Mesh Generation Using Numerical Solutions of an Optimal Transport Problem

By

Joshua Prettyman

CID number: 00825958
UoR ID number: 23022044

Project Supervisors:

Hilary Weller and **Phil Browne**
Meterology, University of Reading

Thesis submitted as part of the requirements for the award of the MRes in Mathematics of Planet Earth, Imperial College London and University of Reading

August 28, 2015

Contents

Chapter 1

Introduction

1.1 The need for adaptive mesh generation

Within a system of time-dependent PDEs which may arise in applications such as meteorology [Weller et al., 2015], engineering, medicine [Lee et al., 2013] and astrophysics [Fryxell et al., 2000], there may be features that change or advance significantly during the integration of the system [Budd et al., 2009]. Discretising these equations on a fixed, uniform grid (such at the Lat-lon coordinate grid traditionally used in meteorology [Griffies et al., 2000]) can pose various barriers to computational efficiency [Browne et al., 2014, Weller, 2009]: such grids may "waste" computational power solving equations in regions where the solution does not require high resolution to achieve low errors, or may fail to resolve important features on scales just below the grid scale. Such observations motivate the study of adaptive grid methods. Adaptive grids are generated in parallel with the equation solutions and adapt to the solutions [Budd et al., 2009], they can thus resolve moving features (advancing weather fronts, for example) at fine scales whilst the rest of the computational grid remains at a coarser level. It is the use of adaptive grids which motivates this project.

Adaptive methods come in three main flavours, $h-$, $p-$ and $r-$adaptivity [Tang, 2005, Budd and Williams, 2009]. p-adaptivity involves increasing or decreasing the order of the polynomial approximation on each element according to a monitor function. h-adaptivity describes methods where points are removed, or new points added, to coarsen or to refine the grid locally. This can result in block-structured adaptivity where a rectangular grid is locally subdivided into smaller rectangles; or nonconforming adaptivity (see [Power et al., 2006, Saint-Cyr et al., 2007]) if new points are added in a non-uniform way which changes the structure. r-adaptive methods, where the r stands for *relocation*, or *redistribution*, do not add in new mesh points but move the existing points. The basic principle of r-adaptive mesh generation is to find a continuous map $\mathbf{F}$ from the set of *fixed* computational coordinates $\xi \in \Omega_C$ onto the set of *moving* physical coordinates $\mathbf{x} \in \Omega_P$ so that $\mathbf{x} = \mathbf{F}(\xi, t)$.

In this report we consider an adaptivity method based on the equidistri-

bution problem, which makes use of solutions to the Monge-Ampère equation. This Monge-Ampère (MA) approach is an example of a *location*-based r-adaptive method, where the locations of mesh points are determined by an equation (typically a non-linear differential equation) governing how the mesh point density moves. This is as opposed to fully Lagrangian methods where individual mesh points are advected with the solutions of the underlying equations. For a discussion of location-based and Lagrangian methods, see [Cao et al., 2003, Budd et al., 2009, Chacón et al., 2011, Lee et al., 2013].

1.2 Adaptive mesh redistribution and optimal transport

Adaptive mesh redistribution, or r-adaptivity, attempts to create a new mesh by deforming an existing mesh in space whilst maintaining the number of mesh points and the topology (or connectivity) of the mesh. In this project we have implemented some r-adaptive methods in two dimensions by deforming an initial computational mesh $\mathcal{T}_C$ in computational space $\Omega_C \subseteq \mathbb{R}^2$ into a new mesh $\mathcal{T}_P$ in physical space $\Omega_P \subseteq \mathbb{R}^2$. The space Ω_C in this report is the unit square $\left[-1/2, 1/2\right]^2$ and the computational mesh $\mathcal{T}_C$ is a uniform subdivision of the space into smaller squares. Ω_P is also the unit square $\left[-1/2, 1/2\right]^2$, and $\mathcal{T}_P$ is a distorted mesh with the same connectivity as $\mathcal{T}_C$. We refer to a point $\xi \in \Omega_C$ using coordinates (ξ, η) and a point $\mathbf{x} \in \Omega_P$ using coordinates (x, y). The analysis in this chapter can be simply extended to three dimensions, however some aspects of the algorithms described in the following chapter are derived specifically from the properties of the 2D case and the analysis of the results obtained from the implementation of those algorithms clearly apply to those particular 2D cases.

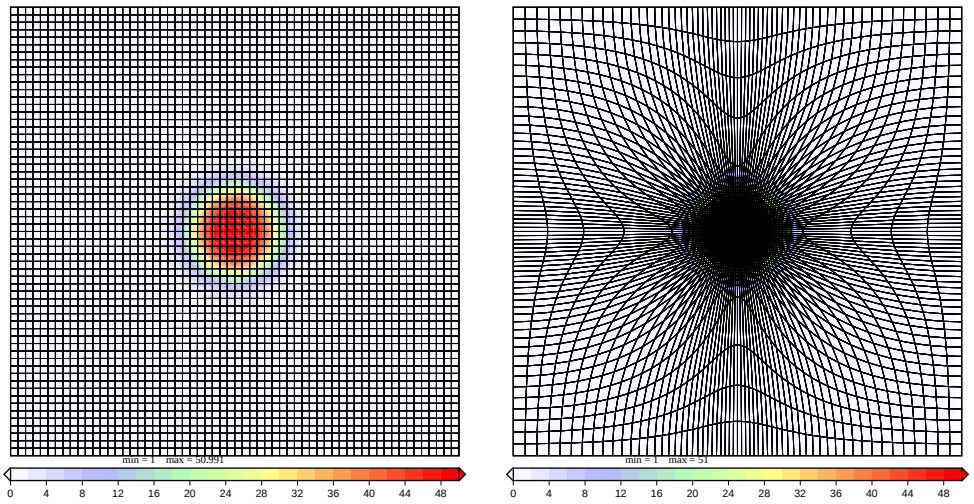

Figure 1.1: A contour plot of a bell-shaped monitor function plotted on the uniform rectangular mesh $\mathcal{T}_C$ on the space $[-1/2, 1/2]^2$ (left); and the new mesh $\mathcal{T}_P$ equidistributed using Optimal Transport to be dense where the monitor function is high (right).

The problem of mesh redistribution is to find a map $\mathbf{F} : \Omega_C \to \Omega_P$ such that the image $\mathcal{T}_P = \mathbf{F}(\mathcal{T}_C)$ of the computational mesh has the properties required or

desired of a mesh on which to solve the underlying equations, as illustrated in figure 1.1. We would like for $\mathbf{F}$ to redistribute the points of the computational mesh so that there are regions in Ω_P with high densities of mesh points and other regions with low densities. In practice we expect that mesh points should be concentrated in regions where the underlying equations are exhibiting interesting, small-scale or complex features which require high resolution to solve accurately. We assume that there exists a monitor function $M(\mathbf{x}, t) > 0$ which is proportional to the required mesh point density at time t.

The map $\mathbf{F}$ will depend on the monitor function $M(\mathbf{x}, t)$, it will also therefore have a time dependency. At each time t, the map $\mathbf{F}$ will be found in order to distribute the mesh points proportionally to the specific monitor function $M(\mathbf{x}, t)$. This is in essence the principal of adaptive mesh redistribution since we desire that the mesh is redistributed in real time as the underlying equations evolve.

Algorithm 1 Adaptive Mesh Redistribution with a static monitor function

Read the initial mesh $\mathcal{T}$
Considering the monitor function M, solve some (possibly iterative) system to find the map $\mathbf{F}$
$\mathcal{T} \leftarrow \mathbf{F}(\mathcal{T})$
return the new mesh $\mathcal{T}$

In this report we consider the *static* mesh redistribution problem described by Algorithm 1. The *dynamic* method, described in Algorithm 2 returns a dynamically changing mesh defined by a dynamic monitor function $M(t)$. It is simply Algorithm 1 repeated over discrete time steps, with a changing monitor function. Since applications which could utilise an adaptive mesh may run for millions of time steps, it is important that Algorithm 1 is a fast as possible.

Algorithm 2 Adaptive Mesh Redistribution with a dynamic monitor function

Read the initial mesh $\mathcal{T}$
$t \leftarrow 0$
while $t < t_{\max}$ **do**
    Apply Algorithm 1 with monitor function $M(t)$
    // Note the mesh returned is assigned to $\mathcal{T}$.
    // It is thus the 'initial' mesh at the next time step.
    $t \leftarrow t + \delta t$
end while
This algorithm will have returned a sequence of meshes $\{\mathcal{T}_t\}_{t=0}^{t_{\max}}$ each defined by the monitor function at time t.

In order to find the map $\mathbf{F}$ we must know which properties we require it to have. We would like that the concentration of mesh points is high where the monitor function is high; we will achieve this by requiring that the mesh $\mathcal{T}_P$ is *equidistributed* with respect to the monitor function. That is, if if some non-empty set $A \in \Omega_C$ occupies a certain fraction of the whole computational space Ω_C then the corresponding image set $\mathbf{F}(A) \in \Omega_P$ occupies the same fraction of the physical

space Ω_P *with respect to* the monitor function. More precisely,

$$\frac{\int_A d\xi}{\int_{\Omega_C} d\xi} = \frac{\int_{\mathbf{F}(A)} M(\mathbf{x},t)d\mathbf{x}}{\int_{\Omega_P} M(\mathbf{x},t)d\mathbf{x}} \tag{1}$$

[Browne et al., 2014]. This will allow us to use the monitor function to control the cell volumes, which is where the adaptivity comes from. Figure 1.1 shows a mesh equidistributed with respect to a bell-shaped monitor function.

Lemma 1.2.1. *[Budd and Williams, 2009] If the mesh* $\mathcal{T}_P = \{\mathbf{x}\} = \{\mathbf{F}(\xi)\}$ *equidistributes the monitor function* M *then it satisfies the Equidistribution Equation*

$$M(\mathbf{x})|J(\xi)| = c \text{ (const.)} \tag{2}$$

Where $c = \frac{\int_{\Omega_P} M(\mathbf{x})d\mathbf{x}}{\int_{\Omega_c} d\xi}$ *and* $J(\xi)$ *is the Jacobian of the map from* Ω_C *to* Ω_P*, given by* $\nabla_\xi \mathbf{x}$.

See [Budd and Williams, 2009] for proof of the Lemma. Clearly c, M and J have a time dependence in moving (dynamic) mesh applications but we will drop this for simplicity whenever an expression is being considered 'frozen' at time t, that is, when we consider the static mesh problem.

1.2.1 Optimal Transport

It is noted by [Budd et al., 2009] and [Chacón et al., 2011] that the equdistribution principle is not alone a good basis for mesh-generation. In greater then one dimension the equdistribution principle can lead to ill-posed problems since there may be infinitely many equidistributed meshes, many of which will be unsuitable for applications. As described in [Budd et al., 2009] we impose the additional constraint of Optimal Transport (OT) to the equidistribution problem. OT is the condition that the new mesh points $\{\mathbf{F}(\xi)\}$ move as little as possible from their initial positions $\{\xi\}$. By 'move as little as possible' we mean that we seek to minimise the Wasserstein metric I [Browne et al., 2014] where

$$I = \int_{\Omega_C} |\mathbf{F}(\xi) - \xi|^2 d\xi \tag{3}$$

that is, $\mathbf{F}$ is as close as possible to the identity map whilst still satisfying the equidistribution property 2. It is suggested that Optimal Transport (OT) is a natural choice of constraint because moving mesh points as little as possible ought to give a mesh close to the initial mesh, which can have useful properties such as orthogonality and uniformity. Furthermore, if solved exactly, OT will not cause any mesh tangling [Weller et al., 2015] which occurs when points move too quickly relative to each other, and OT leads to proof of existence and uniqueness of solutions [Budd et al., 2009]. However, it is not shown or claimed in this report that Optimal Transport is the best choice of constraint; [Budd et al., 2013] note that there is a vast literature on the use of different constraints. In this report we

consider OT because it gives rise to a number of potentially interesting methods for obtaining an equidistributed mesh.

[Brenier, 1991] shows that there exists a unique mapping $\mathbf{F} : \Omega_C \to \Omega_P$ which satisfies equation 2 and minimises I. Additionally, we get the following lemma:

Lemma 1.2.2. *[Brenier, 1991] The mapping $\mathbf{F} : \Omega_C \to \Omega_P$ satisfying 2 is unique and can be written as the gradient of a convex mesh potential P:*

$$\mathbf{F}(\xi) = \nabla_\xi P(\xi) \quad \textit{and} \quad \nabla^2_\xi P(\xi) > 0 \tag{4}$$

If $\mathbf{x} = \mathbf{F}(\xi)$, as in equation 2 and $\mathbf{F}(\xi) = \nabla P(\xi)$ as in equation 4 then

$$|J(\xi)| = \det(\nabla \mathbf{x}) = |\nabla\nabla P(\xi)| = |H(P)| \tag{5}$$

Where $H(P) = \nabla\nabla P(\xi)$ denotes the Hessian of $P(\xi)$, and ∇^2 denotes the Laplacian operator. Note that we have omitted the ξ subscript when taking gradients, all gradient are assumed to be with respect to ξ unless otherwise denoted. The equidistribution equation 2 with the constraint supplied by Optimal Transport then becomes the Monge-Ampère equation

$$M(\nabla P)|H(P)| = c \text{ (const.)} \tag{6}$$

which is a fully non-linear elliptic PDE [Budd et al., 2009]. Finding an equidistributed mesh defined by a map $\mathbf{F}(\xi)$ which satisfies the optimal transport constraint then becomes the problem of finding a solution to the Monge-Ampère equation 6.

1.3 Numerical solutions to the Monge-Ampère Equation

The Monge-Ampère equation (equation 6), with P convex, is a fully non-linear elliptic PDE and as such is difficult to solve accurately [Browne et al., 2014]. There are many solution methods available in the literature [Froese and Oberman, 2011, Froese, 2012, Browne et al., 2014] and many employ a Newton-type algorithm to solve a discretisation of the equation, [Froese and Oberman, 2011]. We are given another route to a solution if we observe that an accurate solution of the MA equation is not necessary, indeed finding such a solution is very costly and in the context of adaptive meshes the purpose of solving the MA equation is only a means of producing a mesh on which other equations can be solved. Providing that the mesh is as regular as is required by the underlying equation solver we may use a method which is cheap and provides only an approximate solution [Browne et al., 2014], or a solution to an approximate equation.

[Weller et al., 2015] pose the problem of mesh redistribution as finding a map $\mathbf{F} : \xi \to \xi + \nabla\phi$, rather than $\mathbf{F} : \xi \to \nabla P$ as stated in Equation 4. The two formulations of the problems are seen to be equivalent if we define

$$\begin{aligned} \phi(\xi) &:= P(\xi) - \frac{\xi^2}{2} \\ \Rightarrow \nabla P(\xi) &= \xi + \nabla\phi(\xi) \end{aligned}$$

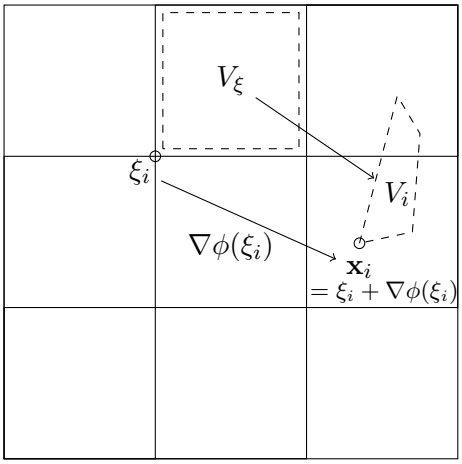

Figure 1.2: An illustration of an area or volume V_ξ being mapped to a volume V_i when points in the computational space Ω_C are translated via $\xi \mapsto \xi + \nabla\phi(\xi)$ for a convex potential ϕ defined on Ω_C.

That is, rather than finding the mesh potential P we are finding the displacement between it and the known function $\xi^2/2$. An illustration is shown in Figure 1.2. The mesh shown in Figure 1.1 (right) is the result of applying the map $\xi \mapsto \xi + \nabla\phi(\xi)$ to the mesh points of the uniform mesh shown on the left of the figure. The scalar ϕ used in this specific case is shown in Figure 1.3 and was found using the AL method defined in Section 2.5, the arrows on the contour plot show the direction and magnitude[1] of the gradient $\nabla\phi$.

With this different formulation of the problem, Equation 5 becomes

$$|J(\xi)| = \det(\nabla\mathbf{x}) = |\nabla(\xi + \nabla\phi(\xi))| = |I + \nabla^2\phi(\xi))| = |I + H(\phi)| \tag{7}$$

And the Monge-Ampère equation 6 thus becomes

$$M(\mathbf{x})|I + H(\phi)| = c \text{ (const.)} \tag{8}$$

[Weller et al., 2015] note that the presence of the identity tensor in this alternative MA equation allows us to expand the Hessian term as

$$|I + H(\phi)| = 1 + \nabla^2\phi + \mathcal{N}(\phi), \tag{9}$$

where $\mathcal{N}(\phi)$ collects all non-linear terms in ϕ. This is a useful step towards linearising the MA equation. [Browne et al., 2014] also note that considering the difference between the potential and the function $\xi^2/2$, as we have done here, rather that the potential itself, is convenient when solving the Parabolic Monge-Ampère (PMA) equation with periodic boundary conditions. In this report we seek to find numerical solutions to the MA equation 8.

[1] as the relative length of the arrows

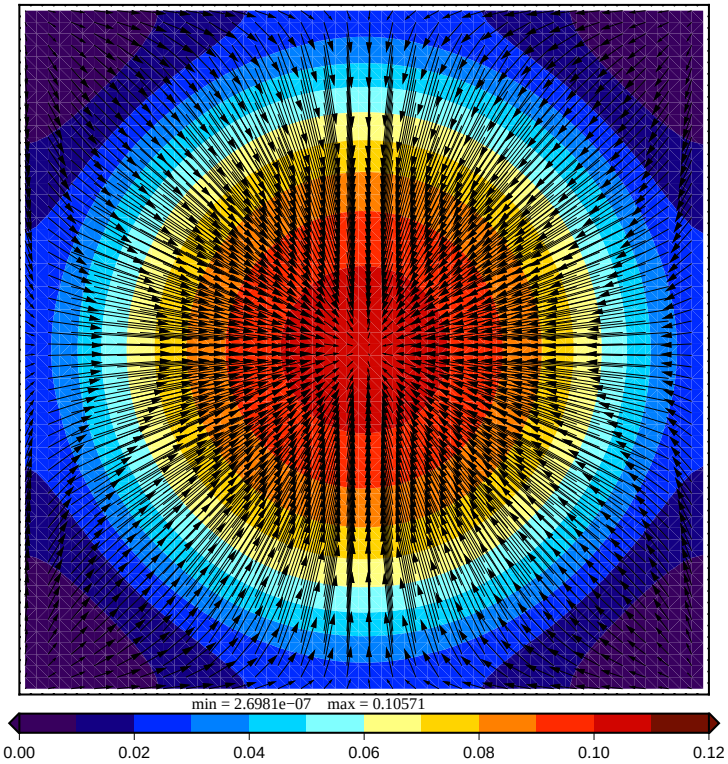

Figure 1.3: The scalar field ϕ used to create the mesh shown in Figure 1.1 (right) via the map $\xi \mapsto \xi + \nabla\phi(\xi)$. Found using the AL method (see section 2.5).

1.3.1 The Parabolic Monge-Ampère Equation (PMA)

[Budd and Williams, 2009] introduce a parabolic relaxation of the MA equation of the form

$$\varepsilon(I - \gamma\nabla^2)Q_t = (M(\nabla Q)|H(Q)|)^{\frac{1}{d}} \tag{10}$$

where Q_t is the derivative of Q with respect to t; H and M are the Hessian and the Monitor function, as elsewhere in this report; and d is the dimension of the space (in the implementation of this method $d = 2$ since we are working in 2D). The gradient $\nabla Q(t)$ evolves towards the gradient ∇P as $t \to \infty$, where P is the solution to the MA equation 6. The t in the equation is an introduced *pseudo-time* variable, not the global time over which a dynamic mesh changes. Throughout this report we consider solutions to the static mesh problem where Q evolves towards a solution as t increases in order to equidistribute a static monitor function M. To solve the dynamic mesh problem this procedure would be repeated at discrete global-time steps, each time using a monitor function specific to that time, with the effect that the monitor function, and hence the mesh, change as the global-time variable increases. Instead of Q, [Browne et al., 2014] consider the difference between Q and the function $|\xi|^2/2$, given by

$$\phi = Q - \frac{1}{2}|\xi|^2. \tag{11}$$

This gives

$$\mathbf{x} = \nabla Q = \nabla\phi + \xi \tag{12}$$

which allows us to rewrite the PMA equation as

$$\varepsilon(I - \gamma\nabla^2)\phi_t = (M(\mathbf{x})|I + H(\phi)|)^{\frac{1}{d}} \tag{13}$$

as in [Browne et al., 2014]. The power $^1/_d$ is included so that the RHS scales linearly with Q to ensure that a solution exists [Budd and Williams, 2009] and $\varepsilon(I - \gamma\nabla^2)$ is a smoothing operator which reduces stiffness the discretised system [Budd et al., 2009].

1.3.2 A fixed-point iterative method

A fixed-point iterative method aims to give an equation of the form

$$\phi^{n+1} = f(\phi^n) \tag{14}$$

which will give a sequence $\{\phi^0, \phi^1, ...\}$ where ϕ^n converges towards a fixed point of f. We hope to provide an f such that the solution ϕ of the MA equation $|I + H(\phi)| = c/M$ is a fixed point. [Weller et al., 2015] introduce a Fixed-Point (FP) method for solving the MA equation: the Hessian term $|I+H(\phi)|$ is linearised to find a Poisson equation which can be solved at each iteration. Here we restate Equation 9:

$$|I + H(\phi)| = 1 + \nabla^2\phi + \mathcal{N}(\phi). \tag{9, copy}$$

Note that in 2D we have $\mathcal{N}(\phi) = |H(\phi)|$. The iterative method used by [Weller et al., 2015] is then based on the approximation

$$\mathcal{N}(\phi^{n+1}) \approx \mathcal{N}(\phi^n) = |I + H(\phi^n)| - (1 + \nabla^2\phi^n). \tag{15}$$

That is, at each iteration we approximate the non-linear terms in the MA equation by the non-linear terms at the previous iteration. Thus we have the following approxination for $|I + H(\phi^{n+1})|$:

$$\begin{aligned} |I + H(\phi^{n+1})| &= 1 + \nabla^2\phi^{n+1} + \mathcal{N}(\phi^{n+1}) \\ &\approx 1 + \nabla^2\phi^{n+1} + \left[|I + H(\phi^n)| - (1 + \nabla^2\phi^n)\right]. \end{aligned}$$

Substituting this into the MA equation

$$|I + H(\phi^{n+1})| = \frac{c^{n+1}}{M^{n+1}} \tag{16}$$

gives

$$\begin{aligned} \frac{c^{n+1}}{M^{n+1}} &= 1 + \nabla^2\phi^{n+1} + \left[|I + H(\phi^n)| - (1 + \nabla^2\phi^n)\right] \\ \Rightarrow \quad 1 + \nabla^2\phi^{n+1} &= (1 + \nabla^2\phi^n) - |I + H(\phi^n)| + \frac{c^{n+1}}{M^{n+1}}. \end{aligned}$$

We then make the further approximation:

$$\frac{c^{n+1}}{M^{n+1}} \approx \frac{c^n}{M^n} \tag{17}$$

And so at each iteration we are solving the equation

$$\nabla^2\phi^{n+1} = \nabla^2\phi^n - |I + H(\phi^n)| + \frac{c^n}{M^n}, \tag{18}$$

which we refer to as the FP equation. The solution to the MA equation is a fixed point of this iterative sequence. If ϕ is a fixed point then

$$\nabla^2\phi = \nabla^2\phi - |I + H(\phi)| + \frac{c}{M} \tag{19}$$

$$\Rightarrow \quad |I + H(\phi)| = \frac{c}{M} \tag{20}$$

Following [Weller et al., 2015], the constant (in space) c^n is calculated retrospectively at the $(n+1)^{th}$ iteration in order that $\int_{\Omega_C} \text{RHS} = 0$, which insures a Poisson equation. We thus require that

$$\begin{aligned} \int_{\Omega_C} \left[\nabla^2\phi^n - |I + H(\phi^n)| + \frac{c^n}{M^n}\right] d\xi &= 0 \\ \Leftrightarrow \int \left[\nabla^2\phi^n - |I + H(\phi^n)|\right] d\xi + c^n \int \frac{1}{M^n} d\xi &= 0 \end{aligned}$$

since c is constant over Ω_C. We also note that since $\nabla^2\phi^n$ is the LHS of equation 19 at the previous iteration, $\int_{\Omega_C} \nabla^2\phi^n = 0$. Rearranging then gives

$$c^n = \frac{\int |I + H(\phi^n)|}{\int \frac{1}{M^n}} \tag{21}$$

The use of this constant c^n makes Equation 19 a Poisson equation. [Weller et al., 2015] use a form of under-relaxation by introducing a constant γ thus:

$$\gamma\nabla^2\phi^{n+1} = \gamma\nabla^2\phi^n - |I + H(\phi^n)| + \frac{c^n}{M^n} \tag{22}$$

which will have no effect on the converged solution but will change the convergence of the FP iterations. A larger value of γ will slow down convergence by decreasing the effect of the source term which is where the dependence on the monitor function comes from. In this case the mesh points will move more slowly towards a converged solution mesh, this has the effect of improving stability of the iterative method by reducing mesh tangling [Weller et al., 2015].

1.3.3 An alternative iterative method

[Benamou et al., 2010] give an iterative solution based on the Poisson equation. They define the operator T:

$$T(\phi) = (\nabla^2)^{-1}\left(\sqrt{(\Delta\phi)^2 + 2\left(\frac{c}{M(X)} - |H(\phi)|\right)}\right) \tag{23}$$

where $(\nabla^2)^{-1}$ is the inverse of the Laplacian operator ∇^2, and prove that if ϕ is convex and satisfies the Monge-Ampère equation

$$|H(\phi)| = \frac{c}{M(X)} \tag{24}$$

then it is a fixed point of T. Their method then consists of iterating $\phi^{n+1} = T(\phi^n)$ by solving:

$$\Delta\phi^{n+1} = \sqrt{(\Delta\phi^n)^2 + 2\left(\frac{c^n}{M(X^n)} - |H(\phi^n)|\right)} \tag{25}$$

1.4 Solutions and convergence

The methods described in this report require the solution for ϕ^{n+1}, at each iteration, of an equation involving a Laplacian operator acting on ϕ^{n+1}. This is done using a Conjugate Gradient (CG) solver with a diagonal incomplete Choleski preconditioner, as used in [Weller et al., 2015]. If the field ϕ^{n+1} is discretised over Ω_C, giving the matrix Φ, then a matrix equation $A\Phi = b$ and the CG solver solves it iteratively producing a sequence $\{\Phi_0, \Phi_1, ...\}$. The residual is defined by:

$$i^{\text{th}} \text{ residual } \coloneqq \frac{\sum |b - A\Phi_i|}{\sum |b| - |A\Phi_i|}, \tag{26}$$

where the sum is over all cells in the mesh. The CG solver stops when the i^{th} residual falls below a specified tolerance. Following [Weller et al., 2015] we set this to be 10^{-6}.

The FP method is judged to be converged when the number of iterations taken by the CG solver is ≤ 1, effectively when the *initial* (or 0^{th}) residual is $\approx 10^{-6}$. The value of the initial residual is thus a useful measure of convergence.

The *equidistribution* $M(\mathbf{x})|I + H(\phi)|$ of the mesh also measures convergence. When ϕ is converged to a solution of the MA equation we will have

$$M(\mathbf{x})|I + H(\phi)| = c \text{ (const.)}.$$

The spatial variation in the equidistribution is therefore a measure of convergence. We consider the *equidistribution error* at each iteration of the algorithm:

$$\text{equidistribution error } \coloneqq \frac{\sqrt{\text{Var}(M(\mathbf{x})|I + H(\phi)|)}}{\text{mean}(M(\mathbf{x})|I + H(\phi)|)} \tag{27}$$

as described in [Browne et al., 2014].

1.5 Motivation for research

There exist many methods which aim to obtain numerical solutions to the MA equation for use in r-adaptive mesh generation, including the three described in

the previous section. In this report we implement the Fixed-Point (FP) iterative method taken from [Weller et al., 2015] and the relaxed Parabolic MA equation (PMA) described in [Budd et al., 2009, Browne et al., 2014], both in 2D. Both of these methods contain a relaxation constant γ which relaxes the system to improve stability. In [Browne et al., 2014] and [Weller et al., 2015] the value of γ is found by experimentation which may be unsuitable for applications where the value may need to be adjusted at each time step (as the monitor function changes for example). In [Weller et al., 2015] the value of γ is set to be 'sufficiently' large so that the FP method will work in most cases, but it is also noted that increasing the value of γ - beyond what is necessary to ensure stability - slows down the algorithm. The PMA-based algorithm used by [Browne et al., 2014] also includes smoothing the monitor function, which increases the speed of the method and the quality of the mesh.

[Weller et al., 2015] note that the PMA and FP methods are similar yet have a few significant differences, they also note that there "may be further scope for improvement by combining the best aspects of the two approaches".

These observations motivate this project, which aims to examine the role of the constant γ in both the PMA and FP methods, and to examine the differences between the two methods. We are also motivated to find a method of mesh generation through solutions to the Monge-Ampère equation which remains stable without the need to fine-tune any arbitrary constants, and achieves fast convergence for any monitor function without the need for smoothing.

1.6 Outline

In Sections 2.2 and 2.3 we implement the PMA and FP methods described in [Browne et al., 2014] and [Weller et al., 2015] respectively. In Section 2.4 we implement some variations on the PMA method which seek to combine aspects of the PMA and FP methods. We also, in Section 2.5 derive an alternative linearisation of the Hessian term $|I + H(\phi)|$ to that used in the FP method, which leads to a new iterative method that we call the AL (alternate linearisation) method.

We analyse the results from the implementation of these methods and draw conclusions. We then set out a plan for future work.

Chapter 2

Exploration of Numerical Techniques

2.1 Spatial discretisation

In this chapter we describe three algorithms for solving the set of non-linear algebraic equations resulting from the spatial discretisation of the MA equation. [Weller et al., 2015] references a number of possible spatial discretisations; in this report we use the finite volume discretisation used by [Weller et al., 2015]. We also follow [Weller et al., 2015], [Browne et al., 2014] and [Budd et al., 2009] in using periodic boundary conditions. We use the same spatial discretisation and boundary conditions for all three methods so that the results are consistent and fair comparisons can be drawn.

2.2 The Parabolic Monge-Ampère Equation (PMA)

It is noted in [Budd and Williams, 2009] that the constant c in the MA equation is not present in the PMA equation since the constant will 'arise naturally as a constant of integration'. The addition or subtraction of a constant does not affect the mesh redistribution since only the *gradient* of the mesh potential ϕ is important. In this report, however, we wish to compare the PMA method to the other methods described in this chapter and use the same convergence criteria; we thus seek to bound the solution ϕ by considering instead a PMA equation of the form

$$\varepsilon(I - \gamma\nabla^2)\phi_t = (M(\mathbf{x})|I + H(\phi)|)^{\frac{1}{d}} - c \tag{28}$$

which will not affect the value of $\nabla\phi$. The constant c is calculated such that

$$\int_{\Omega_C} \left[(M(\mathbf{x})|I + H(\phi)|)^{\frac{1}{d}} - c\right] d\xi = 0 \tag{29}$$

which gives

$$c = \frac{\int_{\Omega_C}(M(\mathbf{x})|I + H(\phi)|)^{\frac{1}{d}} d\xi}{\int_{\Omega_C} 1 d\xi}. \tag{30}$$

In order to implement this method as a algorithm, the forward-Euler time-discretisation is used:

$$\frac{d}{dt}\phi^n = \frac{\phi^{n+1} - \phi^n}{\delta t}. \tag{31}$$

We have used ϕ^{n+1} rather than $\phi(t+\delta t)$ to make the notation consistent with the other methods described in this chapter. The PMA Equation 28 then becomes

$$\varepsilon(\phi^{n+1} - \phi^n) - \varepsilon(\gamma\nabla^2\phi^{n+1} - \gamma\nabla^2\phi^n) = (M^n|I + H(\phi^n)|)^{\frac{1}{d}} - c^n \tag{32}$$

where the factor $1/\delta t$ has been incorporated into the relaxation factor ε, M^n is the Monitor function $M(\mathbf{x}^n)$ where $\mathbf{x}^n = \xi + \nabla\phi^n(\xi)$ are the physical coordinates at the n^{th} iteration. This rearranges to give the iterative method:

$$\phi^{n+1} - \gamma\nabla^2\phi^{n+1} = \phi^n - \gamma\nabla^2\phi^n + \frac{1}{\varepsilon}\left[(M^n|I + H(\phi^n)|)^{\frac{1}{d}} - c^n\right] \tag{33}$$

We can thus implement the PMA method as described in Algorithm 3.

Algorithm 3 PMA Method

Set $\phi = 0$
while not converged **do**
  Set $c^n \leftarrow \int_{\Omega_C} |I + H(\phi^n)|M^n d\xi \;/\; \int_{\Omega_C} 1 d\xi$ as in equation 30
  Solve equation 33 to find ϕ^{n+1}

  Calculate $\nabla\phi^{n+1}$ and create the new mesh
  Calculate $|I + H(\phi^{n+1})|$ and M^{n+1}
  $n \leftarrow n + 1$
end while

2.3 An existing fixed-point iterative method

In 1.3.2 we describe a fixed-point iterative method which makes use of Equation 22. This FP method, taken from [Weller et al., 2015], is implemented in this report as described in Algorithm 4.

Algorithm 4 FP Method

Set $\phi = 0$
while not converged **do**
  Set c^n as in equation 21
  Solve equation 22 to find ϕ^{n+1}

  Calculate $\nabla\phi^{n+1}$ and create the new mesh
  Calculate $|I + H(\phi^{n+1})|$ and M^{n+1}
  $n \leftarrow n + 1$
end while

2.4 Combining FP and PMA

In this project we aim to examine the differences between the PMA and FP methods. We note that there are essentially three differences between the two methods, i.e. between Equation 32 and Equation 22:

1. In the PMA equation (32) the term $M^n|I + H(\phi^n)$ is taken to the power of $1/d$ so that the Hessian scales linearly with ϕ. Without taking this power, the equation admits separable solutions which blow up [Budd and Williams, 2009].
2. The PMA equation also has the term $\varepsilon(\phi^{n+1} - \phi^n)$ in addition to the Laplacian term in the FP equation (22).
3. In the PMA the monitor function M^n multiplies the Hessian term $|I+H(\phi^n)|$ whereas in the FP equation, M^n divides the constant term c^n.

In this report we experiment with three variations on the PMA or FP equation, these variations combine aspects of both the PMA and FP equations. It is hoped that the results produced by implementing these experiments may lead to a better understanding of the differences between the PMA and FP methods.

We experiment with a variation of the FP method in which we include the power $1/d$:

$$\gamma\nabla^2\phi^{n+1} = \gamma\nabla^2\phi^n - (|I + H(\phi^n)|)^{1/d} + \frac{c^n}{(M^n)^{1/d}}, \tag{34}$$

which we call Experiment B. The justification for the power law, used by [Budd and Williams, 2009], applies, in theory, to the FP equation as well since if the mesh potential $P = \phi + |\xi|^2/2$ is scaled by a factor α, then $|I + H(\phi)|$ will scale as α^2 (in 2D).

We also consider a variation on the PMA equation where the $\varepsilon(\phi^{n+1} - \phi^n)$ term is removed:

$$\gamma\nabla^2\phi^{n+1} = \gamma\nabla^2\phi^n - (M^n|I + H(\phi^n)|)^{1/d} + c^n \tag{35}$$

This equation we call Experiment D. If we neglect the power $1/d$ in this equation we get

$$\gamma\nabla^2\phi^{n+1} = \gamma\nabla^2\phi^n - M^n|I + H(\phi^n)| + c^n \tag{36}$$

which is very similar to the FP equation, in fact the only difference is the position of the monitor function. We call this equation Experiment F. Despite the different position of the term M^n, we note that a solution of the MA equation is a fixed point of Equation 36.

These three variations of the FP and PMA equations are implemented in the same way as the PMA and FP methods with the constant c being found such that the integral over Ω_C of the LHS of each equation is equal to zero.

2.5 An alternative linearisation of the Hessian

In this report we introduce an alternative linearistion of $|I + H|$ to that in [Weller et al., 2015] which results in the FP method. This is motivated by the approximation made in equation 15 which may be unreasonable if the non-linear terms in ϕ and its derivatives become very large. Instead of solving an equation for ϕ we consider solving an equation for ψ defined as the difference between ϕ and some initial guess $\bar{\phi}$. If our guess is close to ϕ then ψ will be very small. Making the substitution $\phi = \bar{\phi} + \psi$ gives the following expansion of the term $|I + H(\phi)|$:

$$|I + H(\bar{\phi})| + |H(\psi)| + \left[\psi_{xx} + \psi_{yy} + \bar{\phi}_{xx}\psi_{yy} + \bar{\phi}_{yy}\psi_{xx} - 2\bar{\phi}_{yx}\psi_{xy}\right]. \tag{37}$$

If the bracketed part of this expression could be expressed in standard vector calculus notation there may already exist standard methods to solve the equation. It would also be much simpler notationally. The form suggests a laplacian $\nabla \cdot (A\nabla\psi)$, for some matrix A. If we say $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ then

$$\nabla \cdot (A\nabla\psi) = (a_x + c_y)\psi_x + (b_x + d_y)\psi_y + a\psi_{xx} + d\psi_{yy} + (b + c)\psi_{xy}. \tag{38}$$

If we equate the RHS of equation 38 to the bracketed term of equation 37 we get a system of equations for a, b, c and d whose solution gives

$$A = \begin{pmatrix} 1 + \bar{\phi}_{yy} & -\bar{\phi}_{xy} \\ -\bar{\phi}_{xy} & 1 + \bar{\phi}_{xx} \end{pmatrix} = \left[1 + \nabla^2\bar{\phi}\right] I - H(\bar{\phi}). \tag{39}$$

Note that we could alternatively write $A = |I + H(\bar{\phi})|(I + H(\bar{\phi}))^{-1}$ since the two expressions are equal (in 2D). The substitution $\phi = \bar{\phi} + \psi$ therefore gives us the expansion

$$|I + H(\phi)| = |I + H(\bar{\phi})| + |H(\psi)| + \nabla \cdot (A\nabla\psi). \tag{40}$$

We have found that the form of equation 40 does not generalise to 3D. Expanding the matrix $|I + H(\bar{\phi} + \psi)|$ in 3D gives

$$\begin{aligned} & |I + H(\bar{\phi})| + |H(\psi)| \\ + \;& (1 + \bar{\phi}_{ii})\psi_{jj}\psi_{kk} + \psi_{ii}(1 + \bar{\phi}_{jj})(1 + \bar{\phi}_{kk}) + 2\bar{\phi}_{ii}\psi_{jj}\psi_{kk} + 2\psi_{ii}\bar{\phi}_{jj}\bar{\phi}_{kk} \\ - \;& (1 + \bar{\phi}_{ii})(\psi_{jk} + 2\bar{\phi}_{jk})\psi_{jk} - \psi_{ii}\bar{\phi}_{jk}(2\psi_{jk} + \bar{\phi}_{jk}) \end{aligned}$$

summing over the permutations $(i, j, k) \in \{(x, y, z), (y, z, x), (z, x, y)\}$. The linear terms in this expression are

$$|I + H(\bar{\phi})| + \psi_{ii}[\bar{\phi}_{jj} + \bar{\phi}_{kk} + 3\bar{\phi}_{jj}\bar{\phi}_{kk} - \bar{\phi}_{jk}^2] - 2\psi_{jk}[\bar{\phi}_{jk} + \bar{\phi}_{ii}\bar{\phi}_{jk}]. \tag{41}$$

Equating the terms in ψ to the coefficients in $\nabla \cdot (B\nabla\psi)$, where B is the 3×3 matrix $[b_{ij}]$, $1 \leq i, j \leq 3$, we get a system of equations for $\{b_{ij}\}$ for which no solution exists. This is because we have the mixed derivative terms ψ_{jk} which do not appear in the Laplacian. It may be the case that the linear term in ψ can be expressed in some other standard way: the form suggests something similar to

$\nabla \cdot (B\nabla\psi) - 2\nabla \times (B\nabla\psi)$.

From the expansion in Equation 40 we hope to derive a fixed point iterative method to find $\{\phi^n\}$ which will converge to the solution ϕ of the MA equation. If, in the above working we make the replace $\bar{\phi}$ by ϕ^n and ϕ by ϕ^{n+1}, then we have an iterative equation for ϕ^{n+1} where ψ is required to advance the solution. This makes sense for small ψ because ϕ^n should be a reasonably good initial guess for ϕ^{n+1}. From equation 40 we have:

$$|I + H(\phi^n)| + |H(\psi)| + \nabla \cdot \left(A^n \nabla(\phi^{n+1} - \phi^n)\right) = \frac{c^{n+1}}{M^{n+1}} \tag{42}$$

where $A^n = \left[1 + \nabla^2\phi^n\right] I - H(\phi^n)$ as in Equation 39 and we have used the fact that $\psi = \phi^{n+1} - \phi^n$. Assuming $|H(\psi)| \approx 0$, which is similar to the approximation made in Equation 15, and making the approximation in Equation 17, we get the fixed point iterations:

$$\nabla \cdot \left(A^n \nabla\phi^{n+1}\right) = \nabla \cdot (A^n \nabla\phi^n) - |I + H(\phi^n)| + \frac{c^n}{M^n} \tag{43}$$

by the linearity of the Laplacian operator. We note that this is similar to the iterations defined in Equation 22 only the Laplacian operator $\gamma\nabla^2(\cdot)$ is replaced by $\nabla \cdot (A\nabla(\cdot))$. The Alternative Linearisation (AL) method is defined in the same way as the FP method in Algorithm 4 with the single difference that at each step we solve equation 43 to find ϕ^{n+1} rather than equation 22.

We note that the approximation $|H(\psi)| \approx 0$ is similar, but not equivalent to the approximation made in Equation 15. In deriving the Fixed Point method we made the approximation

$$\mathcal{N}(\phi^{n+1}) - \mathcal{N}(\phi^n) \;=\; \mathcal{N}(\phi^n + \psi) - \mathcal{N}(\phi^n) \;\approx\; 0 \tag{44}$$

where $\mathcal{N}(\phi)$ denotes the non-linear terms in $|I + H(\phi)|$, and $\phi^n + \psi = \phi^{n+1}$. In the Alternate Linearisation we have made the approximation

$$\mathcal{N}(\phi^{n+1} - \phi^n) \;=\; \mathcal{N}(\psi) \;\approx\; 0 \tag{45}$$

That is, in the FP method we have non-linear functions of ϕ^n, or its derivatives, which may become very large depending on the monitor function, whereas in the LA method we only have non-linear functions of the (small) difference ψ between successive values of ϕ^n. As an illustration we consider the simple non-linear function $\mathcal{N} : \phi \mapsto \phi^2$, then in the FP method we will have made the assumption

$$(\phi^n + \psi)^2 - (\phi^n)^2 \;=\; 2\phi^n\psi + \psi^2 \;\approx\; 0$$

whereas in the AL method we have $\psi^2 \approx 0$, which is a much more reasonable assumption given that ψ is always relatively small compared to ϕ^n, which is dependent on the case-specific monitor function.

As another illustration we consider the non-linear function $\mathcal{N} : \phi \mapsto |H(\phi)|$ which is relevant to our example. In the FP method we will have made the assumption

$$|H(\phi^n + \psi)| - |H(\phi^n)| \;=\; |H(\phi^n)| + |H(\psi)| + \phi^n_{xx}\psi_{yy} + \phi^n_{yy}\psi_{xx} - 2\phi^n_{yx}\psi_{xy} \;\approx\; 0$$

whereas in the AL method we have

$$|H([\phi^n + \psi] - \phi^n)| \;=\; |H(\psi)| \;\approx\; 0.$$

There is no guarantee that $|H(\psi)| = 0$ will be a good approximation since this will depend specifically on ψ. It is possible to ensure a small error by taking very small $\psi = \epsilon\bar{\psi}$ where $\epsilon \ll 1$, that is, the difference between successive values of ϕ^n is very small. Then we have

$$|H(\psi)| = \epsilon^2 |H(\bar{\psi})| \approx 0.$$

However, we note that although $|H(\psi)| = 0$ may not always be a good approximation, it is almost certain to be a better a better approximation than is made in the derivation of the FP method.

Chapter 3

Results

3.1 Choice of monitor function

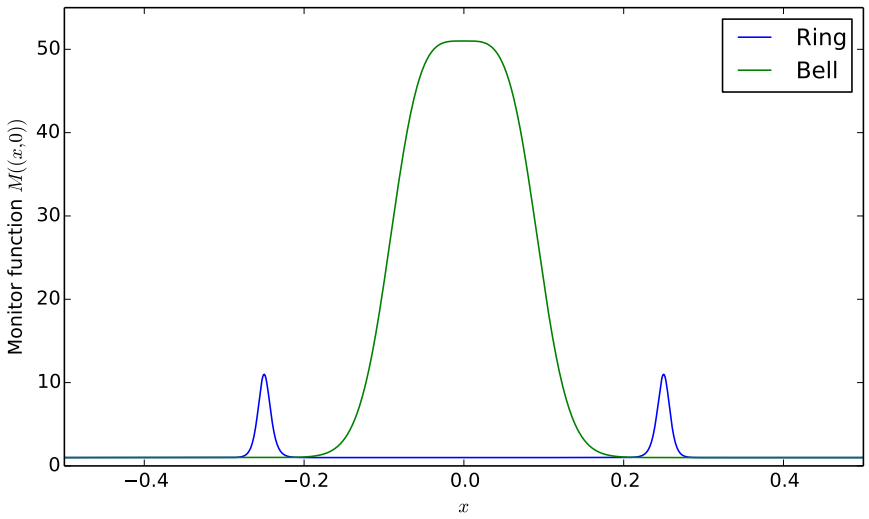

Figure 3.1: Both monitor functions shown in cross-section along the line $y = 0$. The ring takes maximal values at a distance 0.25 from the origin. The bell function takes a maximum value at the origin.

Meshes are generated to equidistribute a static monitor function in two dimensions on the domain $\left[-1/2, 1/2\right]^2$. In this report we follow [Budd et al., 2009] and [Weller et al., 2015] in using a radially symmetric monitor function of the form

$$M(\mathbf{x}) = 1 + \alpha_1 \text{sech}^2\left(\alpha_2\left(|\mathbf{x} - \mathbf{x}_c|^2 - a^2\right)\right) \tag{46}$$

where $\mathbf{x}_c$ is the centre of the region. In our examples the refined region is $\left[-1/2, 1/2\right]^2$ and $\mathbf{x}_c$ is the origin. Using this form we define two different monitor functions: the *ring* function, where $a = 0.25$, $\alpha_1 = 10$ and $\alpha_2 = 200$; and the *bell* function, where $a = 0$, $\alpha_1 = 50$ and $\alpha_2 = 100$. Both are radially symmetric

and cross-sections (along the line $y = 0$) of both functions are shown in Figure 3.1. We generate meshes equidistributing both monitor functions using the PMA, FP and AL methods in order to compare the convergence towards an optimally transported mesh.

3.2 Comparison of convergence criteria

We experimented with using the equidistribution error to define convergence in a similar way to the use of the initial residual, by setting an equidistribution error tolerance of 10^{-5}. However, with both monitor functions used in this report there was little difference in these two approaches. Figures 3.4 and 3.5 show the initial residuals and the equidistribution errors (resp.) at each iteration of the PMA, FP and AL methods when they are used to equidistribute the Ring function and the bell function. The patterns in both figures are very closely correlated, that is, examining the equidistribution errors gives little additional information over the initial residuals when using both monitor functions defined in Section 3.1 [Choice of monitor function]. Throughout this chapter we use the definition of convergence given in Section 1.4 [Solutions and convergence]: when the number of iterations of the CG solver is ≤ 1 (or, when the initial residual is $\approx 10^{-6}$).

3.3 Comparison varying γ in PMA and FP

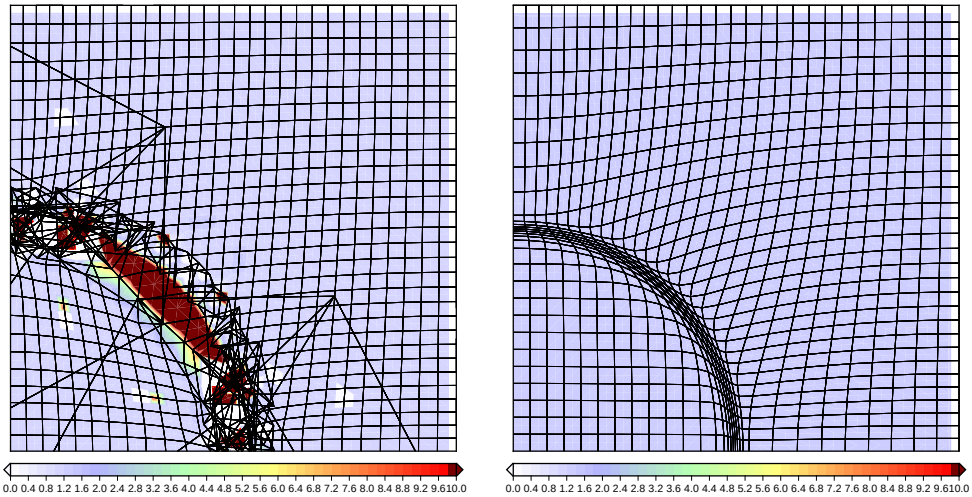

Figure 3.2: The first quarter of the bell mesh using the FP method with $\gamma = 0.5$ (left) and $\gamma = 1$ (right). The colour contours show the value of the equidistribution. With $\gamma = 0.5$ the mesh tangled after 8 iterations whereas with $\gamma = 1$ the method converged after 36 iterations (shown).

The PMA and FP methods were implemented with the ring monitor function and the value of the parameter γ was varied to investigate the effect on the time taken for the mesh to converge to the solution. In the FP method it is expected that a higher value of γ will result in slower convergence but be more stable. Figure 3.2 shows the result of using the FP method with the ring monitor function.

With $\gamma = 0.5$ (left) mesh started to tangle after 8 iterations, increasing the value of γ to 1 made the method stable, however, it took longer to converge than the same method with $\gamma = 0.9$ (36 compared to 34 iterations). For these experiments using the ring monitor function we found that $\gamma < 0.78$ resulted in mesh tangling.

The PMA equation has an additional parameter ε equivalent to $1/\delta t$ which comes from the fact that this is a time-stepping method (see Equation 33). The PMA method was implemented with $\varepsilon \in \{0.625, 1.25, 2.5, 10\}$, or equivalently $\delta t \in \{1.6, 0.8, 0.4, 0.1\}$.

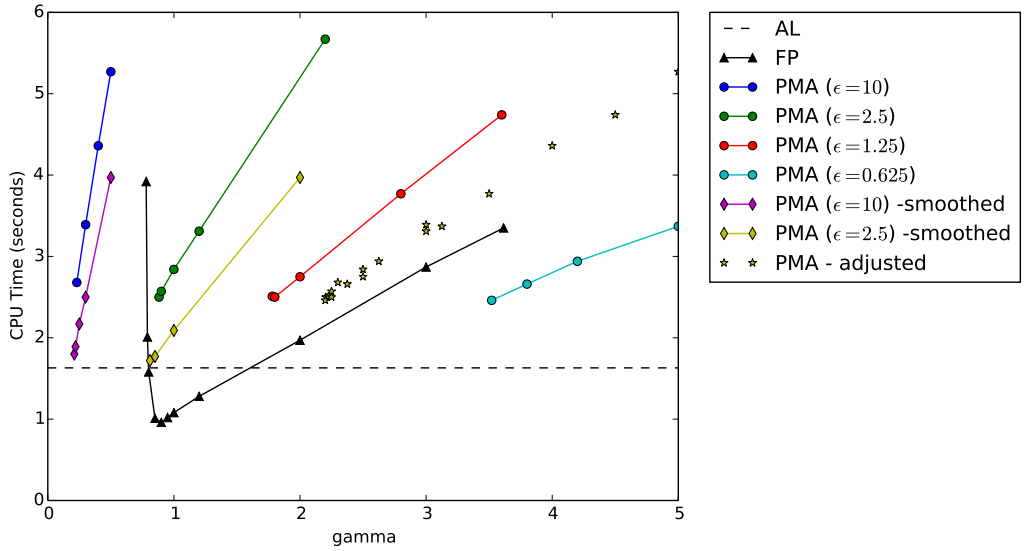

Figure 3.3: Time taken (in seconds) for the PMA and FP methods to converge using the ring monitor function with varying values of γ. The AL method, with no free parameters, is represented by the dashed line for comparison purposes.

Figure 3.3 shows a comparison of the times taken to reach a converged solution using the different methods. The lowest value of γ used in each case is the lowest one such that the algorithm is still stable. The largest γ in each case is chosen to give a good range of values which represent what is happening. As expected, a larger value of γ resulted in an algorithm which took longer to converge, with the exception of the FP method with very low values of γ. For $0.78 \leq \gamma \leq 0.9$ the FP method remained stable but was apparently not optimal. Except for these low values of γ in the FP method, the CPU time appears to correlate linearly with γ. Further experiments were done (not shown in figure 3.3) where much larger values of γ were used: where the algorithm still converged in ≤ 1000 iterations, the times taken suggested the same linear relationships with the value of γ as are suggested by Figure 3.3.

This linear relationship between CPU time and γ (for the PMA method) suggests that the algorithm could converge faster if it were possible to decrease γ without losing stability. In [Browne et al., 2014] the PMA method is implemented

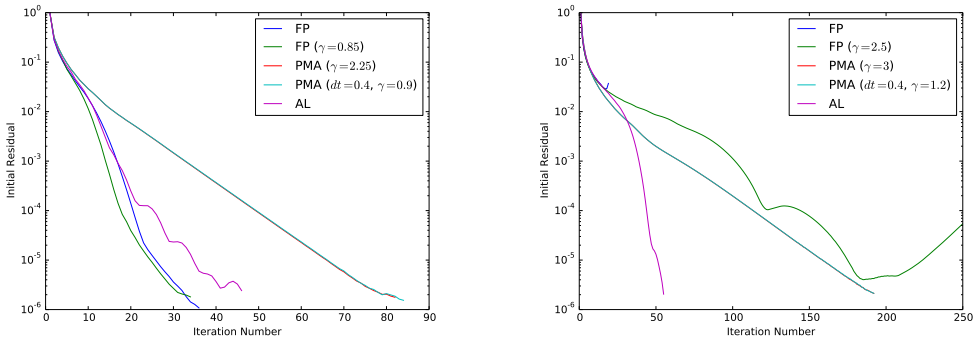

Figure 3.4: Initial Residuals using the Ring (left) and the Bell (right). Note that the lines for the two PMA methods (with different parameters) overlap almost perfectly and are therefore difficult to distinguish.

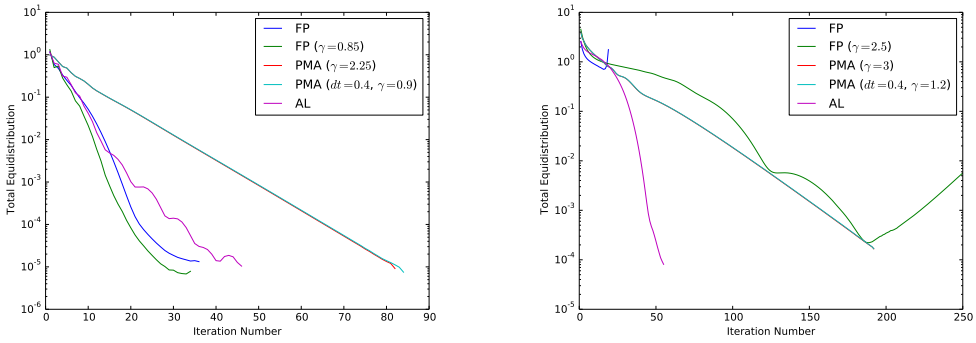

Figure 3.5: Total Equidistribution using the Ring (left) and the Bell (right). Note that the lines for the two PMA methods (with different parameters) overlap almost perfectly and are therefore difficult to distinguish.

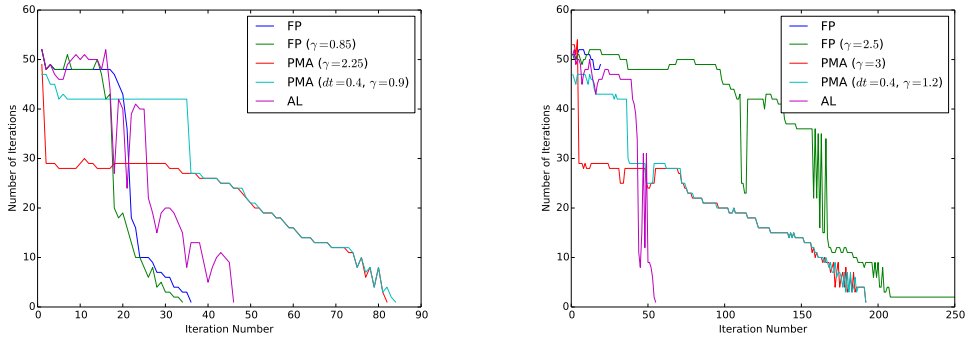

Figure 3.6: Number of iterations taken by the CG solver at each iteration of the algorithm. Using the Ring (left) and the Bell (right).

in using smoothing of the monitor function to speed convergence. This technique was replicated for $\varepsilon = 10$ and $\varepsilon = 2.5$. As illustrated in Figure 3.3, smoothing did speed convergence but did not allow the value of γ to be brought any lower than is possible without smoothing. Following [Weller et al., 2015], smoothing of the monitor function is not considered any further in this report since the aim is to provide a solution of the MA equation for any monitor function.

3.4 Effect of varying ε in PMA

Figure 3.3 shows the PMA method implemented using four different values of ε. The times taken by the different implementations of the PMA method suggest that there is nothing to be gained from varying ε. If, given ε, we had a method to determine the 'best' value of γ (that which results in the fastest possible stable method) the time taken would be roughly 2.5 seconds independent of ε. The results of all four implementations of the PMA method were adjusted by scaling γ by a factor of ε, these values are also plotted in Figure 3.3. The correlation of the adjusted results suggests that only the value of $\gamma\varepsilon$ affects the speed of the algorithm, in which case we can set $\varepsilon = 1$ and vary γ only.

Two different PMA methods were implemented for each monitor function by changing the parameters γ and ε. For the ring function we used ($\gamma = 0.9$, $\varepsilon = 2.5$) and ($\gamma = 2.25$, $\varepsilon = 1$) in both cases we have $\varepsilon\gamma = 2.25$. For the bell function we used ($\gamma = 1.2$, $\varepsilon = 2.5$) and ($\gamma = 3$, $\varepsilon = 1$) in both cases we have $\varepsilon\gamma = 3$. Figures 3.4 and 3.5 show the initial residuals and the equidistribution errors over number of iterations. The figures show the two lines for the two PMA methods very close together, in fact over-lapping. These observations support the suggestion that only $\varepsilon\gamma$ affects the algorithm. Considering the PMA equation we are solving:

$$\varepsilon(\phi^{n+1} - \phi^n) - \varepsilon\gamma(\nabla^2\phi^{n+1} - \nabla^2\phi^n) = (M^n|I + H(\phi^n)|)^{\frac{1}{d}} - c^n \tag{47}$$

we see that if $\gamma\varepsilon$ is the only parameter to affect the solution then the

$$\varepsilon(\phi^{n+1} - \phi^n) = \frac{d}{dt}\phi^n \tag{48}$$

term must have no effect. However, Figure 3.6 shows that, although the residuals and equidistribution error of the two PMA methods $\varepsilon = 2.5$; and $\varepsilon = 1$ show the same behaviour, the numbers of CG solver iterations do not. Using $\varepsilon = 2.5$: the algorithm involves more difficult matrix equations for the first 35 to 40 outer iterations. The PMA method with a larger ε takes ≈ 40 CG solver iterations per iteration for at the start of the algorithm, compared to ≈ 30 iterations for the smaller ε. The parameters with larger ε give more weight to the $\varepsilon(\phi^{n+1} - \phi^n)$ term relative to the Laplacian term in the PMA equation. Although this requires more CG solver iterations in total, the overall speeds (measured in CPU seconds) of both algorithms are very similar. Using $\varepsilon = 2.5$ the algorithm converged in 2.64s on for the Ring and 5.89s for the bell, compared to 2.56s and 5.8s respectively for the $\varepsilon = 1$ implementations: hence the correlation of the 'adjusted' PMA results in Figure 3.3. Figure 3.7 shows that the meshes produced by these two sets of parameters are almost identical after 10 iterations.

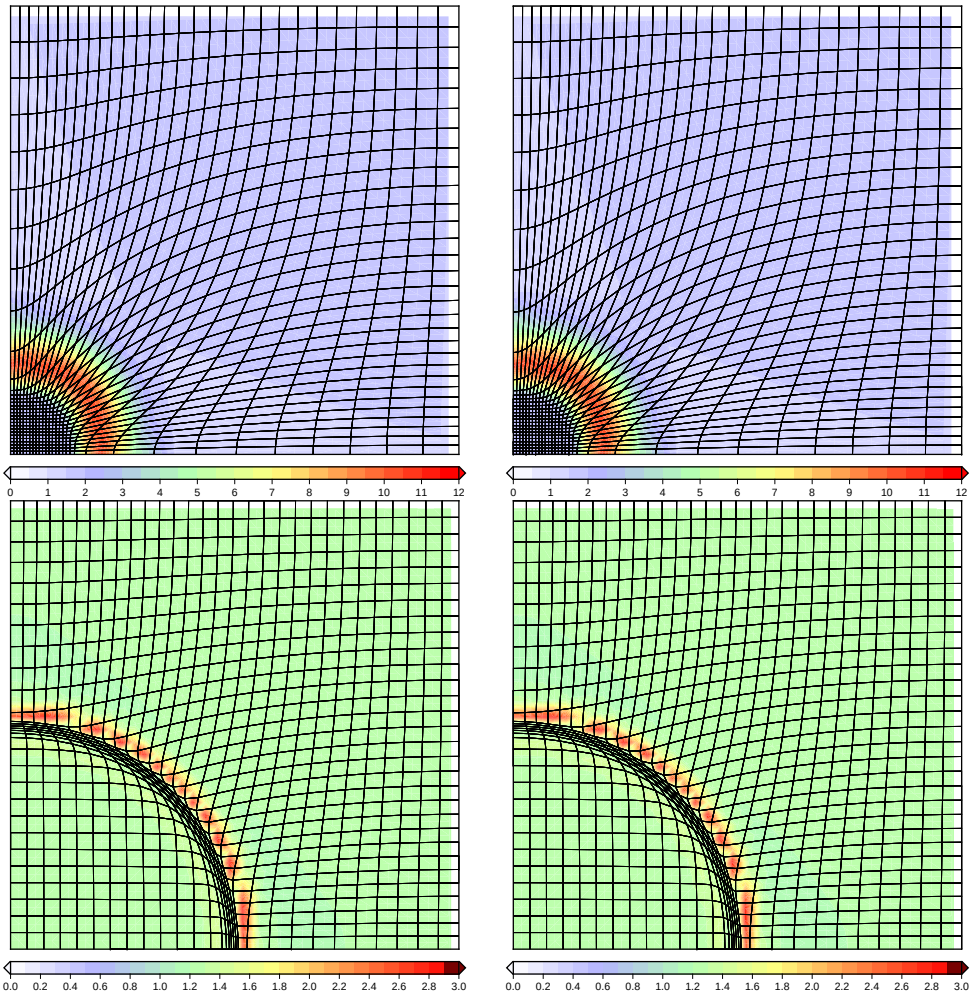

Figure 3.7: The first quarter of the Bell (top) and Ring (bottom) meshes after ten iterations of the PMA method using $\varepsilon = 2.5$ (left) and $\varepsilon = 1$ (right). The contour plot is of the value of the equidistribution.

3.5 Experiments in changing the PMA method

3.5.1 Experiment D

[Weller et al., 2015] notes that there may be scope for improvement by combining aspects of the FP and PMA methods. We have already seen that the first term in equation 47 does not appear to have an affect on the convergence of the PMA algorithm. In fact using a larger value of ε, which gives more weight to this term, actually increases the number of CG solver iterations required (see Figure 3.6). It therefore seems reasonable to experiment with an alternative PMA equation where this term is omitted. This Experiment D is described by Equation 35:

$$\gamma\nabla^2\phi^{n+1} = \gamma\nabla^2\phi^n - (M^n|I + H(\phi^n)|)^{1/d} + c^n. \qquad \text{(35, copy)}$$

The equation could be derived from an alternative statement of the PMA equation (Equation 10) where the identity matrix I is omitted from the smoothing operator

$\varepsilon\left(I - \gamma\nabla^2\right)$.

Figure 3.9 shows the initial residuals in the PMA and Experiment D methods overlapping entirely, both using the Ring function and the bell function. Figure 3.10 shows the same for the equidistribution errors, and Figure 3.11 shows the same for the number of iterations of the CG solver at each outer iteration of the algorithm. Using the Ring function, the value of γ is set to 2.25 in both the PMA and Experiment D methods as in both cases this was the lowest value of γ that resulted in a stable solution. Using the bell function both methods require $\gamma \geq 3$ to remain stable. Analysis of the initial residuals, equidistribution errors and the CG solver iterations suggests no difference between Experiment D and the PMA method, using either monitor function.

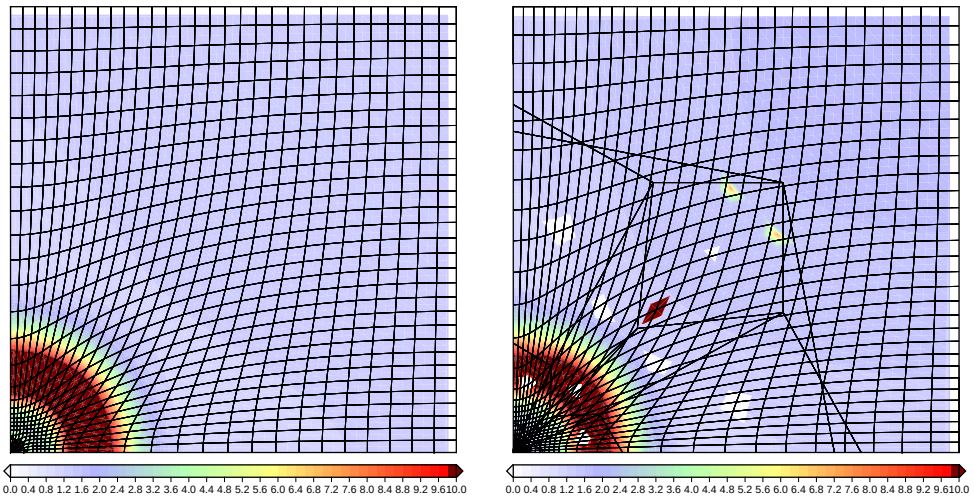

Figure 3.8: The first quarter of the bell mesh using Experiment F with $\gamma = 19$. After 4 iterations (left) and 8 iterations (right). The colour contours show the value of the equidistribution.

3.5.2 Experiment F

The Experiment F method uses Equation 36:

$$\gamma\nabla^2\phi^{n+1} = \gamma\nabla^2\phi^n - M^n|I + H(\phi^n)| + c^n, \qquad \text{(36, copy)}$$

which is the same as the FP equation except for the position of the monitor function (M^n) which multiplies the Hessian term rather than dividing the constant term. We note in Section 2.4 [Combining FP and PMA] that a solution of the MA equation is a fixed point of the Experiment F equation. Although we have described the Experiment F equation as a variation of the FP equation, we note that it is also essentially a variation of the PMA equation. If we conclude from our analysis of Experiment D that the $\varepsilon(\phi^{n+1} - \phi^n)$ term in the PMA equation dose not affect the algorithm or the solutions, then when considering the PMA equation we may as well consider Equation 35 used in Experiment D. As noted

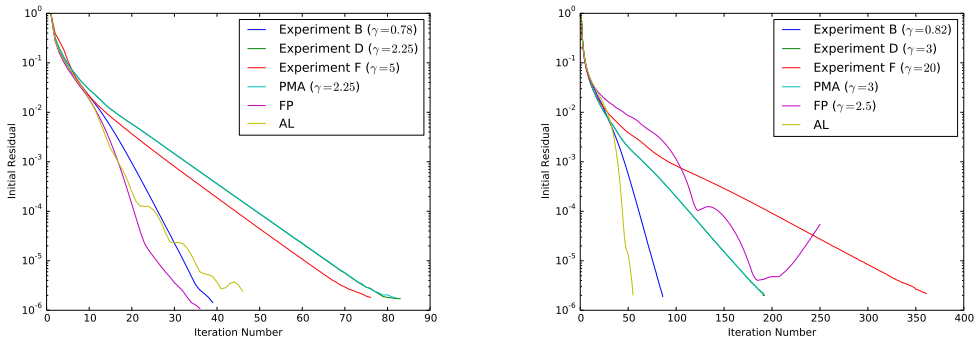

Figure 3.9: Initial Residuals using the Ring (left) and the Bell (right). Note that the lines for PMA and Experiment D overlap almost perfectly and are therefore difficult to distinguish.

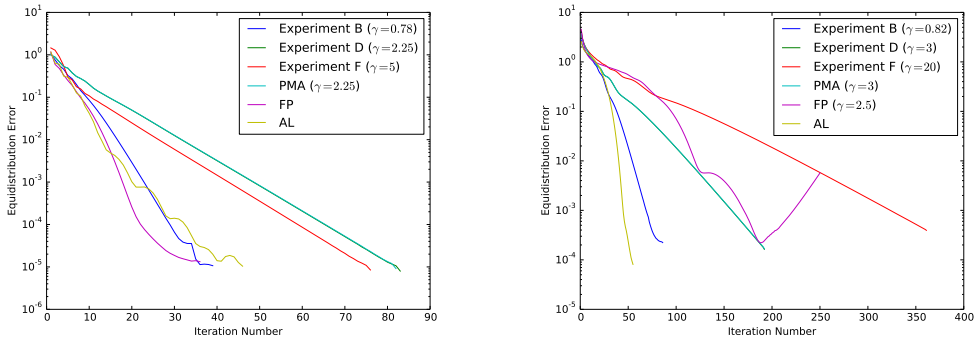

Figure 3.10: Total Equidistribution using the Ring (left) and the Bell (right). Note that the lines for PMA and Experiment D overlap almost perfectly and are therefore difficult to distinguish.

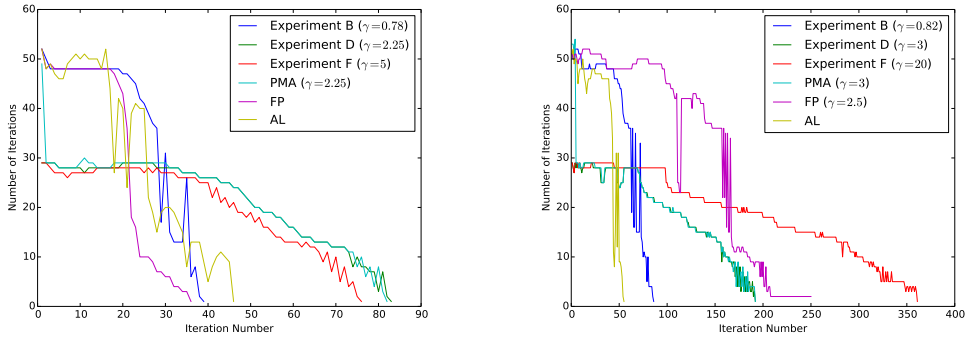

Figure 3.11: Number of iterations taken by the CG solver at each iteration of the algorithm. Using the Ring (left) and the Bell (right). Note that the lines for PMA and Experiment D overlap almost perfectly and are therefore difficult to distinguish.

in Section 2.4, the F equation is essentially the D equation without the $^1/_d$ power law (in the 2D case - a square root).

Figures 3.9, 3.10 and 3.11 (left) show that using the ring function, Experiment F performs similarly to - in fact slightly better than - the PMA and Experiment D methods. It is suggested that taking the square root actually increases the complexity of the problem in this case. However, the plots on the right of Figures 3.9, 3.10 and 3.11 show that with the bell function the opposite is true: F performs similarly to the PMA and D methods, but much worse. We also note that Experiment F requires a comparatively large value of γ to remain stable using the bell function: $\gamma = 20$, compared to $\gamma = 3$ for the standard PMA method. Figure 3.8 shows the result of using $\gamma = 19$, which causes the mesh to tangle after eight iterations when the points close to the origin move away from the origin very rapidly.

3.5.3 Experiment B

Experiment B uses Equation 34:

$$\gamma\nabla^2\phi^{n+1} = \gamma\nabla^2\phi^n - (|I + H(\phi^n)|)^{1/d} + \frac{c^n}{(M^n)^{1/d}}, \qquad \text{(34, copy)}$$

which is the same as the FP equation only we have taken a power of $1/d$ of the Hessian and monitor function terms, similarly to the PMA method. Experiment B is implemented using the 'optimal' value of γ: that which converges to a stable solution in the least time. These optimal values are $\gamma = 0.78$ for the ring function and $\gamma = 0.82$ for the bell function. These are much less variable than those required by the other methods, for example the FP method is fastest to converge when $\gamma = 0.85$ using the ring function, but with $\gamma = 2.5$ using the bell function; for Experiment F the values are $\gamma = 5$ and $\gamma = 20$ respectively.

Figures 3.9, 3.10 and 3.11 show that Experiment B performs similarly to the FP method using the ring function. Using the bell function the FP method does not converge but the lowest value of the equidistribution error, and initial residual, is reached after 185 iterations (the mesh after 250 iterations is shown in Figure 3.14); however, Experiment B converged in 86 iterations. For the bell monitor function, experiment B appears to be a significant improvement over the FP method and indeed the PMA method. The Experiment B initial residuals and equidistribution errors (Figures 3.9 and 3.10) show an exponential convergence of the method. This is similar to the PMA (thus Experiment D) method which is also a straight line on the logarithmic plot; the exponential convergence of the PMA method is predicted theoretically in [Budd and Williams, 2009,Browne et al., 2014]. We note that Experiment B in fact has a better rate of exponential convergence than the PMA method.

Experiment B is implemented using a steeper bell-shaped monitor function,

defined by Equation 46:

$$M(\mathbf{x}) = 1 + \alpha_1 \text{sech}^2\left(\alpha_2\left(|\mathbf{x} - \mathbf{x}_c|^2 - a^2\right)\right), \qquad \text{(46, copy)}$$

with $a = 0$, $\alpha_1 = 100$ and $\alpha_2 = 200$. Figure 3.12 shows the initial residuals and the number of CG-solver iterations using this steeper bell function, the PMA and AL methods are also plotted: the FP method and Experiment F did not converge using this monitor function.

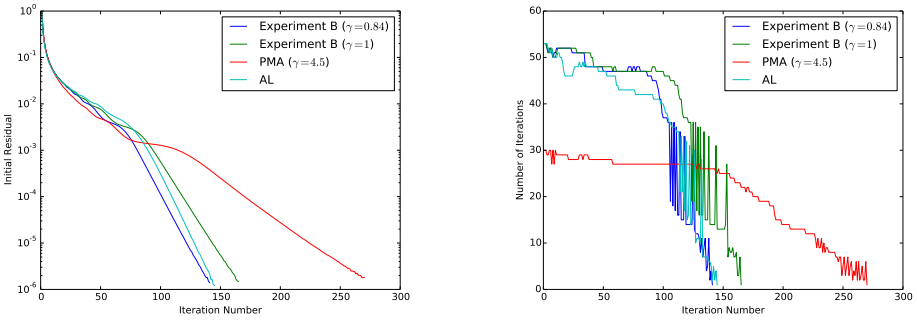

Figure 3.12: Results from implementing Experiment B, PMA and AL methods using the new, steeper bell-shaped monitor function. The equidistribution error (left) and the number of CG-solver iterations (right).

Figure 3.12 shows that using the steeper bell function, Experiment B performs very similarly to the AL method and is an improvement over the PMA method. Experiment B is certainly an improvement over Experiment F and the FP method which do not use the $1/d$ power law and do not converge. We note that the value $\gamma = 0.84$ used in Experiment B with the steeper bell function is very close to the $\gamma = 0.78$ and $\gamma = 0.82$ values used with the ring and bell functions respectively. A value of $\gamma = 1$ is sufficient to give fast[1], stable convergence using all three monitor functions. If this were true for a large class of monitor functions it would allow us to simplify Experiment B (Equation 34) by omitting the constant γ:

$$\nabla^2\phi^{n+1} \quad = \quad \nabla^2\phi^n - (|I + H(\phi^n)|)^{1/d} + \frac{c^n}{(M^n)^{1/d}}. \qquad (49)$$

This is unlikely to be a useful simplification because it will not work for all monitor functions. The power $1/d$ is likely to improve, but not guarantee, stability since some degree of relaxation will probably be necessary for certain monitor functions, as is the case with the PMA method. Setting $\gamma = 1$ in Experiment B is similar to the technique used to give stability of the FP method in [Weller et al., 2015] where γ is set to sufficiently high that the method is stable for a large class of monitor functions, but significantly slows down the method in cases where a large γ is not required. Figure 3.13 shows the initial residuals plotted for Experiment B with the 'optimal' value of γ, and Experiment B with $\gamma = 1$ (effectively the equation given above which omits γ), plotted together for the ring and the bell

[1] in comparison to the FP and PMA methods

functions. Figure 3.12 (left) shows the same for the steeper bell function. These figures show that if Experiment B is to converge in a similar number of iterations to the AL method then a $\gamma < 1$ must be found through trial and error.

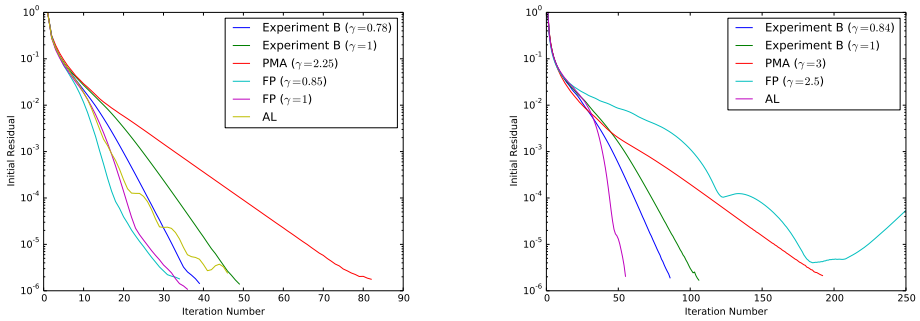

Figure 3.13: The initial residuals for the ring (left) and bell (right) monitor functions. These plots include Experiment B with $\gamma = 1$.

3.6 Comparison of the AL method to the FP and PMA methods

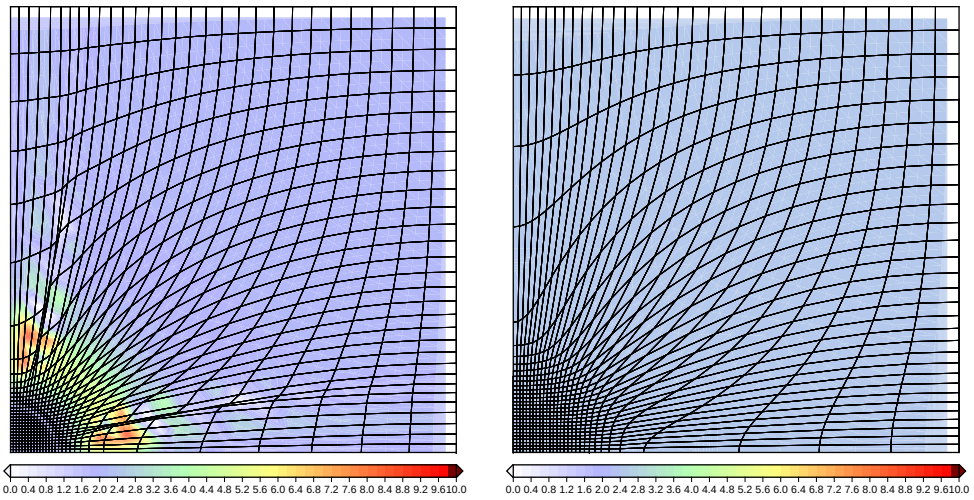

Figure 3.14: The first quarter of the bell mesh using the FP method with $\gamma = 1$ (left) and $\gamma = 2.5$ (right). The colour contours show the value of the equidistribution. With $\gamma = 1$ the mesh tangled after 18 iterations whereas with $\gamma = 2.5$ the method was stable but did not converge in under 1000 iterations, the mesh after 250 iterations is shown.

In figure 3.3 we have seen that using the ring function the FP method performs better that the PMA method, that is, it converges to an equidistributed solution more quickly, for $0.8 \leq \gamma \leq 2.5$, and better than the AL method for $0.8 \leq \gamma \leq 1.5$. Figure 3.4 shows that that the FP method with $\gamma = 1$ converges in fewer iterations than the PMA and AL methods when using the ring function but becomes unstable when using the bell function. Figure 3.14 shows the result of using the FP method with the bell monitor function: with $\gamma = 1$ (left) the mesh tangled

after 18 iterations, increasing the value of γ to 2.5 made the method more stable but did not converge. By adjusting the value of γ the PMA method is stable using the bell function but takes more than twice as many iterations to converge as it does with the ring function.

The most striking feature of the results shown in Figure 3.4 is that the AL method takes roughly the same number of iterations (≈ 50) to converge in both the ring and bell cases, and there is no arbitrary parameter γ which must be found by experimentation. Figure 3.12 shows the results of using the steeper bell function described in Section 3.5.3 [Experiment B]: in this case the AL method converges after ≈ 150 iterations, and in fact performs similarly to Experiment B.

The AL method is a significant improvement over the FP method, when using the bell function, and does not require a relaxation constant. The role of the relaxation constant is effectively to slow down the method. The mesh points are transported by the gradient of the mesh potential ϕ, as described in Lemma 1.2.2. When $\nabla\phi$ is large the mesh points are transported quickly, which can lead to tangling as illustrated in Figures 3.2 and 3.8. We thus require a large γ when the gradient of ϕ is very large. It would therefore be sensible to replace γ by a scalar function $g(\nabla\phi^n)$ which defines the relaxation locally based on the size of the gradient at the previous iteration.

We have noted in Chapter 2 [Exploration of Numerical Techniques] that the equations being solved in the FP and AL methods have very similar forms due to the similar ways in which they are derived. Both equations have the form:

$$L(\phi^{n+1}) = L(\phi^n) + \text{source term} \tag{50}$$

where L is a Laplacian operator. In FP the operator is $\gamma\nabla^2(\cdot)$ and in AL the operator is $\nabla \cdot (A^n \nabla(\cdot))$. We can expand these in 2D in order to compare:

$$\gamma\nabla^2 v = \gamma\, v_{xx} + \gamma\, v_{yy} \tag{51}$$

$$\nabla \cdot (A^n \nabla v) = (1 + \phi^n_{yy}) v_{xx} + (1 + \phi^n_{xx}) v_{yy} - 2\phi^n_{xy} v_{xy}. \tag{52}$$

In the AL method we have effectively replaced γ by something else. It is not as simple as a scalar function $g(\nabla\phi^n)$ but we note that the terms multiplying v_{xx} and v_{yy} are dependent on derivatives of ϕ^n, albeit second derivatives.

Chapter 4

Conclusions

Three algorithms are described which aim to equidistribute a mesh with respect to a given monitor function using numerical solutions of the Monge-Ampère equation: the FP, PMA and AL methods defined in Chapter 2 [Exploration of Numerical Techniques]. The FP and PMA methods include a relaxation parameter γ. Three experiments (Experiments B, D and F, defined in Section 2.4) are also described, which are designed to shed light on the differences between the FP and PMA methods.

All of the methods are implemented on a ring-shaped and a bell-shaped monitor function, defined in Section 3.1 [Choice of monitor function]. The convergence of each method is analysed by examining the initial residuals, equdistribution errors, number of CG-solver iterations, and CPU time taken.

From our experiments with the FP and PMA methods we conclude that both of these methods require adjustments to the arbitrary parameter γ in order to give stable solutions on some monitor functions. If the parameter γ is not included in the FP and PMA equation (i.e. $\gamma = 1$) then both methods are unstable when using the bell-shaped monitor function. The results from Experiment D suggest that the $\varepsilon(\phi^{n+1} - \phi^n)$ term in the PMA equation has no effect on the stability of the solutions and no significant effect on the speed of the algorithm.

The results from Experiment F suggest that the power $1/d$ in the PMA equation is required to ensure a stable solution using some monitor functions. In particular, a comparison of Experiment D and Experiment F shows that the power $1/d$ significantly improves the convergence of the PMA method using the bell-shaped monitor function.

From the results from Experiment B we conclude that the power $1/d$ also significantly improves the convergence of the FP method using the bell function. The power $1/d$ also makes the FP method converge exponentially, similarly to the PMA method, although the rate of convergence of Experiment B (the FP method with the power $1/d$) is greater than in the PMA method. Comparing the results

from Experiment B which uses the equation

$$\gamma\nabla^2\phi^{n+1} = \gamma\nabla^2\phi^n - (|I + H(\phi^n)|)^{1/d} + \frac{c^n}{(M^n)^{1/d}},$$

and Experiment D which uses the equation

$$\gamma\nabla^2\phi^{n+1} = \gamma\nabla^2\phi^n - (M^n|I + H(\phi^n)|)^{1/d} + c^n$$

and is similar to the PMA method, we conclude that it is the position of the monitor function M^n which makes the improved FP method (Experiment B) faster to converge, and more stable[1] than the PMA method.

We conclude that the AL method is stable and comparatively fast[2] to converge, and does not appear to require a relaxation constant, although we have only tested this using a small sample of monitor functions. The lack of an arbitrary parameter potentially makes the AL method much more useful than the FP and PMA methods in moving-mesh applications where the monitor function changes during calculations.

[1] In our experiments, Experiment B required a much lower degree of relaxation than the PMA method in order to remain stable.

[2] Compared to the FP and PMA methods. We note that the AL method is actually slightly slower than the FP method using the ring function, but is much better than the FP method using the bell function.

Chapter 5

Future work

5.1 Alternative convergence criteria

In this report we have analysed the PMA, FP and AL methods by comparing the number of iterations or CPU time taken for each method to converge to an equdistributed solution. The convergence criteria we have used is described in Section 1.4 [Solutions and convergence]. Figure 3.5 shows that when each method has converged (by our criteria) the equidistribution error is $\approx 10^{-5}$ in the case of the Ring function and $\approx 10^{-4}$ in the case of the Ring function. For practical applications it may be possible to considerably relax this criteria, [Budd et al., 2009] note that an accurate solution to the MA equation is not necessary for r-adaptivity. Relaxing our convergence criteria will result each method requiring fewer iterations: it may also result in different behaviour of the algorithms, and the degree of relaxation will depend on the application.

In this report the methods described have been implemented such that the equidistribution $M|I+H(\phi)|$ of each cell in the mesh is evaluated at the cell centre. The equidistributed state is reached when $M|I + H(\phi)| \approx c$ (approximately equal to because there is a small equidistribution error) for every cell in the mesh. In fact there is no convincing reason for evaluating at the cell centres. Alternatively, we could evaluate $M|I + H(\phi)| - c$ at the cell vertices. The Intermediate Value Theorem gives us the result that if the sign of $M|I + H(\phi)| - c$ is not the same for every cell vertex, then we must have that $M|I + H(\phi)| = c$ at some point in the cell. This alternative method for evaluating the equidistribution may speed up the methods described in this report. It remains a subject for future work.

5.2 Wider classes of monitor functions

In future work we would like to implement the methods described in this report on a wider range of different monitor functions. It would be particularly interesting to use non-symmetric functions and to investigate whether the conclusions drawn in Chapter 4 [Conclusions] regarding the performance of the different methods still apply.

5.3 Further application of the AL method

In this report we have used methods to redistribute a 2-dimensional mesh. The obvious next step is to adapt these methods to 3D problems. The PMA method is implemented in 3D in [Browne et al., 2014] and the derivation of the FP method used by [Weller et al., 2015] (see Section 1.3.2 [A fixed-point iterative method]) does not specifically use the dimension of the space. It is possible therefore that the FP method and hence the Experiment B method (which appears to be an improvement on the FP method) may be scaled up to 3D. In Section 2.5 we note that the derivation of the AL method relies specifically on the fact that the calculations are in 2D since the Laplacian term $\nabla \cdot (A\nabla(\psi))$ is determined by the linear terms in the expansion of $|I + H(\bar{\phi} + \psi)|$ in 2D. In 3D the linear terms in the expansion are given in Equation 41:

$$|I + H(\bar{\phi})| + \psi_{ii}[\bar{\phi}_{jj} + \bar{\phi}_{kk} + 3\bar{\phi}_{jj}\bar{\phi}_{kk} - \bar{\phi}_{jk}^2] - 2\psi_{jk}[\bar{\phi}_{jk} + \bar{\phi}_{ii}\bar{\phi}_{jk}] \qquad (41, \text{copy})$$

and, it is noted, do not simplify as a Laplacian term. Thus some changes would have to be made to the AL method in order to implement the method in 3D. This should be the subject of future work.

In this report the FP method is used to redistribute a mesh on the space $\left[-1/2, 1/2\right]^2 \subseteq \mathbb{R}^2$. [Weller et al., 2015] goes further by also implementing the FP method on the surface of a sphere. This requires some changes to the FP equations in order to take the curvature of the sphere into account. [Weller et al., 2015] compare their results on the sphere to a Voronoi meshing technique as described by [Ringler et al., 2008] which makes use of Lloyd's algorithm to construct a mesh on a sphere. Spherical calculations are particularly relevant to large-scale meteorological applications since the Earth is approximately spherical. No work has been done to experiment with using the AL method on the sphere but this should be the subject of future work.

5.4 Lloyd's algorithm and Centroidal Voronoi Tessellations

In [Weller et al., 2015] the FP method is used to generate a mesh on the sphere: the mesh is compared to that generated using Voronoi Meshing, using Lloyd's algorithm [Lloyd, 1982]. Voronoi Meshing - or the generation of a Centroidal Voronoi Tessellation (CVT) - is presented as an alternative to r-adaptive methods. CVTs are naturally orthogonal by construction and can be created so that the resolution of the resulting mesh is smoothly-varying [Ringler et al., 2008]. We here give a definition of a Voronoi Tessellation due to [Du et al., 1999]:

Definition 5.4.1. *Given a space Ω with norm $|\cdot|$, the Voronoi Tessellation $\{V_i\}_{i=1}^k$ of Ω is generated from the set of k points $\{z_i \in \Omega\}_{i=1}^k$ thus:*

$$V_i = \{x \in \Omega \mid \ |x - z_i| < |x - z_j| \text{ for } j = 1, ..., k, \ j \neq i\} \qquad (53)$$

When $\Omega = \mathbb{R}^2$ and $|\cdot|$ is the Euclidean norm, the Voronoi tessellation is a set of polyhedra and its dual is a Delaunay Triangulation.

As noted, it is often useful to consider a CVT $\{(z_i, V_i)\}_{i=1}^k$ where the V_i are the Voronoi regions generated from the z_i whilst, simultaneously, the z_i are the mass centres or means of the V_i [Lloyd, 1982]. That is, the z_i satisfy $z_i = \frac{1}{\int_{y\in V}\rho(y)dy}\int_{y\in V_i} y\rho(y)dy$ given a density ρ on Ω. This centroidal tessellation minimises the *energy* or *mean square error* given by

$$\mathcal{F}(\{(z_i, V_i)\}_{i=1}^k) = \sum_{i=1}^{k}\int_{y\in V_i}\rho(y)|y - z_i|^2 dy \tag{54}$$

[Du et al., 1999]. Note that to generalise to discrete Ω we may replace the integral over y with a sum.

[Ringler et al., 2008] discuss several examples in which CVTs may be utilised in climate models, including the resolution of eddies in global ocean simulations. To create a CVT which distributes the points z_i according to a monitor function, points in Ω are sampled according to a density ρ and then an algorithm is applied to produce a CVT based on those points. Here we discuss Lloyd's Method described in [Lloyd, 1982] and given in Algorithm 5.

Algorithm 5 Lloyd's algorithm

Construct the Voronoi tessellation defined by the generating points $\{z_i\}$.
Compute the centroids of the Voronoi regions. These centroids are then the new $\{z_i\}$ for the next iteration.
If the $\{z_i\}$ meet some convergence criterion, stop. Else, return to step 1.

Figure 5.1 shows (left) 256 generator points randomly distributed according to the density function $\rho((x, y)) = e^{-10(x^2+y^2)}$ on $[-1, 1]^2$, and the corresponding Veronoi tessellation; and (right) a centroidal Veronoi tessellation achieved through applying Lloyd's method to the random points.

[Ringler et al., 2008] give examples of areas in meteorology, and other fields, where the use of CVTs is a 'promising approach' to meeting the needs of high-resolution modelling requirements. However, because Lloyd's method as described in Algorithm 5 involves computing the centroids of each Voronoi region at each iteration, it is extremely expensive [Jacobsen et al., 2013, Weller et al., 2015]. [Du et al., 1999] discuss several variations on Lloyd's method which combine Lloyd's algorithm with dynamical programming methods, steepest-descent search algorithms, Newton iterations and probabilistic methods. [Jacobsen et al., 2013] describe a parallel algorithm for the construction of a Delauney triangulation (dual to a Voronoi tessellation) which is based on Lloyd's algorithm but does not require the integration over Voronoi regions and is thus markedly faster at producing CVTs.

It is proposed that the AL method described in this report could be combined with Lloyd's method and used to speed up generation of CVTs for use in adaptive mesh applications. [Dietachmayer and Droegemeier, 1992] note that

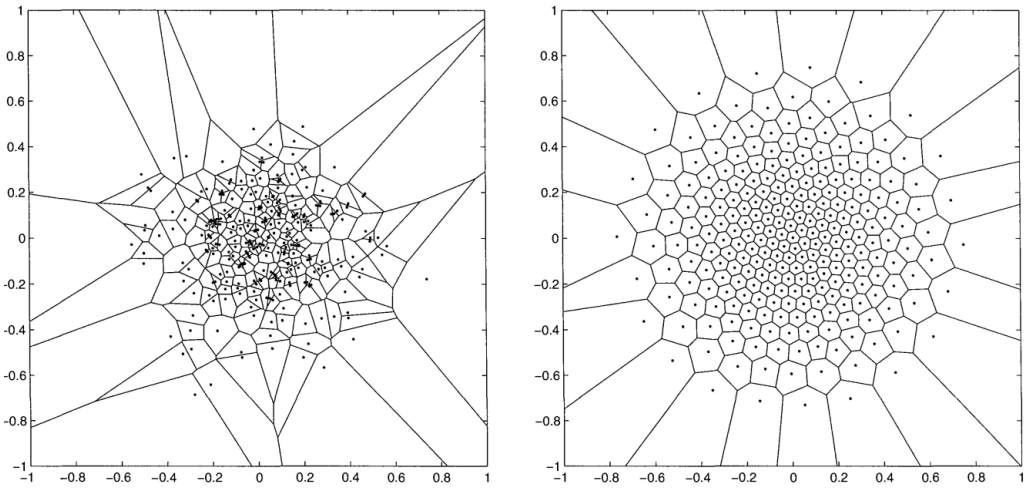

Figure 5.1: Reproduced from [Du et al., 1999]. 256 points randomly distributed according to the density function $\rho((x, y)) = e^{-10(x^2+y^2)}$ on $[-1, 1]^2$ (left) and the resulting CVT after applying Lloyd's algorithm (right).

meshes generated using r-adaptivity can be very distorted, this problem increases on larger domains where mesh points may be moved very large distances. [Weller et al., 2015] note that it may be desirable, using r-adaptivity, to start with a mesh in which the points are already in roughly the right places. It is proposed that a CVT, generated using Lloyd's algorithm, may provide the initial mesh for r-adaptive methods. This is a topic for future work.

Bibliography

[Benamou et al., 2010] Benamou, J.-D., Froese, B. D., and Oberman, A. M. (2010). Two numerical methods for the elliptic monge-ampere equation. *ESAIM: Mathematical Modelling and Numerical Analysis*, 44(04):737–758.

[Brenier, 1991] Brenier, Y. (1991). Polar factorization and monotone rearrangement of vector-valued functions. *Communications on pure and applied mathematics*, 44(4):375–417.

[Browne et al., 2014] Browne, P., Budd, C., Piccolo, C., and Cullen, M. (2014). Fast three dimensional r-adaptive mesh redistribution. *Journal of Computational Physics*, 275:174–196.

[Budd et al., 2013] Budd, C. J., Cullen, M., and Walsh, E. (2013). Monge–ampére based moving mesh methods for numerical weather prediction, with applications to the eady problem. *Journal of Computational Physics*, 236:247–270.

[Budd et al., 2009] Budd, C. J., Huang, W., and Russell, R. D. (2009). Adaptivity with moving grids. *Acta Numerica*, 18:111–241.

[Budd and Williams, 2009] Budd, C. J. and Williams, J. (2009). Moving mesh generation using the parabolic monge-ampere equation. *SIAM Journal on Scientific Computing*, 31(5):3438–3465.

[Cao et al., 2003] Cao, W., Huang, W., and Russell, R. D. (2003). Approaches for generating moving adaptive meshes: location versus velocity. *Applied Numerical Mathematics*, 47(2):121–138.

[Chacón et al., 2011] Chacón, L., Delzanno, G. L., and Finn, J. M. (2011). Robust, multidimensional mesh-motion based on monge–kantorovich equidistribution. *Monthly Weather Review*, 230(1):87–103.

[Dietachmayer and Droegemeier, 1992] Dietachmayer, G. and Droegemeier, K. (1992). Application of continuous dynamic grid adaption techniques to meteorological modeling. part i: Basic formulation and accuracy. *Journal of Computational Physics*, 120(8):1675–1706.

[Du et al., 1999] Du, Q., Faber, V., and Gunzburger, M. (1999). Centroidal voronoi tessellations: applications and algorithms. *SIAM review*, 41(4):637–676.

[Froese, 2012] Froese, B. D. (2012). *Numerical methods for the elliptic Monge-Ampere equation and optimal transport*. PhD thesis, Science: Department of Mathematics.

[Froese and Oberman, 2011] Froese, B. D. and Oberman, A. M. (2011). Convergent finite difference solvers for viscosity solutions of the elliptic monge-ampere

equation in dimensions two and higher. *SIAM Journal on Numerical Analysis*, 49(4):1692–1714.

[Fryxell et al., 2000] Fryxell, B., Olson, K., Ricker, P., Timmes, F., Zingale, M., Lamb, D., MacNeice, P., Rosner, R., Truran, J., and Tufo, H. (2000). Flash: An adaptive mesh hydrodynamics code for modeling astrophysical thermonuclear flashes. *The Astrophysical Journal Supplement Series*, 131(1):273.

[Griffies et al., 2000] Griffies, S. M., Böning, C., Bryan, F. O., Chassignet, E. P., Gerdes, R., Hasumi, H., Hirst, A., Treguier, A.-M., and Webb, D. (2000). Developments in ocean climate modelling. *Ocean Modelling*, 2(3):123–192.

[Jacobsen et al., 2013] Jacobsen, D., Gunzburger, M., Ringler, T., Burkardt, J., and Peterson, J. (2013). Parallel algorithms for planar and spherical delaunay construction with an application to centroidal voronoi tessellations. *Geoscientific Model Development*, 6(4):1353–1365.

[Lee et al., 2013] Lee, T., Baines, M. J., Langdon, S., and Tindall, M. J. (2013). A moving mesh approach for modelling avascular tumour growth. *Applied Numerical Mathematics*, 72:99–114.

[Lloyd, 1982] Lloyd, S. (1982). Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137.

[Power et al., 2006] Power, P., Piggott, M., Fang, F., Gorman, G., Pain, C., Marshall, D., Goddard, A., and Navon, I. (2006). Adjoint goal-based error norms for adaptive mesh ocean modelling. *Ocean Modelling*, 15(1):3–38.

[Ringler et al., 2008] Ringler, T., Ju, L., and Gunzburger, M. (2008). A multiresolution method for climate system modeling: application of spherical centroidal voronoi tessellations. *Ocean Dynamics*, 58(5-6):475–498.

[Saint-Cyr et al., 2007] Saint-Cyr, A., Jablonowski, C., Tufo, H., Thomas, S., and Dennis, J. (2007). A comparison of two shallow water models with nonconforming adaptive grids: classical tests. Technical report.

[Tang, 2005] Tang, T. (2005). Moving mesh methods for computational fluid dynamics. *Contemporary mathematics*, 383:141–174.

[Weller, 2009] Weller, H. (2009). Predicting mesh density for adaptive modelling of the global atmosphere. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 367(1907):4523–4542.

[Weller et al., 2015] Weller, H., Browne, P., Budd, C., and Cullen, M. (2015). Numerical solution of the optimal transport problem for mesh generation on the sphere. *Manuscript in preparation*.