boldsymbol{\nu}_j^T \mathbf{M} \boldsymbol{\nu}_j. \tag{5.10}$$

The MA operator, given by $\det(D^2u)$ can now be defined as

$$\det(D^2u) = \min_{(\boldsymbol{\nu}_1,\ldots,\boldsymbol{\nu}_d)\in V} \prod_{j=1}^{d} u_{\boldsymbol{\nu}_j\boldsymbol{\nu}_j}, \tag{5.11}$$

where $u_{\boldsymbol{\nu}_j\boldsymbol{\nu}_j}$ is the second derivative of $u$ along the direction $\boldsymbol{\nu}_j$. This operator can be regularised by bounding, away from zero, as follows

$$\det{}^+(D^2u) = \min_{(\boldsymbol{\nu}_1,\ldots,\boldsymbol{\nu}_d)\in V} \prod_{j=1}^{d} \left(u_{\boldsymbol{\nu}_j\boldsymbol{\nu}_j}\right)^+. \tag{5.12}$$

Obviously, the value of the determinant is independent of the coordinate system used. We also know that there exists a basis such that the matrix is diagonal, i.e., the basis of eigenvectors. Therefore this minimum is realised by the basis consisting of the eigenvectors.

In [Fro12], Froese adds an extra term to (5.12) to decrease any nonconvexity. This term guarantees that if the iterant is locally nonconvex the operator takes on a negative value. This leads to a large residual and therefore prevents concave solutions. This modification gives

$$\det{}^+(D^2u) = \min_{(\boldsymbol{\nu}_1,\ldots,\boldsymbol{\nu}_d)\in V} \left\{ \prod_{j=1}^{d} \max\{u_{\boldsymbol{\nu}_j\boldsymbol{\nu}_j}, \delta\} + \sum_{j=1}^{d} \min\{u_{\boldsymbol{\nu}_j\boldsymbol{\nu}_j}, \delta\} \right\}. \tag{5.13}$$

Next a discrete set of ortho*gonal*, not ortho*normal*, vectors $\mathcal{G}$ is chosen with a directional discretisation parameter $d\alpha$. The derivatives are then discretised in space with parameter $h$ using central differences. The discretised Monge-Ampère operator becomes

$$MA^M[u] \equiv \min_{(\boldsymbol{\nu}_1,\ldots,\boldsymbol{\nu}_d)\in V} \left\{ \prod_{j=1}^{d} \left(\mathcal{D}_{\boldsymbol{\nu}_j\boldsymbol{\nu}_j} u\right)^+ + \sum_{j=1}^{d} \left(\mathcal{D}_{\boldsymbol{\nu}_j\boldsymbol{\nu}_j} u\right)^- \right\} \tag{5.14}$$

where

$$\mathcal{D}_{\boldsymbol{\nu}\boldsymbol{\nu}} u_i = \frac{1}{|\boldsymbol{\nu}|^2 h^2} \left(u(x_i + \boldsymbol{\nu}h) + u(x_i - \boldsymbol{\nu}h) - 2u(x_i)\right)$$

is the finite difference operator in the $\boldsymbol{\nu}$ direction, which should be on the discretisation grid. Some of these directions make wide stencils. At the boundary one has to use interpolation to fill in for the missing grid points. Note that $u_i$ here is the discrete solution indexed using a single index as described in Section 5.2.1

For the Jacobi matrix, Benamou et al. [BFO12c] show that the gradient $\nabla \mathbf{N}_M[u]$ is equal to the gradient of the argument of the minimum. So the gradient for the Jacobi Matrix is evaluated using the 'active' basis for each point. For more detailed descriptions of the Jacobi matrix see appendix B.

For the evaluation of the right hand side of the equation (5.1) we need to discretise the gradient of $u$. Froese [Fro12] has shown that a simple discretisation is not a good approach. Therefore the numerical derivatives are taken along the same directions (basis) as are used in the discrete MA-operator. The gradient is then written as a linear combination of these derivatives, i.e.,

$$\nabla u = \left( \sum_{j=1}^{d} \frac{\boldsymbol{\nu}_j \cdot \mathbf{e}_1}{|\boldsymbol{\nu}_j|} u_{\boldsymbol{\nu}_j}, \ldots, \sum_{j=1}^{d} \frac{\boldsymbol{\nu}_j \cdot \mathbf{e}_d}{|\boldsymbol{\nu}_j|} u_{\boldsymbol{\nu}_j} \right), \tag{5.15}$$

and $u_{\boldsymbol{\nu}_j}$ is approximated using central differences

$$u_{\boldsymbol{\nu}_j} = \frac{1}{2|\boldsymbol{\nu}_j|h} (u(x_i + \boldsymbol{\nu}_j h) - u(x_i - \boldsymbol{\nu}_j h)). \tag{5.16}$$

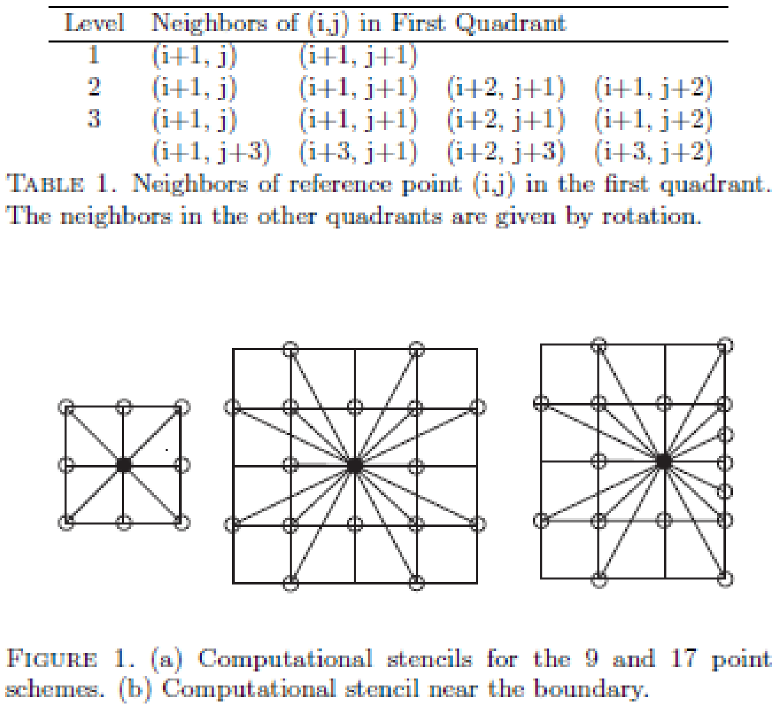
Combining everything from this section gives the following discretisation for the Monge-Ampère equation. The discretisation is done using a finite number of directions, the set of bases vectors is called $\mathcal{G}$, resulting in the following formula for the residual of the stable scheme:

$$\begin{aligned} \mathbf{N}_m[u] = \min_{(\boldsymbol{\nu}_1,\ldots,\boldsymbol{\nu}_j)\in\mathcal{G}} &\left\{ \prod_{j=1}^{d} \max\{\mathcal{D}_{\boldsymbol{\nu}_j\boldsymbol{\nu}_j} u, \delta\} + \sum_{j=1}^{d} \min\{\mathcal{D}_{\boldsymbol{\nu}_j\boldsymbol{\nu}_j} u, \delta\} \right\} \\ &- f(\mathbf{x})/g \left( \sum_{j=1}^{d} \frac{\boldsymbol{\nu}_j \cdot \mathbf{e}_1}{|\boldsymbol{\nu}_j|} u_{\boldsymbol{\nu}_j}, \ldots, \sum_{j=1}^{d} \frac{\boldsymbol{\nu}_j \cdot \mathbf{e}_d}{|\boldsymbol{\nu}_j|} u_{\boldsymbol{\nu}_j} \right), \end{aligned} \tag{5.17}$$

where the $(\cdot)^+$ operator has been regularised as $(\cdot)^+ = \max\{\cdot, \delta\}$ for some small parameter $\delta > 0$, and correspondingly $(\cdot)^- = \min\{\cdot, \delta\}$. This to bound the derivatives away from zero. A typical value is $\delta = 10^{-6}$.

### 5.4.1 Worked out stencils

In [Obe08] the different stencils used for the discretisation are described. In this subsection the numerical details of the different stencils is written out explicitly for the two-dimensional situation. In most examples they use a set of 9, 17 or 33 points stencils, this be replicated here, see Figure 5.3 for a visual representation.



| Level | Neighbors of (i,j) in First Quadrant | | | |
|---|---|---|---|---|
| 1 | (i+1, j) | (i+1, j+1) | | |
| 2 | (i+1, j) | (i+1, j+1) | (i+2, j+1) | (i+1, j+2) |
| 3 | (i+1, j) | (i+1, j+1) | (i+2, j+1) | (i+1, j+2) |
| | (i+1, j+3) | (i+3, j+1) | (i+2, j+3) | (i+3, j+2) |

TABLE 1. Neighbors of reference point (i,j) in the first quadrant. The neighbors in the other quadrants are given by rotation.

FIGURE 1. (a) Computational stencils for the 9 and 17 point schemes. (b) Computational stencil near the boundary.

Figure 5.3: Table 1 and Figure 1 from [Obe08]

The stencils are of the form

$$(\mathcal{D}_{\boldsymbol{\nu}\boldsymbol{\nu}} u_{i,j})^+ = D^+[i,j;a,b] = \frac{1}{(a^2+b^2)h^2}\left(u_{i+a,j+b} + u_{i-a,j-b} - 2u_{i,j}\right)^+ . \tag{5.18}$$

Note that if one of the vectors in a certain basis is given by $(a, b)$ the other vector is $(-b, a)$ by orthogonality. Therefore we can describe a basis in $\mathbb{R}^2$, with just the first vector.

#### The 9-point stencil

In this stencil the minimum is taken over two direction sets, i.e., $\mathcal{G} = \{(1,0),(1,1)\}\}$. This means that combined with (5.14), we get the following discrete operator for the interior of the domain

$$\begin{aligned} MA[u_{i,j}] = \min\{ \quad & D^+[i,j;1,0]D^+[i,j;0,1] + D^-[i,j;1,0] + D^-[i,j;0,1], \\ & D^+[i,j;1,1]D^+[i,j;-1,1] + D^-[i,j;1,1] + D^-[i,j;-1,1] \ \} \end{aligned} \tag{5.19}$$

For this stencil we do not need to do anything special near the boundaries.

### The 17-point stencil

This stencil incorporates also the directions of the 9-point stencil. The extra directions are $(2,1)$ and $(1,2)$, resulting in $\mathcal{G} = \{(1,0),(1,1),(2,1),(1,2)\}\}$. We get the following scheme.

$$MA[u_{i,j}] = \min_{(a,b)\in\mathcal{G}} \left\{ D^{+}[i,j;a,b]D^{+}[i,j;-b,a] + D^{-}[i,j;a,b] + D^{-}[i,j;-b,a] \right\} \tag{5.20}$$

**Boundary modifications** As mentioned in [Obe08] it is necessary to make modifications near the boundary as not all points in the stencil are available. This is solved by modifying the scheme to use an intermediate point at the boundary whose value is determined using quadratic interpolation along the boundary. As an example, suppose we need to calculate $D[i, M_y - 1, 1, 2]$, then the value in $j + b = M_y + 1$ is not available. Then we compute

$$D[i, M_y - 1; 1, 2] = \frac{1}{\frac{3}{4}(1^2 + 2^2)h^2} \left( 2u_{i+1/2,j+1} + u_{i-1,j-2} - 3u_{i,j} \right), \tag{5.21}$$

where $u_{i+1/2,j+1}$ is calculated from the interpolation

$$u_{i+1/2,j+1} = -\frac{1}{8}u_{i-1,j+1} + \frac{6}{8}u_{i,j+1} + \frac{3}{8}u_{i+1,j+1}. \tag{5.22}$$

### The 33-point stencil

This stencil incorporates also the directions of the 17-point stencil. Therefore only the extra directions have be to specified, which are $(3,1),(3,2),(2,3)$ and $(1,3)$. Making in total the set $\mathcal{G} = \{(1,0),(1,1),(2,1),(1,2),(3,1),(3,2),(2,3),(1,3)\}$, and MA-operator becomes

$$MA[u_{i,j}] = \min_{(a,b)\in\mathcal{G}} \left\{ D^{+}[i,j;a,b]D^{+}[i,j;-b,a] + D^{-}[i,j;a,b]D^{-}[i,j;-b,a] \right\}. \tag{5.23}$$

This version is *not implemented* by us due to the complications at the boundary which are more difficult than the 17-point stencil.

## 5.5 Accurate discretisation

For the accurate scheme a standard centred finite difference approach is used [Fro12]. The Monge-Ampère equation takes the form

$$u_{x_1x_1}u_{x_2x_2} - u_{x_1x_2}^2 = f(x_1,x_2)/g(u_{x_1},u_{x_2}). \tag{5.24}$$

The finite difference discretisation operator is given by

$$MA^{A}[u] = (\mathcal{D}_{x_1x_1}u)(\mathcal{D}_{x_2x_2}u) - (\mathcal{D}_{x_1x_2}u)^2 - f(x_1,x_2)/g(\mathcal{D}_{x_1}u,\mathcal{D}_{x_2}u), \tag{5.25}$$

with the finite difference operators defined as

$$\begin{aligned}
[\mathcal{D}_{x_1x_1}u]_{i,j} &= \frac{1}{h^2}(u_{i+1,j}+u_{i-1,j}-2u_{i,j}),\\
[\mathcal{D}_{x_2x_2}u]_{i,j} &= \frac{1}{h^2}(u_{i,j+1}+u_{i,j-1}-2u_{i,j}),\\
[\mathcal{D}_{x_1x_2}u]_{i,j} &= \frac{1}{4h^2}(u_{i+1,j+1}+u_{i-1,j-1}-u_{i-1,j+1}-u_{i+1,j-1}),\\
[\mathcal{D}_{x_1}u]_{i,j} &= \frac{1}{2h}(u_{i+1,j}-u_{i-1,j}),\\
[\mathcal{D}_{x_2}u]_{i,j} &= \frac{1}{2h}(u_{i,j+1}-u_{i,j-1}).
\end{aligned}$$

## 5.6 Boundary condition

The interior is now discretised. The transport boundary condition is given by

$$\nabla u(X) = Y. \tag{5.26}$$

It is claimed by Froese [Fro12], on unclear arguments refering to others, that it is sufficient to demand that the source boundary is mapped to the target boundary

$$\nabla u(\partial X) = \partial Y. \tag{5.27}$$

This BC is implemented by creating a nonlinear equation for each grid point on $\partial X$ that is local, and can still enforce the correct behaviour. The idea behind the approach of [BFO12c, Section 2] is to define a *convex* function $H$ with the property that $H(y) = 0 \Leftrightarrow y \in \partial Y$. This results in the the following nonlinear local boundary condition

$$H(\nabla u(\mathbf{x})) = 0 \qquad \text{for } \mathbf{x} \in \partial X. \tag{5.28}$$

In the paper they suggest the use of the signed distance function, i.e.,

$$H(\mathbf{y}) = \begin{cases} +\operatorname{dist}(\mathbf{y}, \partial Y) & \mathbf{y} \notin Y, \\ -\operatorname{dist}(\mathbf{y}, \partial Y) & \mathbf{y} \in Y, \end{cases} \tag{5.29}$$

where $\operatorname{dist}(\mathbf{y}, \partial Y) = \min_{\mathbf{y}_0 \in \partial Y} \|\mathbf{y} - \mathbf{y}_0\|_2$.

The next step is to discretise this boundary condition. To this purpose Benamou et al. [BFO12c] rewrite the function $H$ using the *Supporting Hyperplane Theorem* [Wik12d], which states that for each point on the boundary of a convex set there exists a supporting hyperplane. Using $\mathbf{n}$ to denote the outward unit normal to $\partial Y$, let $\mathbf{n}(\mathbf{y}_0)$ denote the normal at point $\mathbf{y}_0$ and let $\mathbf{y}(\mathbf{n})$ denote a point on $\partial Y$ with normal $\mathbf{n}$ then the following statements can be made.

$$\begin{aligned}
H(\mathbf{y}) &= \max_{\mathbf{y}_0 \in \partial Y} \left\{\mathbf{n}(\mathbf{y}_0) \cdot (\mathbf{y} - \mathbf{y}_0)\right\},\\
&= \max_{\|\mathbf{n}\|=1} \left\{\mathbf{n} \cdot (\mathbf{y} - \mathbf{y}_0(\mathbf{n}))\right\}\\
&= \max_{\|\mathbf{n}\|=1} \left\{\mathbf{y} \cdot \mathbf{n} - H^*(\mathbf{n})\right\}
\end{aligned}$$

where $H^*$ is an auxiliary function defined as

$$H^*(\mathbf{n}) = \max_{\mathbf{y}_0 \in \partial Y} \{\mathbf{y}_0 \cdot \mathbf{n}\}.$$

The first step is going from taking the normal at each boundary point, to looking at all possible normals, and the corresponding point on the boundary. For a detailed explanation and derivation of the discretisation, see Section 2 of [BFO12c]. In the discrete case we have to represent our boundary with a discrete set of normal vectors. First we need an important lemma.

**Lemma 5.6.1.** *If $X$ is a convex domain, mapped by $\nabla u$ to the convex set $Y = \nabla u(X)$, then for a point $\mathbf{x} \in \partial X$ with image $\mathbf{y} = \nabla u(\mathbf{x}) \in \partial Y$, the normal vectors $\mathbf{n}(\mathbf{x})$ at $\mathbf{x}$ and $\mathbf{n}(\mathbf{y})$ at $\mathbf{y}$ make an acute angle, i.e.,*

$$\mathbf{n}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{y}) \geq 0$$

This means we can restrict ourselves in the evaluation of $H$ to normal vectors making an acute angle with the unit outward normal at $\mathbf{x}$. (At a corner the intersection of the allowed sets for the various normals has to be used.) As an example Benamou et al. give the directions at the left boundary of a square domain. That means the normal $\mathbf{n}(\mathbf{x}) = (-1, 0)$. So the allowed directions are

$$\Gamma = \{\mathbf{n} = (n_1, n_2 | n_1 < 0, \|\mathbf{n}\| = 1\}.$$

We can discretise this at any point using an upwind scheme

$$\begin{aligned} H(\nabla u(\mathbf{x}_{i,j})) = \;& \max_{(n_1,n_2)\in\Gamma}\{\nabla u(\mathbf{x}_i) \cdot \mathbf{n} - H^*(\mathbf{n})\} \\ \approx \;& \max_{(n_1,n_2)\in\Gamma}\Big\{\max\{n_1, 0\}\tfrac{u_{i,j}-u_{i-1,j}}{h} + \min\{n_1, 0\}\tfrac{u_{i+1,j}-u_{i,j}}{h} \\ & + \max\{n_2, 0\}\tfrac{u_{i,j}-u_{i,j-1}}{h} + \min\{n_2, 0\}\tfrac{u_{i,j+1}-u_{i,j}}{h} \\ & - H^*(n_1, n_2)\} \end{aligned} \tag{5.30}$$

For extra accuracy we also implemented a 3-point upwind difference scheme, instead of the 2-point one. This helps to reduce the error near the boundaries for smooth problems. In Section 6.1, examples show the effect on the accuracy. It is important to realise that this method uses the implicit assumption that the target set $Y$ is convex, but not necessarily rectangular. Otherwise the identification between normal vectors and boundary points does not work.

## 5.7 Implementation details

For this numerical approach to be complete there are some important details left to discuss. First the Jacobi matrix as it is fundamental to the Newton method. Second, the initialisation and stopping criteria of Newton's method are described. Finally this chapter is concluded with a discussion on the interpolation and extrapolation method used during the numerical calculations.

We also note that we implemented this algorithm first in MATLAB. Later we ported the calculation of the residuals and the Jacobi matrices to C++ running as MEX-files in MATLAB. This means we have a program written partly in MATLAB and partly in C++.

### 5.7.1 Jacobi matrix

In the previous section only the discretisation of the MA equation itself was discussed. For Newton's method also the Jacobi matrix must be calculated. For the Jacobi matrix it is pointed out in [FO12] that it can be written as a combination of the Jacobi matrix for the stable method $\nabla N_M$ and the accurate scheme $\nabla N_A$ as

$$\nabla \mathbf{N}_{k,l} = \left(1 - S'\left(\frac{\mathbf{N}_{A,k} - \mathbf{N}_{M,k}}{\epsilon(h, d\theta)}\right)\right) \nabla \mathbf{N}_{M,k,l} + S'\left(\frac{\mathbf{N}_{A,k} - \mathbf{N}_{M,k}}{\epsilon(h, d\theta)}\right) \nabla \mathbf{N}_{A,k,l} \tag{5.31}$$

where for the filter from (5.7) the derivative is given by

$$S'(x) = \begin{cases} 1 & |x| < 1 \\ -1 & 1 < |x| < 2 \\ 0 & |x| > 2. \end{cases} \tag{5.32}$$

As we do not want the coefficients before the subscheme Jacobi matrices to become negative, we approximate this with

$$\tilde{\nabla} \mathbf{N}[u] = \left(1 - S'\left(\frac{\mathbf{N}_A - \mathbf{N}_M}{\epsilon(h, d\theta)}\right)\right) \nabla \mathbf{N}_M + \max\left\{S'\left(\frac{\mathbf{N}_A - \mathbf{N}_M}{\epsilon(h, d\theta)}\right), 0\right\} \nabla \mathbf{N}_A \tag{5.33}$$

The elements of the Jacobi matrices of the various schemes can be found in Appendix B.

### 5.7.2 Initialisation and stopping criteria

For Newton's method it is very important to start with a good initial guess, otherwise convergence is not guaranteed. Unfortunately Benamou et al. [BFO12c] do not explain what they use for this algorithm. From their earlier papers, it can be extracted that they propose to use a numerical solution of

$$\Delta u_0(\mathbf{x}) = f(\mathbf{x})/g(\mathbf{x} - \mathbf{x}_0), \tag{5.34}$$

where $\mathbf{x}_0$ is a reference point in $Y$. The key question then is which boundary conditions to use. This suggestion was made in the context of the MA-equation with Dirichlet boundary conditions. For a simple rectangular target this could be implemented with Neumann boundary conditions as they are equivalent to the transport boundary condition. By mapping the edges of the source to the corresponding edges on the target the boundary condition is satisfied. For example take $X = Y = [0, 1]^2$. Then sending the right edge at $x_1 = 1$ to the right edge of the target at $y_1 = 1$ means that we need $\frac{\partial u}{\partial x_1}|_{x_1=1} = 1$.

However, this method is relatively expensive, and it does not work for non-rectangular geometries. In general the following observations were made:

1. $u_0$ must be a continuous convex function on $X$.
2. For a general rectangular source domain mapped to a rectangular target, a simple linear mapping can be specified.

This was used to define a quadratic function on $X$ whose gradient maps $X$ to a rectangular bounding box which fully includes $Y$. It turns out that this works very well for almost all examples tested. Suppose we write $X = [a_1, b_1] \times [a_2, b_2]$ and $Y \subset [c_1, d_1] \times [c_2, d_2]$. Then the initial guess is written as

$$\frac{d_1 - c_1}{b_1 - a_1}\frac{1}{2}x^2 + \frac{b_1c_1 - a_1d_1}{b_1 - a_1}x + \frac{d_2 - c_2}{b_2 - a_2}\frac{1}{2}y^2 + \frac{b_2c_2 - a_2d_2}{b_2 - a_2}y - u_r. \tag{5.35}$$

Note that for example $\nabla u(a_1, a_2) = (c_1, c_2)$.

Another important aspect of Newton iteration is when to stop. There are two important quantities involved. The first is the residual vector $\mathbf{N}(u)$, the second is the update vector $\mathbf{w}$. If they are both small, it is reasonable to conclude we are close to the solution, or at least as close as the iterations can bring us. In practice the algorithm continues until all of the following are true

$$\frac{\beta}{h}\frac{\|\mathbf{w}\|_1}{\|u\|_1} < tol_1, \tag{5.36a}$$

$$\frac{\beta}{M_xM_y}\|\mathbf{w}\|_1 < tol_2, \tag{5.36b}$$

$$\frac{\|\mathbf{N}\|_1}{M_xM_y} < tol_3. \tag{5.36c}$$

for some user chosen tolerances $tol_1, tol_2$ and $tol_3$ and $\beta$ is the damping parameter of the Newton iteration. Typical values for these threshold are $tol_1 = 10^{-4}$ and $tol_2 = tol_3 = 10^{-5}$.

### 5.7.3 Interpolation and extrapolation

During the numerical solution procedure we need to be able to evaluate the target density $g(y_1, y_2)$ as part of the right hand side of (5.1) for any point $\mathbf{y} = (y_1, y_2) \in \mathbb{R}^2$, this means not only restricted to the target set $Y$, but sometimes outside it. This is due to the fact that the iterant is not a perfect mapping, and may send some parts of the source outside the target area, before it is restricted by the boundary condition. Moreover, even inside $Y$ we can not restrict ourselves to grid point as the image under the mapping of the source grid, can be any point. In practice $g$ is often specified by the user as a table of values on grid points. These observations lead us to the need to use an interpolation routine for the evaluation of $g$, and sometimes also an extrapolation method for extending the functions.

#### Interpolation

For determining an interpolation algorithm the following was taken into consideration

- The function $g$ is continuous, but its derivatives need not be continuous. This is problem dependent so in general we can only assume Lipschitz continuity.
- This means any higher order and smooth interpolation algorithm is unsuited.
- It must be computationally cheap as there are many evaluations during the calculation of the algorithm.

Therefore we decided that bilinear interpolation is the most suitable way to accomplish this. The implementation is based on the implementation in [PTVF07].

### Extrapolation

For the extension of the density outside the domain, the situation is more complex. According to [BFO12c] we need to find a Lipschitz continuous and positive extension of the function $g$. They refer to [Obe05] for details. However the procedure mentioned in [Obe05] is very complex and computationally intensive. Therefore an alternative is needed. We have the following requirements

- The function must be extended in a continuous way.
- The derivative should stay bounded, i.e. the Lipschitz constant should not increase.
- It should allow fast evaluation.

The method chosen is to extend the function by the average value of $g$ on the grid, when we are 4 times $r_Y$ from the boundary, where $r_Y$ is the maximum distance between nay two points in $Y$. In the space between the edge of the grid, and the domain with average value, a simple bilinear interpolation between this average and the value in the closest point on the grid is used. This way the function is extended continuously and after some distance will obtain a uniform positive value. This method guarantees the extension is strictly positive as $g$ is strictly positive.

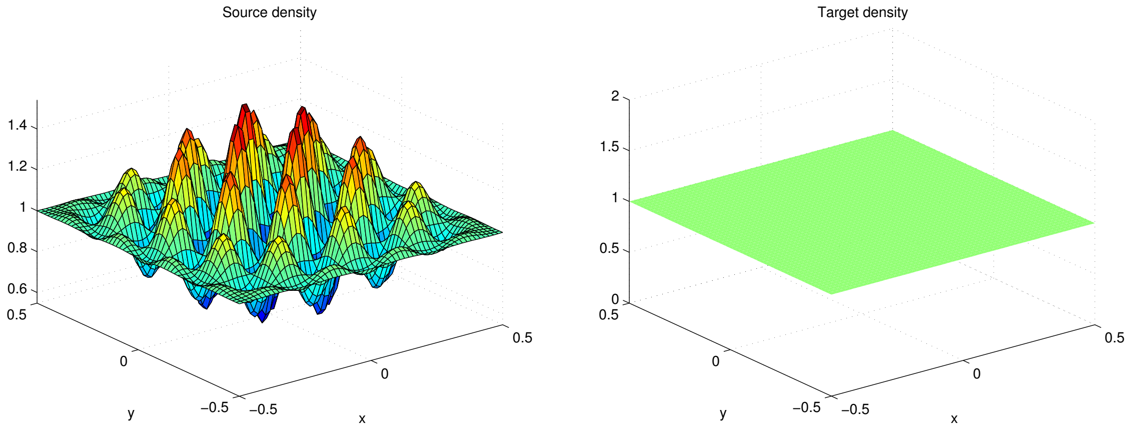
# Chapter 6

# Numerical Examples

In this chapter the numerical results for some selected problems are shown and discussed. The examples are chosen such that the different aspects of the code can be shown. Before discussing the results the default parameters are described. All examples in this chapter are ran using the following parameter set, unless specified otherwise. For definitions see Appendix C.1.

$$
\begin{aligned}
M_x &= 257 & M_y &= 257 & N_y &= 32\\
\text{maxit} &= 40 & \text{rtol} &= 10^{-4} & \text{atol} &= 10^{-5}\\
\text{stencil} &= 4 & \text{delta} &= 10^{-6} & \text{H\_scheme} &= 1\\
\text{debug\_mode} &= \text{false} & \text{min\_damping} &= 1 & \text{init\_damping} &= 1\\
\text{anchor\_weight} &= 1
\end{aligned}
$$

## 6.1 Smooth example



(a) source

(b) target

Figure 6.1: The source (a) and target (b) densities from (6.2)

The first test is an example from Benamou et al.[BFO12c, Section 5.3.1.] for a problem on a square source domain and a corresponding square target. First define the function

$$q(z) = \left(-\frac{1}{8\pi}z^2 + \frac{1}{256\pi^3} + \frac{1}{32\pi}\right)\cos(8\pi z) + \frac{1}{32\pi^2} z\sin(8\pi z). \tag{6.1}$$

The problem is then given by

$$\begin{aligned} X =&[-\tfrac{1}{2},\tfrac{1}{2}]^2, & f(x_1,x_2) =&1 + 4\left(q''(x_1)q(x_2) + q(x_1)q''(x_2)\right) & (6.2)\\ & & &+ 16\left(q(x_1)q(x_2)q''(x_1)q''(x_2) - q'(x1)^2q'(x_2)^2\right), & \\ Y =&[-\tfrac{1}{2},\tfrac{1}{2}]^2, & g(y_1,y_2) =&1, & (6.3)\end{aligned}$$

and the densities are plotted in Figure 6.1. Benamou also gives the analytical solution as

$$(x_1,x_2) \mapsto (x_1 + 4q'(x_1)q(x_2), x_2 + 4q(x_1)q'(x_2)). \tag{6.4}$$

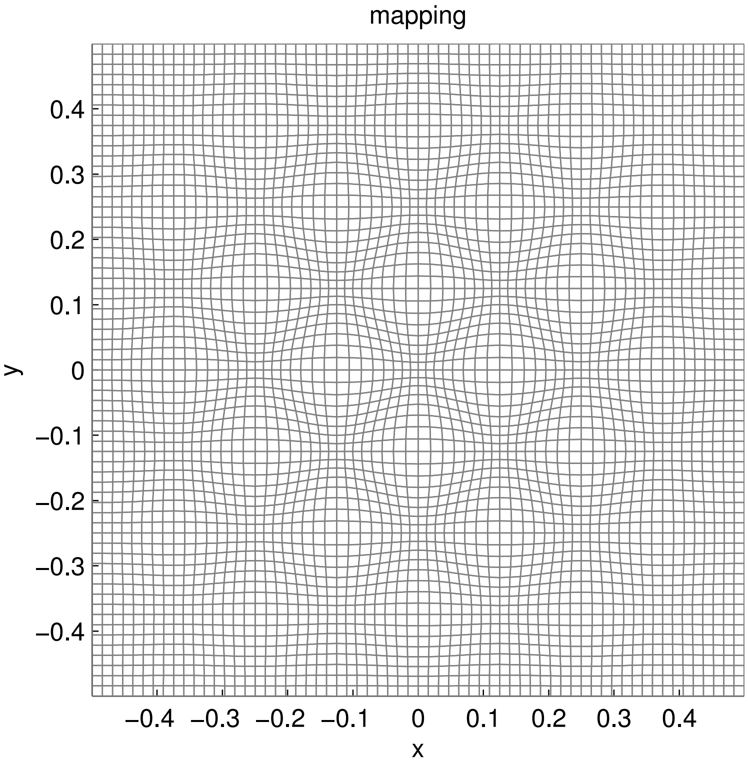
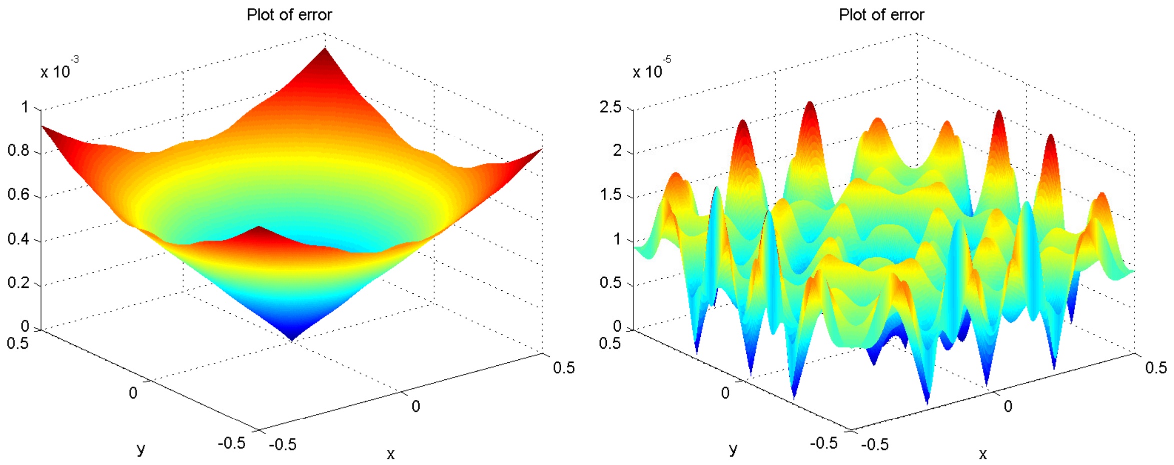


Figure 6.2: The image of a regular grid on the source under the numerical solution of problem (6.2).

The mapping created from the numerical solution is plotted in Figure 6.2, and the corresponding error in Figure 6.3. The algorithm ran twice. The first time with only the two-point scheme on the boundary, the second time with a three-point scheme. As can been seen in Figure 6.3 the error reduces dramatically by two orders of magnitude. The run time for the two-point scheme was 9.89 seconds resulting in an overall error in the 1-norm of $6.73 \cdot 10^{-4}$, while the three-point scheme was 11.18 seconds with an error of $2.04 \cdot 10^{-5}$. In all cases the number of iterations is 7.



(a) H_scheme=0

(b) H_scheme=1

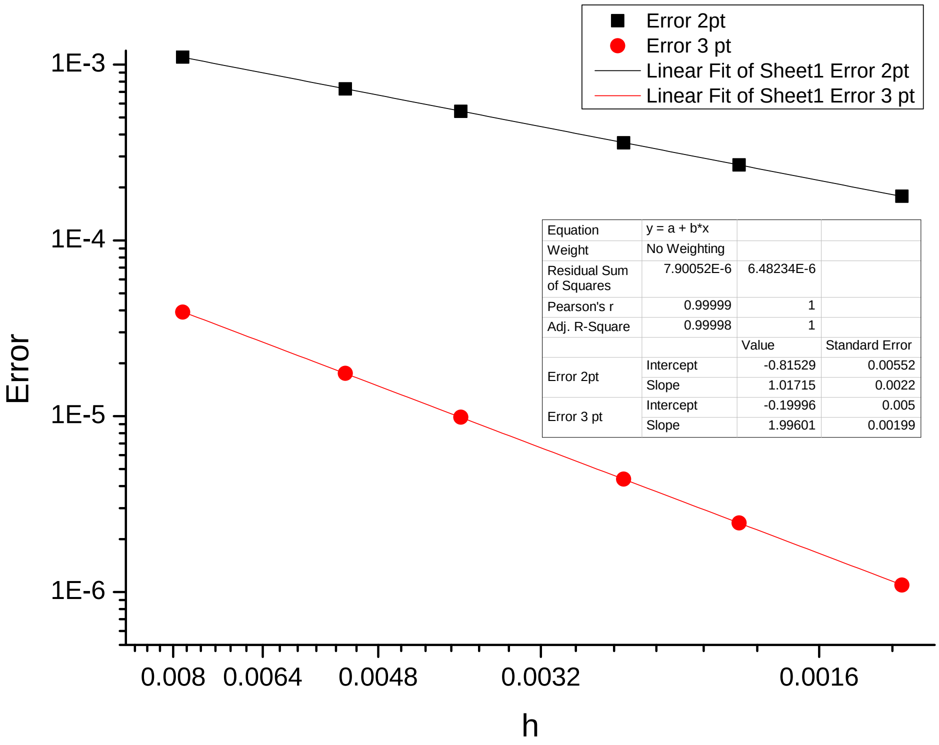
Figure 6.3: The pointwise error for the two-point (a) and three-point (b) schemes for the boundary discretisation

For both cases also convergence test have been conducted. This means the algorithm was applied for different grid sizes ($2^7+1$,$2^8+1$,$2^9+1$ and some in between to have more data for fitting) and the errors and run times can be found in Table 6.1. From this table we see that the three-point scheme gives us quadratic convergences while the two-point scheme is only linear. Furthermore the runtime scales like $M^{1.2}$ respectively $M^{1.24}$ where $M$ is the total amount of points ($M = M_x \cdot M_y$).

Table 6.1: Distance between exact solutions and numerical solutions for the mapping from (6.2) in the 1-norm, including also computation times, see also Figure 6.4

| $M_x$ | Runtime 2pt (s) | Error 2pt | Runtime 3pt (s) | Error 3pt |
|---|---:|---|---:|---|
| 129 | 3.1 | $1.1 \cdot 10^{-3}$ | 2.3 | $3.9 \cdot 10^{-5}$ |
| 193 | 6.1 | $7.3 \cdot 10^{-4}$ | 6.3 | $1. \cdot 10^{-5}$ |
| 257 | 11 | $5.4 \cdot 10^{-4}$ | 11 | $9.9 \cdot 10^{-6}$ |
| 385 | 31 | $3.6 \cdot 10^{-4}$ | 34 | $4.4 \cdot 10^{-6}$ |
| 513 | 55 | $2.7 \cdot 10^{-4}$ | 67 | $2.5 \cdot 10^{-6}$ |
| 769 | 150 | $1.8 \cdot 10^{-4}$ | 190 | $1.1 \cdot 10^{-6}$ |

These results are very interesting as the paper by Benamou et al. [BFO12c] claims that for this problem there should be quadratic convergence even with the simple scheme on the boundary. The fact that we do not observe this might point to an implementation error. However, as the three-point scheme does give quadratic convergence this is strange. Surprisingly the run-times grow only very modestly and are in line with the results from Benamou et al. who calculated $M^{1.3}$. The biggest practical restriction using large grids is not computing times but memory usages. On my test machine (a laptop with a Dual core 2.53 GHz processor and 4 GB of memory running Windows 7, MATLAB R2011a) the grid size of $M_x = M_y = 769$ is about the largest that can be done. The most memory usage is observed



| | | | |
|---|---|---|---|
| Equation | y = a + b*x | | |
| Weight | No Weighting | | |
| Residual Sum of Squares | 7.90052E-6 | 6.48234E-6 | |
| Pearson's r | 0.99999 | 1 | |
| Adj. R-Square | 0.99998 | 1 | |
| | | Value | Standard Error |
| Error 2pt | Intercept | -0.81529 | 0.00552 |
| | Slope | 1.01715 | 0.0022 |
| Error 3 pt | Intercept | -0.19996 | 0.005 |
| | Slope | 1.99601 | 0.00199 |

Figure 6.4: Plot of convergence results from Table 6.1

when MATLAB is solving the linear system.

## 6.2 Examples exploring geometries

In this section several examples using different geometries are shown.

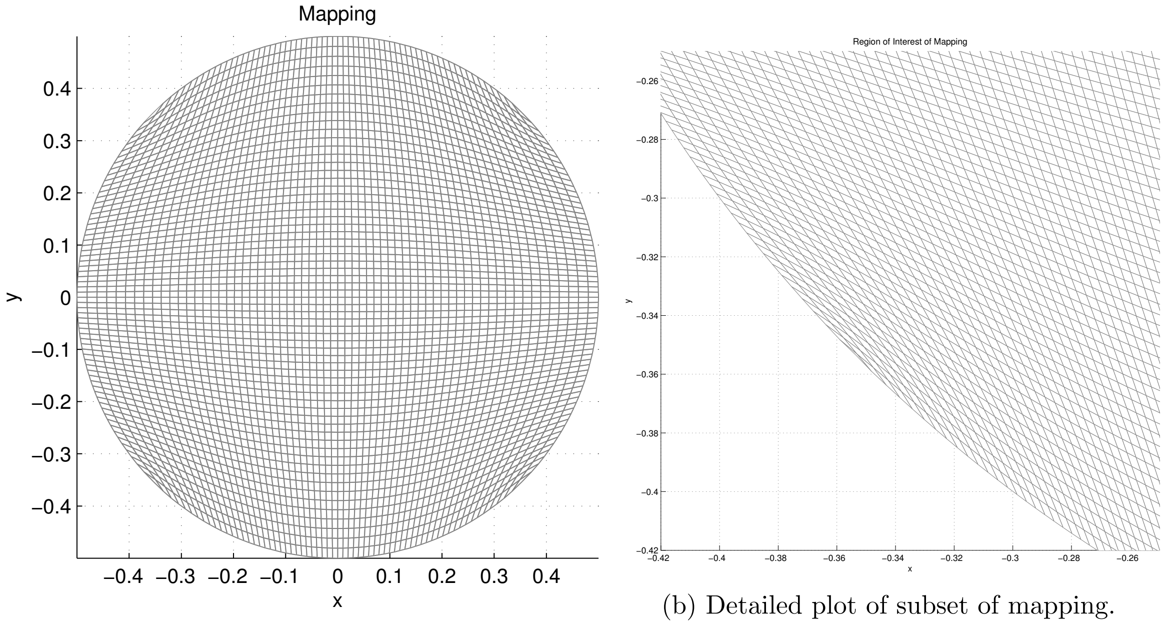
### 6.2.1 Square to circle

A simple test case for the boundary condition is to see how it handles the transport of a uniform density on a square to a uniform density on a circle. The problem is defined as

$$X = \left[-\tfrac{1}{2}, \tfrac{1}{2}\right]^2, \qquad f(x_1, x_2) = \frac{\pi}{4}, \tag{6.5}$$

$$Y = \left\{(y_1, y_2) \in \mathbb{R}^2 | y_1^2 + y_2^2 < \left(\tfrac{1}{2}\right)^2\right\}, \qquad g(y_1, y_2) = 1. \tag{6.6}$$

Calculating the numerical solution for such a geometry we obviously require more normals on the boundary compared to the square of the previous example, therefore $N_y$ is set to 512. This results in the mapping shown in Figure 6.5. The calculation is relatively expensive with 63 seconds (compare with 11 sec, for the previous example on the same grid size), using 38 iterations (compared to 11). The most interesting is of course the approximation of the circular target domain. As can be seen in Figure 6.5b the approximation is good. The reverse problem, with the source and target swapped can also be solved, as shown in Figure 6.6. The mapping computed from the inverse problem, can itself be inverted, see Section A, and



(a) Full solution $\nabla u(X)$

(b) Detailed plot of subset of mapping.

Figure 6.5: The image of a regular grid after transforming to a circle (6.5)

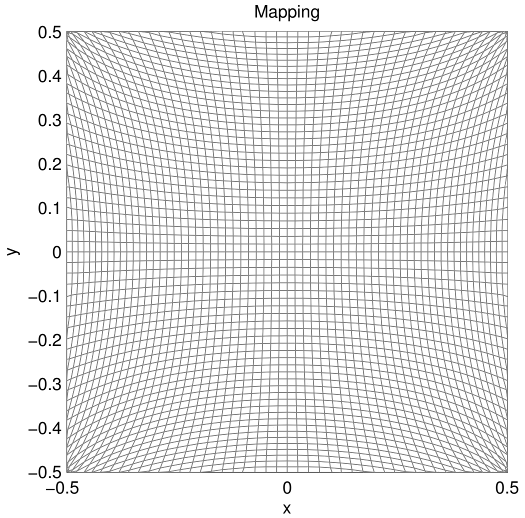
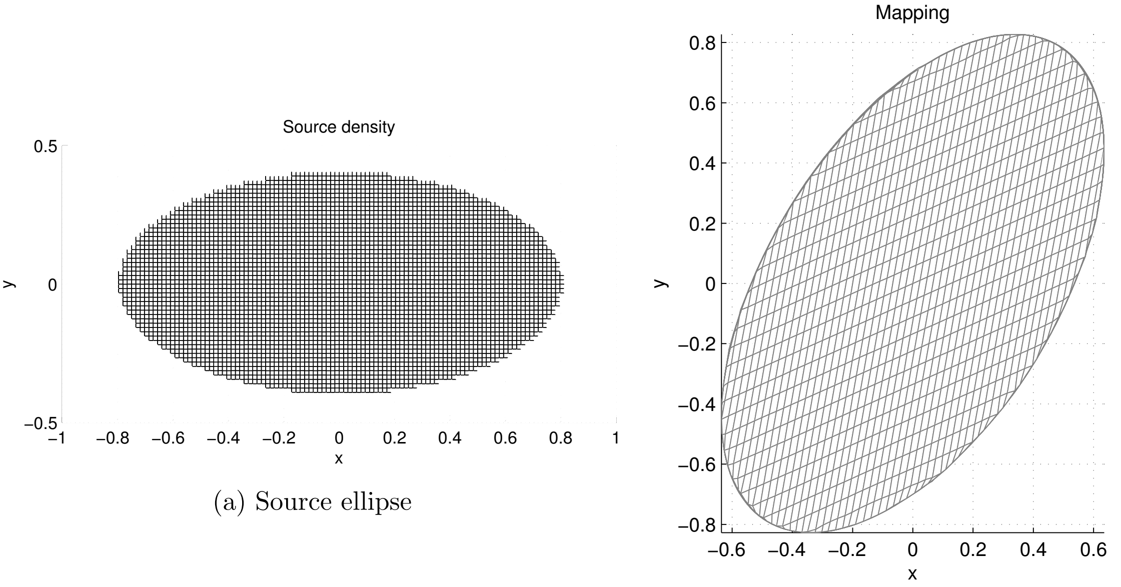


Figure 6.6: The image of a rectangular grid in a circle after transforming to a square (6.2).

compared to the direct mapping. However, this problem is much harder as the new source domain is a circle which is extended with zeros to a square. Although it works reasonably well it does introduce some inaccuracies. The comparison between direct problem and the reversed gives in most points an difference of $O(h)$. This suggest that the results from directly calculating and trough the inverse mapping are consistent with each other.

### 6.2.2 Ellipse to ellipse



(a) Source ellipse

(b) Numerical image of the source ellipse

Figure 6.7: The image of a regular grid on the source under the numerical map calculated for 6.2.2

In Figure 6.7 you see the numerical solution for mapping an ellipse with uniform density to a different ellipse with the same uniform density . This example is taken from Benamou et al. [BFO12c, Section 5.3.2]. The ellipses are defined using matrices $\mathbf{A_x}$ and $\mathbf{A_y}$. Let $B$ be the unit ball in $\mathbb{R}^2$ then $X = \mathbf{A_x}B, Y = \mathbf{A_y}B$, where

$$\mathbf{A_x} = \begin{pmatrix} 0.8 & 0 \\ 0 & 0.4 \end{pmatrix}, \mathbf{A_y} = \begin{pmatrix} 0.6 & 0.2 \\ 0.2 & 0.8 \end{pmatrix}.$$

Comparing the numerical solution with the analytical solution provided by Benamou et al. for different number of grid points shows that the error is of order $h$ which agrees with their findings. Note that the need to extend the source with zeros, introduces a singularity to this problem that makes it hard to find the right values of the solver parameters needed for the best results.

### 6.2.3 Nonconvex source domain

In this example a nonconvex source domain is used to put the algorithm to the test. We take a disk with a hole in the middle with uniform density as a source, and a disk with uniform

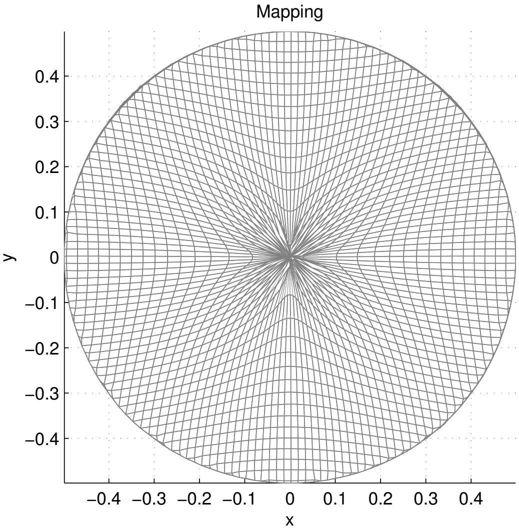


Figure 6.8: The image of a regular grid on a disk after mapping with the numerical solution of (6.7) to a square.

density as a target, i.e.,

$$X = \{(x_1, x_2) \in \mathbb{R}^2 | 0.25^2 < x_1^2 + y_1^2 < 0.5^2\}, \qquad f(x_1, x_2) = 1 \tag{6.7}$$

$$Y = \left\{(y_1, y_2) \in \mathbb{R}^2 | y_1^2 + y_2^2 < 0.5^2\right\}, \qquad g(y_1, y_2) = 0.75. \tag{6.8}$$

This problem does not satisfy the theoretical requirements, the others do, for the algorithm to work. The source domain is nonconvex and also not even simply connected. For this example a grid size of 192 by 192 points is used. It turns out that the solution is very sensitive to changes in many parameters. The result is shown in Figure 6.8. One should avoid these kinds of geometries as the results are unpredictable. The good news from this example is that these kind of singular problems do not cause the algorithm to break down. The bad news is that the results are clearly not very accurate as the hole is not completely filled.

## 6.3 Challenging smooth example

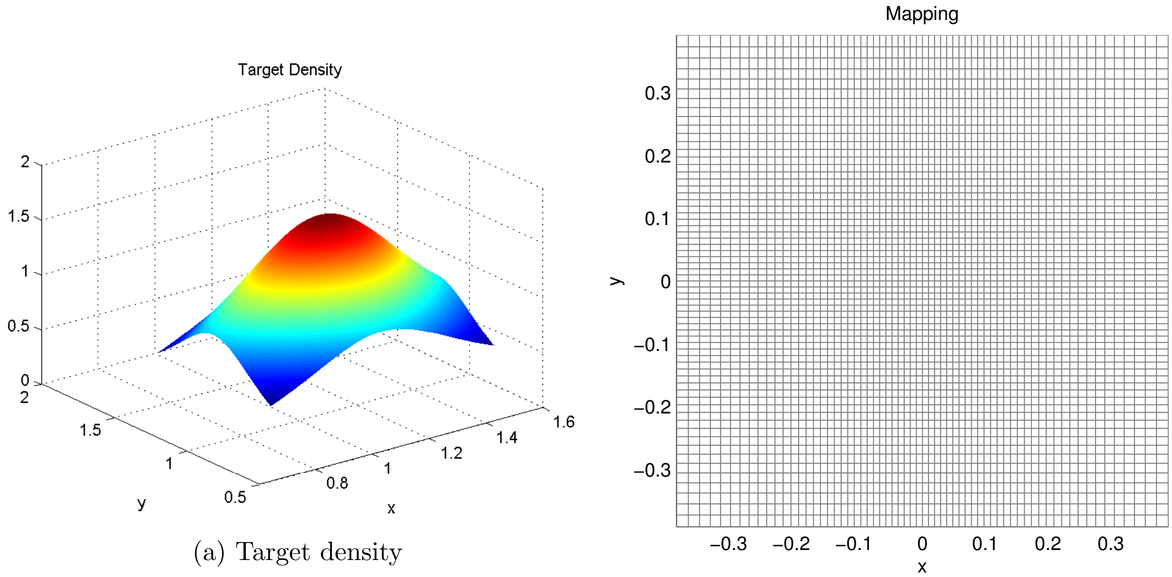
From Maes [Mae97, Section 3.3.3., Example 3.3.3] an example is used that is known to stress existing algorithms in the symmetric case. The example was extended to the full domain by rotation around the origin. The problem is described by:

$$X = \left[\frac{\pi}{4}, \frac{\pi}{2}\right]^2, \qquad f(x_1, x_2) = 1, \tag{6.9}$$

$$Y = \left[-\frac{\pi}{8}, \frac{\pi}{8}\right]^2, \qquad g(y_1, y_2) = \left(\frac{4\pi}{\pi^2 + 64y_1^2}\right) \cdot \left(\frac{4\pi}{\pi^2 + 64y_2^2}\right). \tag{6.10}$$

The analytical solution is given by

$$\nabla u = \frac{\pi}{8}\left(\tan\left(2x_1 - \frac{3\pi}{4}\right), \tan\left(2x_2 - \frac{3\pi}{4}\right)\right). \tag{6.11}$$



(a) Target density

(b) Numerical mapping

Figure 6.9: The image of a regular grid on the source under the numerical solution of (6.9)

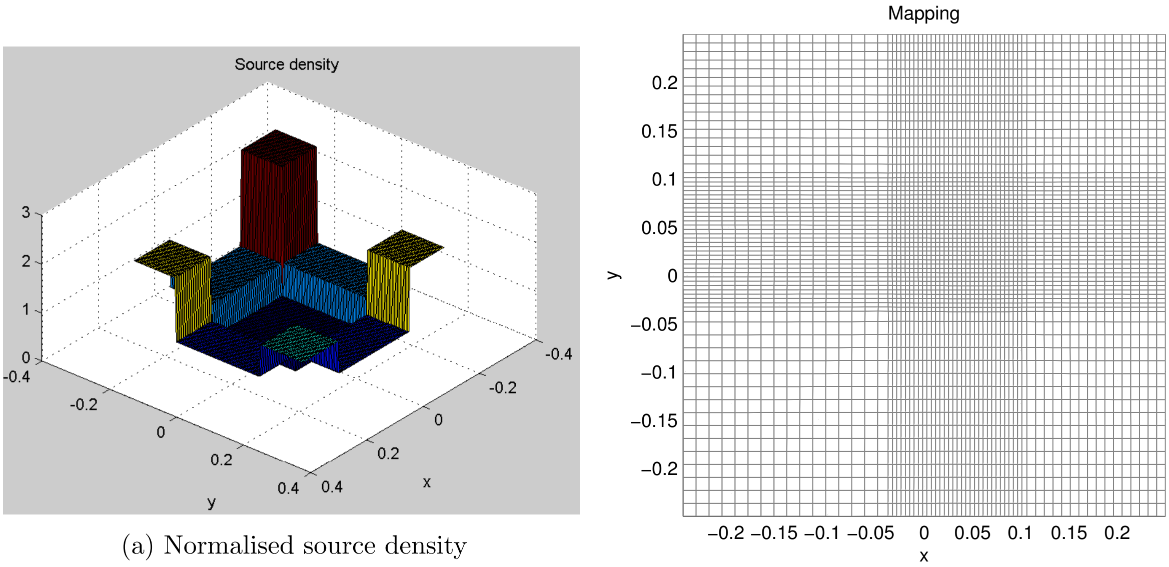
The numerical solution is shown in Figure 6.9, and convergence results are shown in Table 6.2. The results are very surprising. For both boundary schemes the total convergence is first order, although the three-point scheme gives and error of half the size as the two-point scheme. The reason for his lower order behaviour is unclear as the functions involved are all smooth and the geometry is easy. Obviously it might be that there is a mistake in the computer code.

Table 6.2: Distance in 1-norm, divided by the number of points, between exact solutions and numerical solutions for the mapping from (6.9), including also computational time

| $M_x$ | Runtime 2pt (s) | Error 2pt | Runtime 3pt (s) | Error 3pt |
|---|---:|---|---:|---|
| 129 | 11 | $3.5 \cdot 10^{-3}$ | 11 | $1.8 \cdot 10^{-3}$ |
| 193 | 20 | $2.3 \cdot 10^{-3}$ | 18 | $1.2 \cdot 10^{-3}$ |
| 257 | 38 | $1.7 \cdot 10^{-3}$ | 42 | $8.7 \cdot 10^{-4}$ |
| 385 | 90 | $1.2 \cdot 10^{-3}$ | 140 | $5.8 \cdot 10^{-4}$ |
| 513 | 220 | $8.6 \cdot 10^{-4}$ | 430 | $4.3 \cdot 10^{-4}$ |

## 6.4 Example with a discontinuous source density.

In this example based on Maes [Mae97, example 3.3.4] we use a piecewise continuous source density and a uniform target density. The result is given in Figure 6.10. The problem is specified as a piecewise constant function in the first coordinate multiplied with the same



(a) Normalised source density

(b) Numerical mapping

Figure 6.10: The image of a regular grid on the source under the numerical solution of (6.12)

function in the second coordinate:

$$X =[-\frac{1}{4},\frac{1}{4}]^2, \qquad f(x_1,x_2) = \left(\begin{cases} 6 & x_1 < -\frac{1}{8} \\ 2 & -\frac{1}{8} < x_1 < \frac{1}{8} \\ 4 & x_1 > \frac{1}{8} \end{cases}\right) \cdot \left(\begin{cases} 6 & x_2 < -\frac{1}{8} \\ 2 & -\frac{1}{8} < x_2 < \frac{1}{8} \\ 4 & x_2 > \frac{1}{8} \end{cases}\right) \tag{6.12}$$

$$Y =[-\frac{1}{4},\frac{1}{4}]^2, \qquad g(y_1,y_2) =1. \tag{6.13}$$

In Figure 6.10 one can nicely see how the grid cells are compressed or expanded independently along the two directions. The abrupt change in source density is captured accurately as an abrupt change in the density of the grid lines of the mapping. Plotting the solution after each iterations allowed us to observe that around the discontinuities artifacts arose. These artifacts are straightened out quickly by the algorithm and disappear. This problem took 18 iterations in 31 seconds to compute the solutions shown.

## 6.5 Program profiling

It is often insightful to know how much computing time the different parts of the program cost. To generate good estimates it is important to take a big task, to eliminate any initialisation or plotting from the estimates. we took the calculation of the second lens of Chapter 7 on a grid of 768 by 768 points. The boundary was defined using 3072 points and also 3072 normal vectors. This problem ran for 89 iterations before stopping. In Table 6.3 a list of most time consuming functions is given and their run times. From this we can see that more than 80 % of the computation time is spend in builtin routines of MATLAB, which are highly optimized. Therefor any speed up in the computation must come from changing the algorithm, as the implementation has not much room for improvement.

Table 6.3: Table containing run times of the most computationally costly routines

| Task | Code type | Run time (s) | Relative runtime (%) |
|---|---|---:|---:|
| Precalculating $H*$ | MATLAB | 42.6 | 2.5 |
| Calculating Residual $\mathbf{N}$ | C++ | 70.5 | 4.2 |
| Calculating elements of Jacobi matrix | C++ | 102.2 | 6.1 |
| Creating sparse Jacobi matrix | MATLAB (built-in) | 210.9 | 12.6 |
| Solving linear system | MATLAB (built-in) | 1196.7 | 71.6 |
| Sum of above tasks | Mixed | 1623.1 | 98.5 |
| Total calculation | Mixed | 1671.6 | 100 |

Using the windows task manager we can see that both the cpu-usage and memory usage follow a periodic pattern coinciding with the iterations. For this extreme example the memory usage of MATLAB averages around 2.5 GB and peaks at around 3.5 GB.

# Chapter 7

# Parallel Beam Lens

To test the theory from Section 4.2 and see how the program performs for real problems we are going to design a lens. The setting is that of an incoming uniform parallel light beam, with intensity $f(\mathbf{x}) = 1$ on a square domain $X = [-0.18, 0.18]^2$. The target is defined in spherical coordinates as

$$\hat{G}(\theta, \phi) = \begin{cases} 1 & \theta < \frac{5}{180}\pi \\ 0, & \theta > \frac{5}{180}\pi. \end{cases} \tag{7.1}$$

The first surface of the lens is flat, so it has no influence on the light beams. We now calculate the second surface, under the assumption that the lens material has an index of refraction of 1.49, and the surrounding medium is air. The calculated lens is shown in Figure 7.3.

The numerical results were transfered to the illumination simulation program LightTools. LightTools is a profession simulation package for illumination optics. Given a model of an optical system including sources, numerous simulations can be run. We use the Monte-Carlo ray trace functionality. This means that the program picks a random position on the source and send a ray from there. It calculates the reflections and / or refractions the ray undergoes while travelling trough the system. Finally it record the direction the ray leaves this system. Simulating a large amount, e.g. millions, of rays and collecting the simulated rays in bins on a grid we can estimate the intensity distribution in the far field. This process introduces errors due to the numerical approximations of the surfaces involved, and due to the statistical noise introduced by the random character of the ray tracing process. Using this software a Monte-Carlo ray trace was performed using $10^7$ rays. The resulting intensities are show in Figure 7.1, where the intensity is plotted on a so called $uv$-grid which is regular. The relation between this grid and the coordinates $\theta$ and $\phi$ is given by

$$\theta = \sqrt{u^2 + v^2} \tag{7.2a}$$

$$\phi = \arctan\left(\frac{v}{u}\right) + \pi. \tag{7.2b}$$

The intensity shown in the image is not equal to 1, as there is a scaling involved depending on the luminous flux produced by the source. Using the transform we can calculate the following point wise error estimate $e(\theta, \phi)$, where the ray trace results are denoted using $G_{\text{ray}}$,

$$e(\theta, \phi) = \frac{|G(\theta, \phi) - G_{\text{ray}}(\theta, \phi)|}{G(\theta, \phi)}. \tag{7.3}$$

The error is shown in Figure 7.2. These results are very good, as a beam of the expected shape and size is produced. However, the error is an order of magnitude smaller than the error expected from statistical deviations produced in the Monte-Carlo process. The most probable cause is that the error estimates produced by LightTools are upper bounds and for such simple intensity function the accuracy is higher.

As we can now design optics that make a beam, we would like to make a more interesting and challenging example. As the goal of this project is to look for asymmetric optics, we try something asymmetric. A nice target density representing a flower like structure with 7 leaves is given in

$$\hat{G}(\theta, \phi) = 2\exp\left(-\frac{\theta^2}{(\pi/20)^2}\right)(0.8 + 0.2\sin(30\theta))(1 + 0.05\sin(7\phi)) \tag{7.4}$$

on the set $\hat{Y} = [0, \frac{5}{180}\pi] \times [-\pi, \pi]$. This target density was transformed numerically using (4.21) and (4.22). The result is visualised in Figure 7.4. From the coordinate transformation of (4.21) we can derive that this density is defined on the disk with radius approximately (after rounding) of 0.18, so this is our $Y$.

The following Monge-Ampère problem was now formulated for the algorithm

$$\begin{aligned} X =& [-0.18, 0.18]^2, & f(x_1, x_2) =& 0.31 & (7.5)\\ Y =& \left\{(y_1, y_2) \in \mathbb{R}^2 \mid y_1^2 + y_2^2 < 0.18^2\right\}, & & & (7.6)\\ & & & & (7.7) \end{aligned}$$

where $g(y_1, y_2)$ is as shown in Figure 7.4, and the value of $f$ is chosen such that the integrals of $f$ and $g$ are equal. The solver was set using these parameters

$$\begin{aligned} M_x &= 257 & M_y &= 257 & N_y &= 513\\ \text{maxit} &= 200 & \text{rtol} &= 10^{-4} & \text{atol} &= 10^{-6}\\ \text{stencil} &= 4 & \text{delta} &= 10^{-6} & \text{H\_scheme} &= 1\\ \text{debug\_mode} &= \text{false} & \text{min\_damping} &= 0.25 & \text{init\_damping} &= 0.125\\ \text{anchor\_weight} &= 4. \end{aligned}$$

The resulting lens was simulated again in LightTools, a screenshot of the system is shown in Figure 7.8. The simulated intensity is shown in Figure 7.5, compare with the theoretical target in Figure 7.6. We can very nicely see the the features of leaves. The error we calculated is plotted on the same $uv$-grid as the solution in Figure 7.7. It is clear that the error is largest in the centre. This is due to the fact that the seven leaves come together there and they cannot be represented accurately on a square grid. However also in the remainder of the domain the error is of an order of 3 percent. There is also a strange circle at 2° for which we have no explanation. For a numerical artifact it would be more logical to be spots of corruption than a perfect circle.

In order to investigate the behaviour of the system the same experiment was repeated for grid sizes of 129, 257 and 513 respectively. The resulting optics were all traced with 10 million rays. A plot of the local errors is shown in Figure 7.9. Furthermore we have to take in mind that the errors we measure can have multiple sources.

- The first source for errors are the discretisation error and truncation error of the numerical solver. This is the error we would like to estimate here

- A second source is numerical errors made by the ray tracing application, due to the fact that the surface is represented by a finite amount of points
- Thirdly, the random behaviour of the ray tracing results in statistical fluctuations in the perceived intensity. This error scales with $\frac{1}{\sqrt{N_{ray}}}$ where $N_{ray}$ is the number of rays used.

As we cannot measure the individual errors, it is hard to attribute anything to the algorithm. This is further complicated by the fact we have no control over the second source, and suppressing the third requires a lot of computing time. The 2-norm of the relative errors for each of the 3 grid sizes was calculated and are shown in table 7.1. We see that error decreases for larger grids. This is consistent with the results as visualised in Figure 7.9. From these data we can calculate that the error goes like $M_x^{0.6}$, so sub linear. Which part of the system is responsible is for now unclear. A bigger and more intensive test procedure might be able to reveal more, but we did not have the time remaining for that.

Table 7.1: List of error estimates for three different grid sizes

| $M_x$ | 2-norm error |
|---|---|
| 129 | 2.714e-2 |
| 257 | 1.756e-2 |
| 513 | 1.228e-2 |

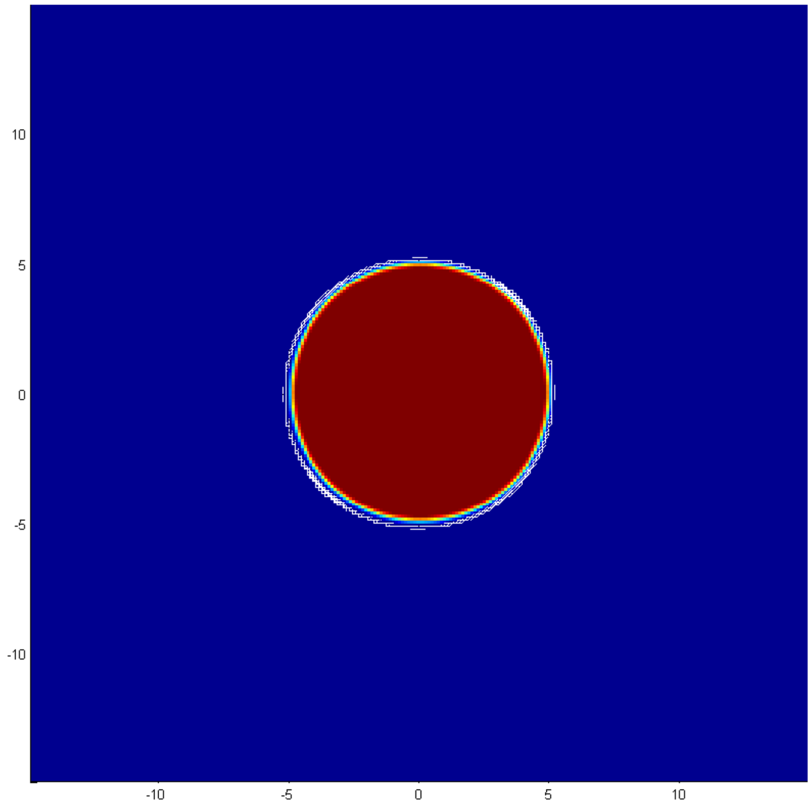


Figure 7.1: The ray trace results for the target $G = 1$ on a cone with radius of 5 degrees on $uv$-grid.

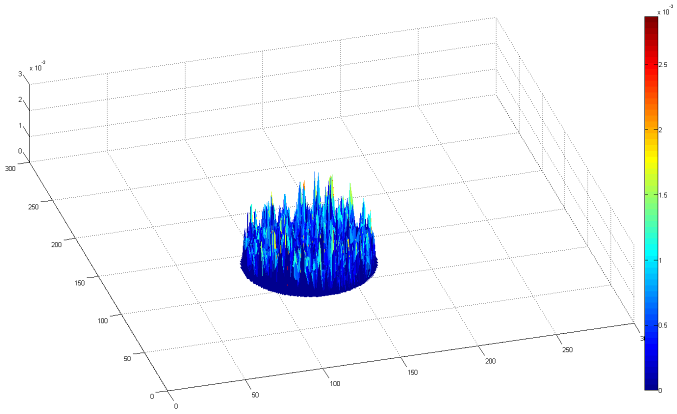


Figure 7.2: The relative absolute difference between the ray trace result and the target function $G = 1$

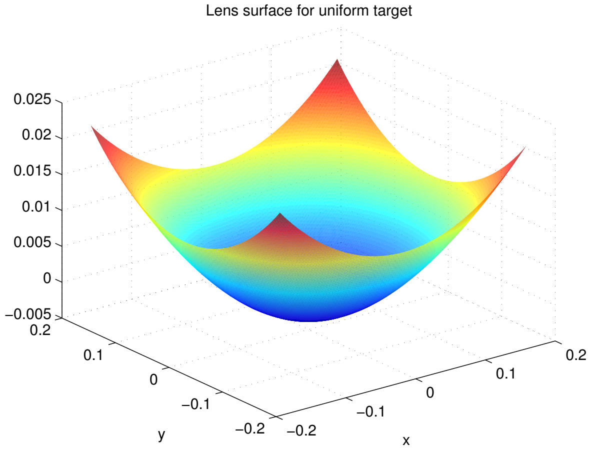


Figure 7.3: Plot of the second lens surface for producing a uniform intensity.

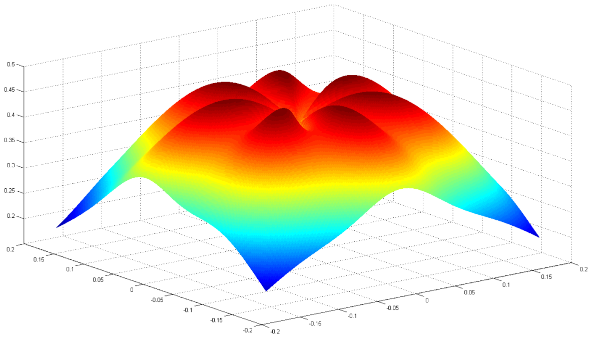


Figure 7.4: The transformed target density $g$

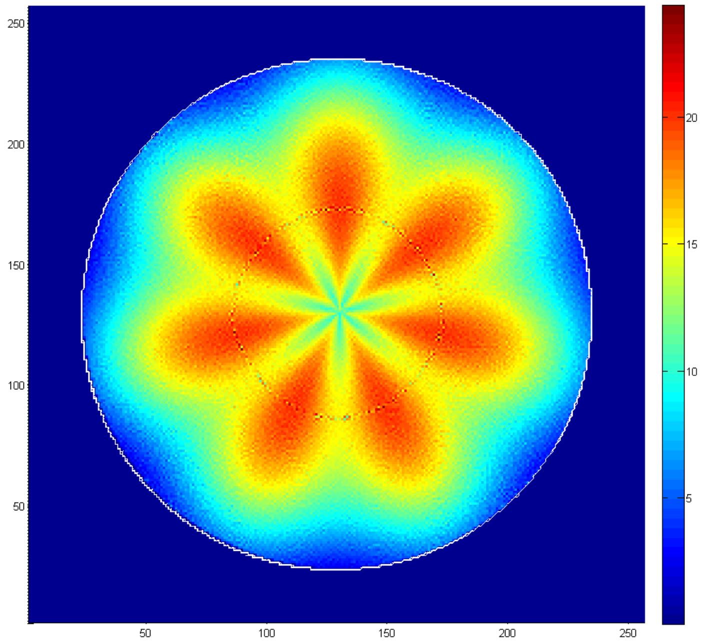


Figure 7.5: The ray tracing results for $\hat{G}$ produced with LightTools

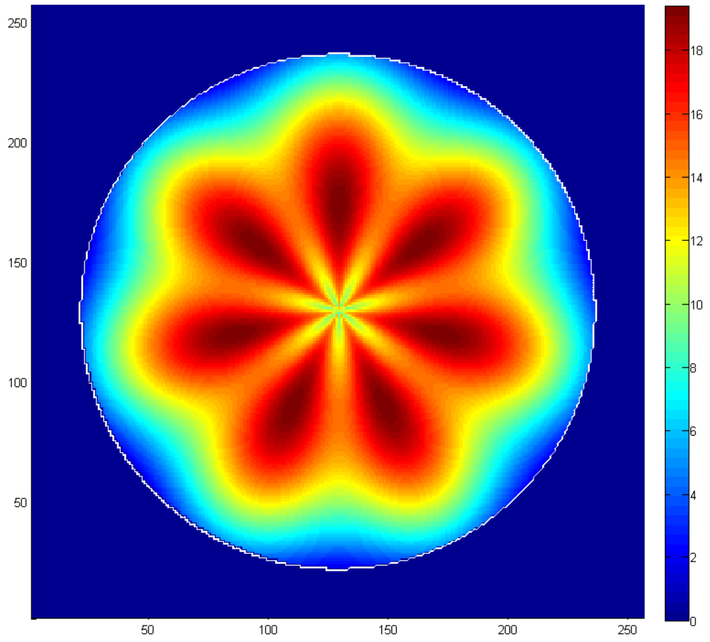


Figure 7.6: The theoretical target $\hat{G}$ on the same grid as the ray trace results

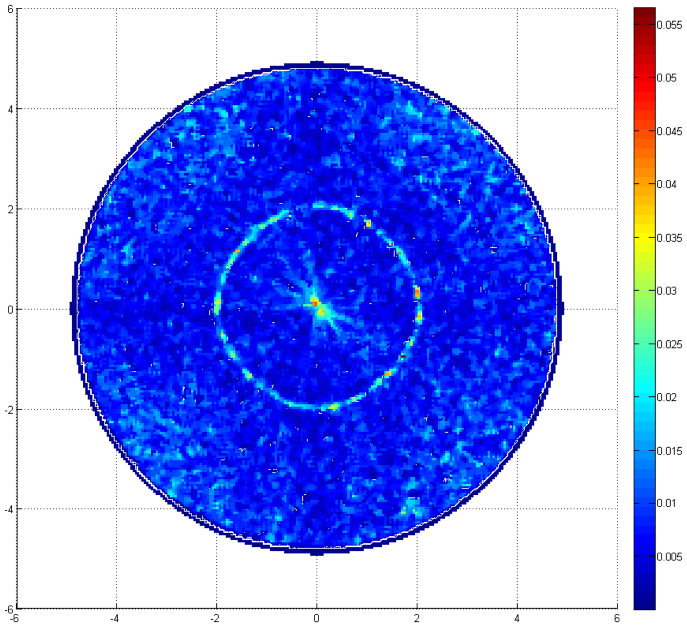
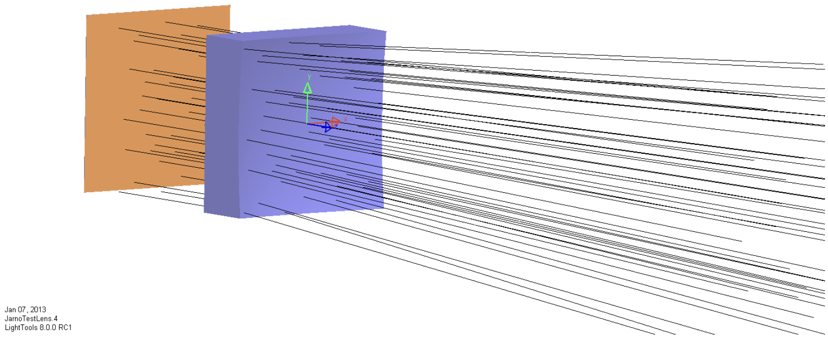
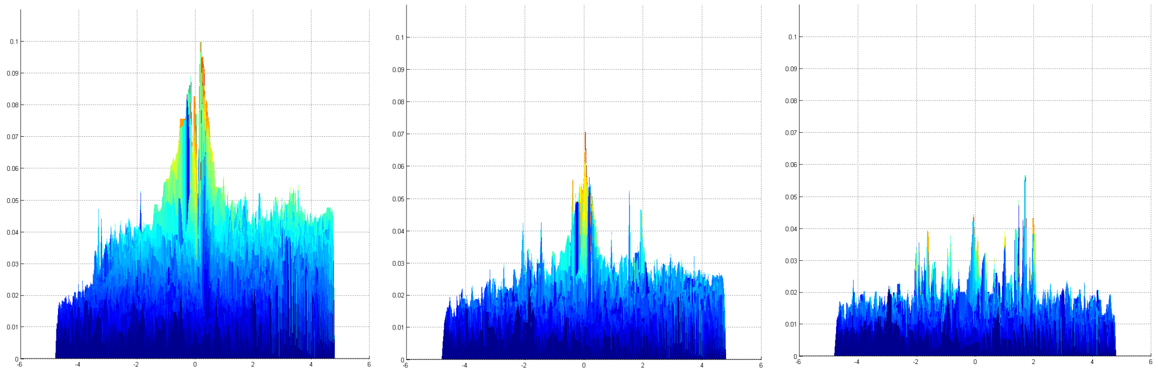


Figure 7.7: The relative error between the ray trace result and the target function $G$.



Jan 07, 2013
JarnoTestLens.4
LightTools 8.0.0 RC1

Figure 7.8: Screenshot of lens in LightTools



(a) $M_x = M_y = 129$

(b) $M_x = M_y = 257$

(c) $M_x = M_y = 513$

Figure 7.9: A plot of the absolute value of the relative error of the ray trace results compared to the intended target for three different grid sizes. For comparison all three plots are made with the same axis.

# Chapter 8

# Conclusions and Discussion

Based on our research the following conclusions can be reached:

1. It is possible to describe an optical system in the framework of optimal transport. This means that we want to find a function (representing the optical surface) that minimises a certain cost function under the condition that the conservation law of luminous flux is obeyed. It is very important to determine exactly which cost function is appropriate /needed for a certain optical problem. Further research is needed on the topic of determining appropriate cost functions for different optical systems.

2. In Chapters 3 and 4 it is shown that using variational calculus a system of partial differential equations can be derived, whose solution is the extremal of the minimisation formulation of the optical problem. Unfortunately a search in literature did not yield a method to solve the system of equations. The highly nonlinear and coupled nature of the equations like (4.11) and the lack of a clear set of boundary conditions make it very hard to solve this directly.

3. For optical problems based on a parallel beam source and a far-field target, it was shown that the various Monge-Ampère equations (4.17) and (4.22) can be written in the same form. This allows us to implement different transformation procedures that take a target distribution and an optical setup and produce a mathematical target distribution fit for a general solver. It would be very interesting for a further study to do the same for point-source problems. The power of this approach is that different geometries can be handled relatively easy and only a generic solver has to be implemented, which does not need any specific knowledge of the optical problem at hand.

4. In Chapter 5 a numerical solver was described. The theory behind the algorithm chosen turned out to include some very interesting facts. A method for handling the boundary condition was introduced using a defining function containing the geometrical information of the domain. This approach has as a big advantage that a global condition is transformed into a non-linear point wise PDE constraint. The approach allows the mapping to move along the boundary during the iteration if that is needed for finding a solution. Note that as the boundary condition is solved together with the interior in a single Newton iteration, the solution is not exact.

5. The numerical solver was shown to be able to handle a wide range of different problems. From the numerical experiments and the theoretical background the following

limitations are present:

- Various target intensities can be handled, as long as they are continuous. Furthermore the target intensity should not (locally) get very close to zero. As a rule of thumb a lower bound of 10 percent of the peak value can be used. If this is not done, the algorithm might produce unexpected results, or local artifacts
- The target geometry should be convex.
- The source density has no a priori constraints. However the source domain is padded with zeros to create a rectangular computation domain. This is something the algorithm can handle, but occasionally gives problems. If large parts of the (computational) source domain have very low to no intensity all the light rays from those points are mapped to the same location in target space. This results in a singularity of the solution, as the mapping loses its invertibility, by being locally many to one. For practical cases this problem might be mitigated by padding with some nonzero number instead of zeros. This modifies the problem, and so the solution is slightly different from the original. However if the numerical instabilities are reduced the actual result could be more accurate.
- Although mappings should be invertible, it turns out that if the source or target densities approach zero in part of their domains, this property is lost. At the very least it becomes much harder and less accurate to invert the mapping. The reason is that the derivatives of the mapping will approach either zero or infinity. The problem of numerically inverting is then very badly conditioned.

6. Despite theoretical limitations the solver is very robust. During the calculations of some of the harder problems artifacts were observed, often caused by discontinuities in the source distribution. By continuing the process the algorithm was able to cope with this and the artifacts disappeared in later iterations. This gives confidence that any error including loss of convexity in the first few iterations will not stop the program from producing useful results.

7. Lastly we concluded that combining the equations for the parallel beam lens the numerical program a real lens could be designed. On a first look (in the simulations) the pattern created by this lens resembles the target very good. The error, calculated by comparing the raytrace results with the intended target, is too large. However, this is only a first experiment that can be improved, and the results show that the approach taken has merit.

As already mentioned before there are some areas were further research is very welcome. Some ideas are

1. A study of the cost functions corresponding to optical systems when they are treated as an optimal mass transport problem. How is this influenced by the type of source, e.g., point source, parallel beam, extended source? How is it influenced by the optical element, e.g. reflector or lens? Are there other aspects, for example if the target is near field instead of far field?

2. It would be interesting to see how the Monge-Ampère equations for other optical problems look like. Do they also adhere to the same common form as was seen in Section 4.2?

3. Look if the treatment of the boundary condition in the numerical chapter can be used in the analysis of the Euler-Lagrange equations. This way of defining the boundary might be useful in more ways then we use now.

4. The optics calculated using the formulae for a parallel beam source should be evaluated. For different densities and geometries, one should see what the errors are, and if the program can be enhanced.

5. Investigating the robustness and convergence properties for typical optical problems.

6. For non rectangular source domains the function $f$ is now extended by zeros onto a rectangular domain. As this creates instabilities and makes inverting the result impossible we recommend to look for alternatives.

# Appendix A

# Inverting a Mapping

A common workaround for some of the limitations of the numerical solver is to switch the role of the source and target. This corresponds to the reversibility of the direction of light rays. This requires us to be able to invert the mapping. The first approach is a very naive and straight forward inverting. Suppose we have a set $X$ and a set $Y$ and calculated the mapping $\mathbf{s} : X \to Y$ as $\mathbf{s} = \nabla u$. We know this mapping on a grid on $X$. Now we create a new grid over the target set $Y$. Now for a vector $\mathbf{y} \in Y$ we calculate

$$\mathbf{s}^{-1}(\mathbf{y}) = \{x \in X | s(\mathbf{x}) = \mathbf{y}\}. \tag{A.1}$$

This can be done using a gradient descent method, or a Newton iteration (as we implemented). Doing this for each grid point, gives a numerical approximation of the inverted mapping. However we do not get the corresponding potential, which we need for the optics.

The second approach is to use an important observation from mass transport [Eva01, Section 3.1]. For quadratic cost functions, which lead to MA-type equations, the convex potential has a dual function. Assume the originally calculated potential is $u : X \to \mathbb{R}^+$, then the dual $v : Y \to \mathbb{R}^+$ is given by

$$v(\mathbf{y}) = \max_{\mathbf{x} \in X} \left( \mathbf{x} \cdot \mathbf{y} - u(\mathbf{x}) \right). \tag{A.2}$$

The current code has a very simple implementation of this, with promising results. It is much more robust against any numerical problems. It is suggested to implement this better and more accurately.

# Appendix B

# Jacobi Matrices

In this section, a more detailed and implementation ready description of the Jacobi matrices are given.

## B.1 Stable scheme

Recall that in (5.17) the system itself is given. We now look at this for a particular point $(i, j)$, with corresponding index $k = (j-1) * M_x + i$ in the interior of the domain. The minimum is obtained for a basis denoted by $\{(a, b), (-b, a)\}$.

The $k$-th equation in $\mathbf{N}$ is then given by

$$\begin{aligned} N_k = & \max\{D[a,b], \delta\} \cdot \max\{D[-b,a], \delta\} + \min\{D[a,b], \delta\} + \min\{D[-b,a], \delta\} \\ & - F(x_1, x_2)/G\left(aD_s[a,b] - bD_s[-b,a], bD_s[a,b] + aD_s[-b,a]\right), \end{aligned} \tag{B.1}$$

where the arguments $i$ and $j$ for $D$ and $D_s$ are dropped for readability. $D_s$ is the approximation of the first derivative, used for the gradient, and $D$ is the derivative operator as defined in (5.18).

The $k$-th row of the Jacobian contains the derivatives of $N_k$ with respect to $u_{i,j}$ (in the right ordering of course). The exact composition is given below. Note however that the Jacobian depends on the fact which of the values the different maximum and minimum operators take. For ease of notation the following rule is established a comparison $(x > a)$ has the value 1 if true, and 0 if false. This way it can be used in formula's without needing extensive case distinctions (or used in programming directly).

Looking at the formula above one sees that there are five non zero values in the $k$-th row, these are at $\{(i,j), (i+a, j+b), (i-a, j-b), (i-b, j+a), (i+b, j-a)\}$. In the formula's double indexing is used for clarity. Note that $J((i,j),(k,l)) = \frac{\partial N_{i,j}}{\partial u_{k,l}}$. We use a short hand for the gradient of u, defined as

$$u_{grad} = \nabla u(x_1(i,j), x_2(i,j)) = \frac{(aD_s[a,b] - bD_s[-b,a], \quad bD_s[a,b] + aD_s[-b,a])}{a^2 + b^2}$$

$$\begin{aligned}\frac{\partial N_{i,j}}{\partial u_{i,j}} =& -\frac{2}{(a^2+b^2)h^2}\Big[(D[a,b]>\delta)\cdot\max\{D[-b,a],\delta\}+(D[a,b]<\delta)\cdot 1\\ &+(D[-b,a]>\delta)\cdot\max\{D[a,b],\delta\}+(D[-b,a]<\delta)\cdot 1\Big],\end{aligned} \tag{B.2}$$

$$\begin{aligned}\frac{\partial N_{i,j}}{\partial u_{i+a,j+b}} =& \frac{(D[a,b]>\delta)\cdot\max\{D[-b,a],\delta\}+(D[a,b]<\delta)\cdot 1}{(a^2+b^2)h^2}\\ &+\frac{F(x)\cdot(a\cdot G_1(u_{grad})+b\cdot G_2(u_{grad}))}{G(u_{grad})^2\cdot h\sqrt{a^2+b^2}},\end{aligned} \tag{B.3}$$

$$\begin{aligned}\frac{\partial N_{i,j}}{\partial u_{i-a,j-b}} =& \frac{(D[a,b]>\delta)\cdot\max\{D[-b,a],\delta\}+(D[a,b]<\delta)\cdot 1}{(a^2+b^2)h^2}\\ &+\frac{F(x)\cdot(-a\cdot G_1(u_{grad})-b\cdot G_2(u_{grad}))}{G(u_{grad})^2\cdot h\sqrt{a^2+b^2}},\end{aligned} \tag{B.4}$$

$$\begin{aligned}\frac{\partial N_{i,j}}{\partial u_{i-b,j+a}} =& \frac{(D[-b,a]>\delta)\cdot\max\{D[a,b],\delta\}+(D[-b,a]<\delta)\cdot 1}{(a^2+b^2)h^2}\\ &+\frac{F(x)\cdot(-b\cdot G_1(u_{grad})+a\cdot G_2(u_{grad}))}{G(u_{grad})^2\cdot h\sqrt{a^2+b^2}},\end{aligned} \tag{B.5}$$

$$\begin{aligned}\frac{\partial N_{i,j}}{\partial u_{i+b,j-a}} =& \frac{(D[-b,a]>\delta)\cdot\max\{D[a,b],\delta\}+(D[-b,a]<\delta)\cdot 1}{(a^2+b^2)h^2}\\ &+\frac{F(x)\cdot(+b\cdot G_1(u_{grad})-a\cdot G_2(u_{grad}))}{G(u_{grad})^2\cdot h\sqrt{a^2+b^2}}.\end{aligned} \tag{B.6}$$

### B.1.1 Boundary modifications

Let's look at the modifications near the boundary, and the resulting changes to the Jacobian. We show the case where $i+a$ is out of bounds. This means that we apply approximations near the boundary, and as such that $D$ and $D_s$ are no longer the same in bases directions. So the relevant is again (B.1),

$$\begin{aligned}N_k =& \max\{D[a,b],\delta\}\cdot\max\{D[-b,a],\delta\}+\min\{D[a,b],\delta\}+\min\{D[-b,a],\delta\}\\ &-F(x_1,x_2)/G\left(aD_s[a,b]-bD_s[-b,a],bD_s[a,b]+aD_s[-b,a]\right)\end{aligned} \tag{B.7}$$

where $D$ and $D_s$ now take the following definitions depending on there argument

$$\begin{aligned}D[-b,a] &= \left(u_{i-b,j+a}+u_{i+b,j-a}-2u_{i,j}\right)/(h^2(a^2+b^2)),\\ D_s[-b,a] &= \left(u_{i-b,j+a}-u_{i+b,j-a}\right)/(2h\sqrt{a^2+b^2}),\\ D[a,b] &= \left(-\tfrac{1}{4}u_{i+\frac{a}{2},j+b}+\tfrac{6}{4}u_{i+\frac{a}{2},j}+\tfrac{3}{4}u_{i+\frac{a}{2},j-b}+u_{i-a,j-b}-3u_{i,j}\right)/(\tfrac{3}{4}h^2(a^2+b^2)),\\ D_s[a,b] &= \left(-\tfrac{1}{8}u_{i+\frac{a}{2},j+b}+\tfrac{6}{8}u_{i+\frac{a}{2},j}+\tfrac{3}{8}u_{i+\frac{a}{2},j-b}-u_{i-a,j-b}\right)/(\tfrac{3}{2}h\sqrt{a^2+b^2}).\end{aligned}$$

*Remark.* that $D[-b,a]$ and $D_s[-b,a]$ are unmodified from the regular case. However if close to a corner of the domain, also $j-a$ or $j+a$ might be out of bounds. Then $D[-b,a]$ and $D_s[-b,a]$ are modified and also use interpolation, this however can be treated independently from the current modifications to the Jacobian.

From the changes in the equation, we need to modify the Jacobian. Luckily the terms $\frac{\partial N_{i,j}}{\partial u_{i-b,j+a}}$ and $\frac{\partial N_{i,j}}{\partial u_{i+b,j-a}}$ stay the same. (they are modified only if $D[-b,a]$ has to use interpolation). The $\frac{\partial N_{i,j}}{\partial u_{i+a,j+b}}$ is replaced by 3 new terms. The new terms are

$$\begin{aligned}\frac{\partial N_{i,j}}{\partial u_{i,j}} =& -\frac{3}{\frac{3}{4}(a^2+b^2)h^2}\Bigg[(D[a,b]>\delta)\cdot\max\{D[-b,a],\delta\}+(D[a,b]<\delta)\cdot 1\Bigg] \\ & -\frac{2}{(a^2+b^2)h^2}\Bigg[(D[-b,a]>\delta)\cdot\max\{D[a,b],\delta\}+(D[-b,a]<\delta)\cdot 1\Bigg],\end{aligned} \tag{B.8}$$

$$\begin{aligned}\frac{\partial N_{i,j}}{\partial u_{i+a/2,j+b}} =& -\frac{1}{4}\frac{(D[a,b]>\delta)\cdot\max\{D[-b,a],\delta\}+(D[a,b]<\delta)\cdot 1}{\frac{3}{4}(a^2+b^2)h^2} \\ & +\frac{F(x)\cdot\big(\frac{-1}{8}a\cdot G_1(u_{grad})-\frac{1}{8}b\cdot G_2(u_{grad})\big)}{G(u_{grad})^2\cdot\frac{3}{2}h\sqrt{a^2+b^2}},\end{aligned} \tag{B.9}$$

$$\begin{aligned}\frac{\partial N_{i,j}}{\partial u_{i+a/2,j}} =& \frac{6}{4}\frac{(D[a,b]>\delta)\cdot\max\{D[-b,a],\delta\}+(D[a,b]<\delta)\cdot 1}{\frac{3}{4}(a^2+b^2)h^2} \\ & +\frac{F(x)\cdot\big(\frac{6}{8}a\cdot G_1(u_{grad})+\frac{6}{8}b\cdot G_2(u_{grad})\big)}{G(u_{grad})^2\cdot\frac{3}{2}h\sqrt{a^2+b^2}},\end{aligned} \tag{B.10}$$

$$\begin{aligned}\frac{\partial N_{i,j}}{\partial u_{i+a/2,j-b}} =& \frac{3}{4}\frac{(D[a,b]>\delta)\cdot\max\{D[-b,a],\delta\}+(D[a,b]<\delta)\cdot 1}{\frac{3}{4}(a^2+b^2)h^2} \\ & +\frac{F(x)\cdot\big(\frac{3}{8}a\cdot G_1(u_{grad})+\frac{3}{8}b\cdot G_2(u_{grad})\big)}{G(u_{grad})^2\cdot h\sqrt{a^2+b^2}},\end{aligned} \tag{B.11}$$

$$\begin{aligned}\frac{\partial N_{i,j}}{\partial u_{i-a,j-b}} =& \frac{(D[a,b]>\delta)\cdot\max\{D[-b,a],\delta\}+(D[a,b]<\delta)\cdot 1}{\frac{3}{4}(a^2+b^2)h^2} \\ & +\frac{F(x)\cdot(-a\cdot G_1(u_{grad})-b\cdot G_2(u_{grad}))}{G(u_{grad})^2\cdot\frac{3}{2}h\sqrt{a^2+b^2}}.\end{aligned} \tag{B.12}$$

The other modifications go in a similar fashion. In total there are 4 different exceptional cases and a *normal* case along the $x_1$-axis and the same amount along the $x_2$-axes, resulting in a total of 10 cases.

## B.2 Accurate scheme

For the accurate scheme, the Jacobian is filled along diagonals. This is given by

$$
\begin{aligned}
\frac{\partial N_{i,j}}{\partial u_{i-1,j-1}} =& 2\frac{u_{i+1,j+1}+u_{i-1,j-1}-u_{i-1,j+1}-u_{i+1,j-1}}{16h^4} && \text{(B.13)}\\
\frac{\partial N_{i,j}}{\partial u_{i-1,j+1}} =& -2\frac{u_{i+1,j+1}+u_{i-1,j-1}-u_{i-1,j+1}-u_{i+1,j-1}}{16h^4} && \text{(B.14)}\\
\frac{\partial N_{i,j}}{\partial u_{i+1,j-1}} =& -2\frac{u_{i+1,j+1}+u_{i-1,j-1}-u_{i-1,j+1}-u_{i+1,j-1}}{16h^4} && \text{(B.15)}\\
\frac{\partial N_{i,j}}{\partial u_{i+1,j+1}} =& 2\frac{u_{i+1,j+1}+u_{i-1,j-1}-u_{i-1,j+1}-u_{i+1,j-1}}{16h^4} && \text{(B.16)}\\
\frac{\partial N_{i,j}}{\partial u_{i,j}} =& -2*\frac{u_{i,j+1}+u_{i,j-1}+u_{i+1,j}+u_{i-1,j}-4u_{i,j}}{h^4} && \text{(B.17)}\\
\frac{\partial N_{i,j}}{\partial u_{i-1,j}} =& \frac{u_{i,j+1}+u_{i,j-1}-2u_{i,j}}{h^4}\\
&+\frac{F(x_1,x_2)\cdot G_1\left((u_{i+1,j}-ui-1,j)/(2h),(u_{i,j+1}-ui,j-1)/(2h)\right)}{G\left((u_{i+1,j}-ui-1,j)/(2h),(u_{i,j+1}-ui,j-1)/(2h)\right)^2\cdot 2h} && \text{(B.18)}\\
\frac{\partial N_{i,j}}{\partial u_{i,j-1}} =& \frac{u_{i+1,j}+u_{i-1,j}-2u_{i,j}}{h^4}\\
&+\frac{F(x_1,x_2)\cdot G_2\left((u_{i+1,j}-ui-1,j)/(2h),(u_{i,j+1}-ui,j-1)/(2h)\right)}{G\left((u_{i+1,j}-ui-1,j)/(2h),(u_{i,j+1}-ui,j-1)/(2h)\right)^2\cdot 2h} && \text{(B.19)}\\
\frac{\partial N_{i,j}}{\partial u_{i+1,j}} =& \frac{u_{i,j+1}+u_{i,j-1}-2u_{i,j}}{h^4}\\
&-\frac{F(x_1,x_2)\cdot G_1\left((u_{i+1,j}-ui-1,j)/(2h),(u_{i,j+1}-ui,j-1)/(2h)\right)}{G\left((u_{i+1,j}-ui-1,j)/(2h),(u_{i,j+1}-ui,j-1)/(2h)\right)^2\cdot 2h} && \text{(B.20)}\\
\frac{\partial N_{i,j}}{\partial u_{i,j+1}} =& \frac{u_{i+1,j}+u_{i-1,j}-2u_{i,j}}{h^4}\\
&-\frac{F(x_1,x_2)\cdot G_2\left((u_{i+1,j}-ui-1,j)/(2h),(u_{i,j+1}-ui,j-1)/(2h)\right)}{G\left((u_{i+1,j}-ui-1,j)/(2h),(u_{i,j+1}-ui,j-1)/(2h)\right)^2\cdot 2h}. && \text{(B.21)}
\end{aligned}
$$

# Appendix C

# Implementation Details

This chapter contains details about the implementation, including a list of definitions of the parameters that the program supports.

## C.1 Parameter definitions

The program supports the following options that can be passed using the *setOptions()* function.

$M_x$ Number of grid points along the $x$-axis (default value 256)

$M_y$ Number of grid points along the $y$-axis (default value 256)

$N_y$ Number of normal vectors used for discretising the boundary of the target $Y$ (default value 32)

**maxit** Maximum number of iterations (default value 40)

**rtol** Relative tolerance for stopping criteria (default value $10^{-4}$)

**atol** Absolute tolerance for stopping criteria (default value $10^{-5}$)

**stencil** Allowed values: $2, 4$, number of directions used in the stable operator (default value 4)

**H_scheme** Selector for discretisation scheme used along the boundary of the target. Either a two-point scheme (H_scheme=0) or a three-point scheme (H_scheme=1 (default)).

**min_damping** The amount of damping applied at least in each iteration. The damping factor $\beta <$ min_damping (default value 1).

**init_damping** The amount of damping used for the first iteration (default value 1).

**delta** Value of $\delta$ as defined in the stable scheme (default value $10^{-6}$).

**debug_mode** Boolean value, if *true* the program plots the residual and other diagnostic information during each iteration of the algorithm. (default *false*)

**anchor_weight** Weight of the reference value of $u$. (default value 1).

# Bibliography

[BB00] Jean-David Benamou and Yann Brenier, *A computational fluid mechanics solution to the monge-kantorovich mass transfer problem*, Numerische Mathematik **84** (2000), 375–393. 10.1007/s002110050002.

[BFO10] Jean-David Benamou, Brittany D. Froese, and Adam M. Oberman, *Two numerical methods for the elliptic monge-ampère equation*, ESAIM: Mathematical Modelling and Numerical Analysis **44** (2010), no. 04, 737–758, available at http://journals.cambridge.org/article_S0764583X10000178.

[BFO12a] Jean-David Benamou, Brittany Froese, and Adam Oberman, *Numerical solution of the optimal transportation problem using the monge-ampère equation* (2012), available at http://arxiv.org/abs/1208.4870.

[BFO12b] ———, *Numerical solution of the optimal transportation problem via viscosity solutions for the monge-ampère equation* (2012), available at http://arxiv.org/abs/1208.4870.

[BFO12c] ———, *Numerical solution of the second boundary value problem for the elliptic monge-ampère equation* (June 2012), available at http://hal.inria.fr/hal-00703677/PDF/bof.pdf. http://hal.inria.fr/hal-00703677.

[BM07] Pablo Benitez and Juan C. Minano, *The future of illumination design*, Opt. Photon. News **18** (2007May), no. 5, 20–25, available at http://www.osa-opn.org/home/articles/volume_18/issue_5/features/the_future_of_illumination_design.

[BN12] Susanne Cecelia Brenner and Michael Neilan, *Finite element approximations of the three dimensional monge-ampère equation*, ESAIM: Mathematical Modelling and Numerical Analysis **46** (2012), no. 05, 979–1001, available at http://journals.cambridge.org/article_S0764583X11000677.

[CC03] Andrej Cherkaev and Elena Cherkaev, *Calculus of variations and applications*, 2003. Lecture Notes, http://www.math.utah.edu/~cherk/teach/5710-03/print10-19.pdf.

[CVWB09] R. Chartrand, K. Vixie, B. Wohlberg, and E. Bollt, *A gradient descent solution to the monge-kantorovich problem*, Appl. Math. Sci **3** (2009), 1071–1080.

[DG06] E.J. Dean and R. Glowinski, *Numerical methods for fully nonlinear elliptic equations of the monge-ampère type*, Computer Methods in Applied Mechanics and Engineering **195** (2006), no. 13-16, 1344 –1386. A Tribute to Thomas J.R. Hughes on the Occasion of his 60th Birthday.

[Eva01] Lawrence C. Evans, *Partial differential equations and monge-kantorovich mass transfer*, Department of mathematics, University of California, Berkeley, 2001. http://math.berkeley.edu/~evans/Monge-Kantorovich.survey.pdf.

[FO11a] Brittany D. Froese and Adam M. Oberman, *Convergent finite difference solvers for viscosity solutions of the elliptic monge–ampère equation in dimensions two and higher*, SIAM Journal on Numerical Analysis **49** (2011), no. 4, 1692–1714, available at http://link.aip.org/link/?SNA/49/1692/1.

[FO11b] ———, *Fast finite difference solvers for singular solutions of the elliptic monge-ampère equation*, Journal of Computational Physics **230** (2011), no. 3, 818 –834, available at http://www.sciencedirect.com/science/article/pii/S0021999110005760.

[FO12] ———, *Accurate convergent finite difference approximations for viscosity solutions of the elliptic monge-ampère partial differential equation*, ArXiv e-prints (April 2012), available at http://arxiv.org/abs/1204.5798.

[Fro12] Brittany D. Froese, *A numerical method for the elliptic monge–ampère equation with transport boundary conditions*, SIAM Journal on Scientific Computing **34** (2012), no. 3, A1432–A1459, available at http://link.aip.org/link/?SCE/34/A1432/1.

[Hec02] Eugene Hecht, *Optics*, Fourth edition, Addison Wesley, San Fransisco, CA, 2002.

[HRT10] Eldad Haber, Tauseef Rehman, and Allen Tannenbaum, *An efficient numerical method for the solution of the $l_2$ optimal mass transfer problem*, SIAM J. Sci. Comput. **32** (2010), 197–211.

[Int06] International Energy Agency, *Light's labour's lost*, OECS/IEA, Paris, 2006. http://www.iea.org/publications/freepublications/publication/light2006.pdf.

[Mae97] Maurice Maes, *Mathematical methods for reflector design*, Ph.D. Thesis, 1997.

[MRtTB05] R. M. M. Mattheij, S. W. Rienstra, and J. H. M. ten Thije Boonkkamp, *Partial differential equations*, Society for Industrial and Applied Mathematics, Philadephia, PA, 2005.

[Obe05] Adam M. Oberman, *A convergent difference scheme for the infinity Laplacian: construction of absolutely minimizing Lipschitz extensions*, Math. Comp. **74** (2005), no. 251, 1217–1230 (electronic), available at http://dx.doi.org/10.1090/S0025-5718-04-01688-6. MR2137000 (2006h:65165)

[Obe08] ———, *Wide stencil finite difference schemes for the elliptic monge-ampère equation and functions of the eigenvalues of the hessian*, Discrete Cont. Dyn. Syst. Ser. B **10** (2008), 221–238, available at http://aimsciences.org/journals/displayArticles.jsp?paperID=3350.

[PPP07] Frank L. Pedrotti, Leno M. Pedrotti, and Leno S. Pedrotti, *Introduction to optics*, Third edition, Pearson Education, Upper Saddle River, NJ, 2007.

[Pri12] Corien Prins, *Private communication*, 2012.

[PTVF07] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery, *Numerical recipes*, Cambridge university press, 2007.

[SAK10] Louis-Philippe Saumier, Martial Agueh, and Boualem Khouider, *An efficient numerical algorithm for the l2 optimal transport problem with applications to image processing*, 2010. available at http://arxiv.org/abs/1009.6039.

[SWR11] Mohamed Sulman, J.F. Williams, and R.D. Russell, *Optimal mass transport for higher dimensional adaptive grid generation*, Journal of Computational Physics **230** (2011), no. 9, 3302 –3330, available at http://dx.doi.org/10.1016/j.jcp.2011.01.025.

[Wik12a] Wikipedia, *Ecological light pollution — wikipedia, the free encyclopedia*, 2012. [Online; accessed 14-December-2012], available at http://en.wikipedia.org/w/index.php?title=Ecological_light_pollution&oldid=518976209.

[Wik12b] ———, *Light pollution — wikipedia, the free encyclopedia*, 2012. [Online; accessed 14-December-2012], available at http://en.wikipedia.org/w/index.php?title=Light_pollution&oldid=526597489.

[Wik12c] ———, *Nonimaging optics — wikipedia, the free encyclopedia*, 2012. [Online; accessed 17-December-2012], available at http://en.wikipedia.org/w/index.php?title=Nonimaging_optics&oldid=518995383.

[Wik12d] ———, *Supporting hyperplane — wikipedia, the free encyclopedia*, 2012. [Online; accessed 19-November-2012], available at http://en.wikipedia.org/w/index.php?title=Supporting_hyperplane&oldid=513611392.

# List of Figures