THESIS FOR THE DEGREE OF LICENTIATE OF ENGINEERING

On Some Problems in Systems Biology and Geometric Flows

Tobias Gebäck

CHALMERS | GÖTEBORG UNIVERSITY

Department of Mathematical Sciences
Division of Mathematics
Chalmers University of Technology and Göteborg University
Göteborg, Sweden, 2005

On Some Problems in Systems Biology and Geometric Flows
Tobias Gebäck

©Tobias Gebäck, 2005

Licentiate Thesis
ISSN 1652-9715/No. 2005:8

Department of Mathematical Sciences
Division of Mathematics
Chalmers University of Technology and Göteborg University
SE-412 96 Göteborg
Sweden
Telephone +46 (0)31 772 1000

Printed in Göteborg, Sweden 2005

On Some Problems in Systems Biology and Geometric Flows

Tobias Gebäck

Abstract

This thesis consists of three distinct parts.

The first part concerns spatial modeling of signaling pathways in eukaryotic cells, which are systems that enable cells to respond to outside stimuli and changes in the environment. The signaling is performed through a series of enzymatic reactions that starts at the cellular membrane and ends in the nucleus. We focus on modeling the transport of enzymes through diffusion and show that the inclusion of diffusion in the model may have a large impact on the output, compared to spatially homogeneous models. The system is modeled using partial differential equations in three space dimensions, which are solved using finite differences and the Immersed Interface Method.

The second part of the thesis discusses this Immersed Interface Method and its application to three-dimensional, time-dependent problems. Furthermore, the method is extended to work on Boolean grids, which are grids with a special structure that makes it possible to reduce the number of grid nodes while retaining the same accuracy of approximation.

The third and last part contains the proof of convergence for an algorithm to compute generalized mean curvature flows with right-angle boundary conditions. Mean curvature flows describe the evolution of a surface which at each point is assigned a normal velocity depending on the mean curvature of the surface at that point. We consider flows where the velocity is equal to an increasing, continuous function of the mean curvature. Furthermore, we assume that the surface is located inside a convex domain and that whenever it intersects the domain boundary, it should do so at a right angle. The algorithm is based on a convolution-thresholding scheme and we show the convergence of the output, as the time step tends to zero, to the viscosity solution of the corresponding mean curvature PDE.

Keywords: systems biology; MAPK signaling pathways; reaction-diffusion equation; Immersed Interface Method; Boolean grids; curvature flows; viscosity solutions; convolution-thresholding schemes

Acknowledgements

First of all, I would like to thank my supervisor Alexei Heintz for coming up with many of the ideas behind this work and for his encouragement and good advice, as well as for pleasant times with guitar and violin in hand.

I also wish to thank my co-supervisor Per Sunnerhagen at Dept. of Cell and Molecular Biology, Göteborg University, for sharing his knowledge on signaling pathways and for reading and commenting on Part I of this thesis. Furthermore, my "pair" Ph.D. student in biology Claes Molin deserves thanks for interesting discussions regarding cell biology and other topics.

I would also like to thank Ricards Grzibovskis who has taken time to adapt his code for computing mean curvature flows, which was used to generate the nice illustrations at the end of Part III.

I am grateful to Peter Kumlin for reading and giving valuable comments on Part III of this thesis, and to Mohammad Asadzadeh for doing the same work with Part II.

A thought of gratitude also goes to the National Research School in Genomics and Bioinformatics, which provides the financial support for my Ph.D. studies.

Finally, I would like to thank my family for their care and support and my friends for giving me other things to do and think about.

Tobias Gebäck
Göteborg, March 2005

Preface

This thesis consists of three independent parts. The first part concerns spatial modeling of intracellular signaling pathways, which is the project supported by the National Research School in Genomics and Bioinformatics. The text in part I is supposed to be less demanding than in the other two parts, making it accessible to biologists and other people with limited knowledge of mathematics. That said, there still are some equations appearing now and then, but those parts could be skipped if the reader has trouble understanding the mathematical symbols. Also, having some background knowledge in biology is preferable, although some effort has been made to give the necessary background and to explain the terms used.

Part II describes the computational method used to perform the calculations in Part I, as well as an extension of that method to Boolean grids, which makes it possible to reduce the number of grid nodes when performing the calculations. Part II requires more mathematical knowledge and a reader who is inexperienced in mathematics will find it hard to follow.

Part III is not related to the two first parts, but it is a continuation of my master thesis concerning algorithms for computing mean curvature flows, that is, the motion of surfaces which are assigned a velocity at each point depending on the mean curvature of the surface. Reading (and understanding) this part requires a proper mathematical background.

Spatial modeling of MAPK signaling pathways

Tobias Gebäck

Abstract

MAPK signaling pathways in eukaryotic cells are sequences of enzymatic reactions that convey a signal from the cellular membrane to the nucleus, in response to some stimulus. They constitute an often vital signaling system which enables the cell to react to changes in the environment and to survive such changes.

We investigate the effects of spatial models for signaling pathways. First, we add a diffusion term to the Kholodenko model for general MAPK signaling pathways. This has a large effect on the amplitude of the oscillations that the model predicts, indicating that such oscillations have a smaller effect in a model that takes into account the spatial distribution of proteins, compared to the original space-independent model.

Second, we investigate a simple spatial model for the HOG pathway in the yeast *Saccharomyces cerevisiae*. We are able to reproduce the nuclear relocation of the Hog1 protein and also see that diffusion in the model is so fast that differences in protein concentration throughout the cell are small, even though reactions are localized only at the membrane.

The calculations are performed in three space dimensions using finite differences and the Immersed Interface Method, which is described in part II of this thesis.

Contents

1 Introduction **1**
- 1.1 Signaling pathways . . . 2
- 1.2 Outline . . . 3

2 The Kholodenko model with diffusion **4**
- 2.1 Introduction . . . 4
- 2.2 The Kholodenko model . . . 4
- 2.3 Adding diffusion . . . 6
- 2.4 Results . . . 7
- 2.5 Discussion . . . 16

3 Spatial modeling of the HOG pathway in yeast **17**
- 3.1 The HOG pathway . . . 17
- 3.2 The model . . . 17
- 3.3 Results . . . 22
- 3.4 Discussion . . . 25

References **27**

Glossary

amino acid the 20 different building blocks that make up proteins

differential equation an equation involving derivatives, whose solution is a function of one or more variables

enzyme protein that speeds up specific reactions in the cell

eukaryotic cell a cell containing a nucleus, as opposed to bacteria

gene a DNA sequence coding for a protein

genome the collection of all the genes of an organism, coded for by DNA

HOG High Osmolarity Glycerol

in vitro experiment performed in an artificial environment, outside the organism

in vivo experiment performed inside a living organism

kinase enzyme whose function is to phosphorylate other enzymes

MAPK Mitogen Activated Protein Kinase

ODE Ordinary Differential Equaqtion; a differential equation in one variable, often time

osmosis the process that strives to even out the concentration of solutes across a membrane

PDE Partial Differential Equation; a differential equation in several variables, such as space and time

phosphorylation the addition of a phosphate group (PO_3) to a protein

1. Introduction

The ordinary bakers' yeast *Saccharomyces cerevisiae*, which is a unicellular fungus, is a very widely studied organism among cell biologists. The reasons for this are many. One is that for a long time there has been a commercial interest for brewers and bakers to understand the organism in order to maximize its output of alcohol and carbon dioxide. Another reason is that it is a relatively simple unicellular eukaryotic organism that is easy to handle in the lab and can be used as a model organism for higher eukaryotic organisms, such as plant and mammal cells. Nowadays, another reason for studying yeast is that it is already very well studied, which means that more extensive studies can be performed, trying to understand more complex processes in the cell. For example, the complete yeast genome has been sequenced and it contains approximately 6.300 genes (cf. human genome approx. 30.000 genes) [1]. Furthermore a complete library of gene deletions has been set up, i.e. for (almost) every single gene, there is a yeast strain available that has this particular gene deleted from its genome, enabling biologists to easily study the effects of removing a gene from a cell under different conditions, thus hopefully learning more about the function of that gene. Also, there are many research groups continually working on different aspects of the yeast cell, making the amount of data available comparatively large.

Having said that, however, it should be noted that even the simple yeast cell is not at all understood by the biologists. There are many genes coding for proteins with unknown function and even if the gene codes for a protein that has a known function, this function may depend on other proteins and substances, so that the overall behavior is not very well understood anyway. The cell as a whole is a very complicated system, where proteins, DNA, RNA, lipids and other molecules work together to define the behavior of the cell. And although there has been a tremendous increase in knowledge about the cell during the last decades, only small parts of the complete system are well understood. For example, a single protein coded for by a single gene may be studied to determine its amino acid sequence, its three-dimensional structure, its active sites, where it may bind to other proteins, etc. This gives very valuable information about the protein, but does not tell the whole story, since questions like "When is it expressed from the DNA?", "What activates/deactivates the protein?", "Where is it located?" and so on, must also be answered to give a complete picture. The answers to this kind of questions do not depend solely on the protein itself, but also on other proteins and molecules in the cell, as well as outside stimuli and the overall "state" of the cell.

The complexity of these questions is the basis for *systems biology*, which is the research area that tries to look at larger systems of proteins and cellular functions, often using mathematical modeling in order to understand the behavior of that particular system. There is no single definition of systems biology and it is not very fruitful to try to come up with one, since these "systems" may be very different in character and the methods applied to study them may also vary accordingly. The philosophy of systems biology is not without controversies, since there is no long tradition of using mathematical modeling in cell biology, but there are a few examples where modeling has been successful as a complement to the experimental data in order to understand the behavior of a cellular

system (see [8], [12]).

1.1. Signaling pathways

One type of cellular system that is suited for mathematical modeling is signaling pathways. We will be concerned here with MAPK[1] pathways. Specifically, we have the High Osmolarity Glycerol (HOG) pathway in yeast in mind, but most of what is said here applies to other pathways as well. See [1, chapter 15] for general information about signaling pathways and [4] for a review of the HOG pathway. The signaling in a MAPK pathway starts at the cellular membrane, where it is activated by some stimulus, such as the presence of a specific substance (e.g. pheromones) in the environment or a more general environmental change, such as change in osmotic pressure or oxidative properties of the environment.

This stimulus is sensed in one way or another by receptors or other mechanisms at the membrane. These sensing mechanisms then convey a signal to another protein by phosphorylation (i.e. adding a phosphate group to one of the amino acids of the target). This starts a chain of phosphorylation events, which convey the signal through two or three steps, where each step consists of the phosphorylation of a target kinase, that is an enzyme which, once activated by phosphorylation, may phosphorylate other target proteins. So, as is seen in figure 2.1, the signaling cascade moves from the MAP-kinase-kinase-kinase (MKKK) which activates the MAP-kinase-kinase (MKK), which activates the MAP-kinase (MAPK), which in turn moves to the nucleus where it may activate or deactivate transcription factors that control gene expression. The MAPK may also have other functions by controlling the activity of enzymes throughout the cytoplasm and nucleus. Activation of kinases in the chain may require a double phosphorylation of two amino acids in the protein, which are both performed by the higher-level kinase.

The effect of the signaling pathway is that the cell is able to sense changes in the environment and convey the information of this change to the nucleus or other inner parts. There the cell can produce the appropriate response to the stimulus, which is often vital for the survival of the cell. With the multiple steps in the chain, the cell is able to amplify the signal and may also increase the steepness of the response, creating a switch-like response so that the signal is more or less either "on" or "off" [5]. In addition to the activating kinases, there are also deactivating phosphatases, which remove the phosphate from the enzymes, thereby deactivating them (a dephosphorylation need not be deactivating, but in this case it is). Thus, when the stimulus disappears, or the cell has adapted to the new environment, the signaling pathway switches back off through the action of the phosphatases.

It should also be mentioned that (as always in biology) things are more complicated than they seem. For example, the osmosensing mechanism is often not very well known and may include many proteins; the phosphorylation events may take place when the kinases are organized in scaffolds or large protein complexes; and the cell is full of other enzymes and molecules which may influence the signaling pathway, producing different results depending on which state the cell is in. The models we discuss here focus on

[1] Mitogen Activated Protein Kinase

the phosphorylation cascade and the movement of the phosphorylated MAPK from the membrane to the nucleus. They are of course great simplifications, but may hopefully provide some insight into the reality.

1.2. Outline

In the following sections, we study two spatial models of MAPK signaling pathways. In section 2, we investigate the effects of diffusion on the oscillations predicted by a model of a MAPK cascade, including a negative feedback loop. In section 3, we study a model of the nuclear relocalization of the yeast MAPK Hog1. The two sections are almost independent and contain separate results and discussion parts.

2. The Kholodenko model with diffusion

2.1. Introduction

In the year 2000, Boris N. Kholodenko [6] published a model of general mitogen-activated protein kinase (MAPK) signaling cascades, which was shown to give rise to oscillatory behavior for a range of parameter values. An essential feature of the model is a negative feedback loop, meaning that the end product of the pathway inhibits the activating reaction (see figure 2.1). The model received some attention, since the appearance of oscillations was a rather unexpected effect which could have interesting implications for the biology of the cells. However, no oscillations have actually been observed for signaling pathways in real cells. There could be numerous reasons for this, for example that the abundance of proteins is measured as the total protein content in a large number of cells, which means that unsynchronized oscillations will not show up in measurements. But it is perhaps more probable that the model does not agree well enough with reality, meaning that the oscillations in MAPK-pathways are artifacts which do not occur in nature. Specifically, the feedback loop in the model may be a too simple model of the feedback that is known to be present (since the signaling is turned off after a while). It is also a fact that the signaling does not take place at one point in space but involves movement of proteins through the cell. Some of the reactions take place only at the cell membrane, while some may take place wherever the participating proteins encounter each other. This should have an effect on the oscillatory behavior, since as proteins at the end of the reaction chain move away from the membrane, they cannot take part in the feedback, which should then be attenuated. The purpose of the following sections is to incorporate diffusion of proteins into the model and study the effect that this has on the oscillations and the behavior of the pathway. Although the Kholodenko model may not be an accurate model, it is quite convenient to study an oscillating system, since the oscillations are easily seen and effects on them are easily detected.

2.2. The Kholodenko model

The structure of the Kholodenko model for MAPK signaling pathways is shown in figure 2.1. The pathway is activated by a stimulus of some kind, which causes the MKKK to be phosphorylated. This activates the MKKK, which then in turn is able to phosphorylate the MKK at two different sites. The active double-phosphorylated MKK (MKKPP) may then activate the MAPK by phosphorylation, again at two sites. In a real cell, the MAPKPP continues to perform some action, which sooner or later will turn the signaling cascade off, by some means which are to a large extent unknown (and may differ between different MAPK pathways). In the Kholodenko model, this is modeled by that the MAPKPP inhibits the phosphorylation of the MKKK. This is the negative feedback loop, which is needed in order to create oscillatory behavior.

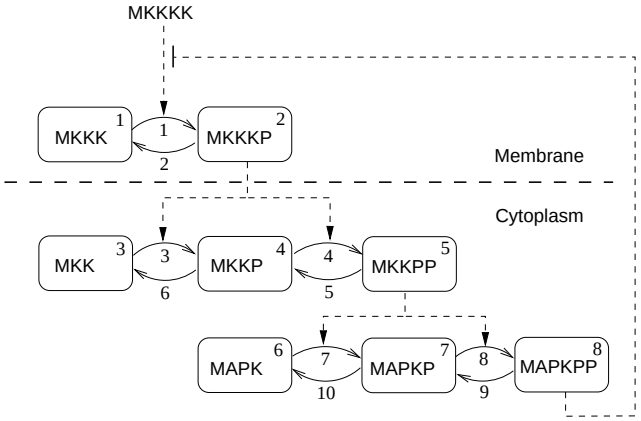

Figure 2.1: The Kholodenko model. The signaling pathway is activated by some stimulus which leads to phosphorylation of the MKKK, which in turn phosphorylates the MKK in two steps. The double-phosphorylated form of MKK then phosphorylates the MAPK, also in two steps. The end product (MAPKPP) then has the effect of inhibiting the stimulus, thus forming a negative feedback loop. Here, we also add the additional assumption that the MKKK is fixed at the cell membrane, while the other proteins are free to diffuse through the cytoplasm.

The corresponding ordinary differential equations may be written as

$$\begin{aligned}
du_1/dt &= v_2 - v_1 \\
du_2/dt &= v_1 - v_2 \\
du_3/dt &= v_6 - v_3 \\
du_4/dt &= v_3 + v_5 - v_4 - v_6 \\
du_5/dt &= v_4 - v_5 \\
du_6/dt &= v_{10} - v_7 \\
du_7/dt &= v_7 + v_9 - v_8 - v_{10} \\
du_8/dt &= v_8 - v_9
\end{aligned}$$

where the concentrations of MKKK through MAPKPP are denoted $u_j,\ j = 1, \ldots, 8$ with numbers as in the boxes in figure 2.1, and the fluxes of the reactions are denoted $v_i,\ i = 1, \ldots, 10$ and also numbered as in figure 2.1. The expressions for the fluxes are given in table 2.1.

Flux	Rate equation	Parameter values
v_1	$V_1 u_1/((1 + (u_8/K_I)^n)(K_1 + u_1))$	$V_1 = 2.5;\ n = 1;\ K_I = 9;\ K_1 = 10;$
v_2	$V_2 u_2/(K_2 + u_2)$	$V_2 = 0.25;\ K_2 = 8;$
v_3	$k_3 u_2 u_3/(K_3 + u_3)$	$k_3 = 0.025;\ K_3 = 15;$
v_4	$k_4 u_2 u_4/(K_4 + u_4)$	$k_4 = 0.025;\ K_4 = 15;$
v_5	$V_5 u_5/(K_5 + u_5)$	$V_5 = 0.75;\ K_5 = 15;$
v_6	$V_6 u_4/(K_6 + u_4)$	$V_6 = 0.75;\ K_6 = 15;$
v_7	$k_7 u_5 u_6/(K_7 + u_7)$	$k_7 = 0.025;\ K_7 = 15;$
v_8	$k_8 u_5 u_7/(K_8 + u_8)$	$k_8 = 0.025;\ K_8 = 15;$
v_9	$V_9 u_8/(K_9 + u_8)$	$V_9 = 0.5;\ K_9 = 15;$
v_{10}	$V_{10} u_7/(K_{10} + u_7)$	$V_{10} = 0.5;\ K_{10} = 15;$

Table 2.1: Fluxes and parameter values in the Kholodenko model. The values are the ones given in the original article [6].

The output of the model with these parameter values is shown in figure 2.2. We see the concentrations of MAPK and MAPKPP and note that they oscillate heavily, and that the oscillations are sustained. The oscillations are present for a range of parameter values, although the frequency and amplitude may vary.

2.3. Adding diffusion

We now wish to add spatial movements of proteins to the Kholodenko model. We assume that the MKKK is fixed at the membrane and that MKK and MAPK may diffuse freely through the cell cytoplasm. This means that the reactions 1, 2, 3 and 4 take place only near the membrane, where MKKK is present, while the other reactions take place all over the cell.

The geometry that we use is a near spherical cell of diameter 9 µm in three dimensions with no inner structure. This is of course a great simplification, but still it is a more advanced model than the original one, which does not include the space dimension at all. The size of the cell is the approximate size of a yeast cell, which is a rather small cell compared to other eukaryotic cells. A diffusion term is added to all the equations for the MKKs and MAPKs, yielding the eight equations

$$\begin{aligned}
du_1/dt &= v_2 - v_1 \\
du_2/dt &= v_1 - v_2 \\
du_3/dt &= d_3\Delta u_3 + v_6 - v_3 \\
du_4/dt &= d_4\Delta u_4 + v_3 + v_5 - v_4 - v_6 \\
du_5/dt &= d_5\Delta u_5 + v_4 - v_5 \\
du_6/dt &= d_6\Delta u_6 + v_{10} - v_7 \\
du_7/dt &= d_7\Delta u_7 + v_7 + v_9 - v_8 - v_{10} \\
du_8/dt &= d_8\Delta u_8 + v_8 - v_9
\end{aligned}$$

with $u_j = u_j(x, y, z, t),\ j = 1, \ldots, 8$, and $\Delta = \partial^2/\partial x^2 + \partial^2/\partial y^2 + \partial^2/\partial z^2$ denoting the

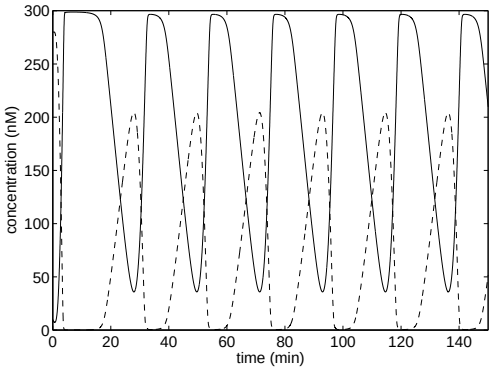

Figure 2.2: Typical behavior of the Kholodenko model. The graph shows the concentration of MAPK (solid) and MAPKPP (dashed) for times up to 150 minutes. We see the sustained oscillations in concentration.

Laplacian. Here d_j are the diffusion coefficients, which are now additional parameters in the model, describing how fast the diffusion of proteins is. These are known (see e.g. [10]) to be much lower inside the cell than in water (up to 10 times), because the cell is full of obstacles such as the cytoskeleton and other proteins. For simplicity, we assume that all the diffusion coefficients are equal, i.e. $d_3 = \ldots = d_8 = d$. For globular (near-spherical) proteins in water, the diffusion coefficient may be estimated from the relation

$$d \approx c \cdot W^{-1/3}, \tag{2.1}$$

where W is the molecular weight (see [9]). Fitting of measurement values for medium-sized proteins tabulated in [2, chapter 7] gives $c \approx 2.7 \cdot 10^3$ if W is measured in Daltons and d in $\mu m^2/s$. For the MAPK Hog1 in yeast, which has a molecular weight of 48.8 kDa, this gives $d = 74\ \mu m^2/s$ for free diffusion in water, which we use as a reference value. The equations for the fluxes are the same as before, i.e. the ones given in table 2.1.

The equations were solved using the Immersed Interface Method and finite differences on uniform grids with $54 \times 54 \times 54$ nodes. The method is described in Part II of this thesis.

2.4. Results

We study the results of the model for two different values of the diffusion coefficient, d, namely $d_H = d_0/10$ and $d_L = d_0/1000$, with $d_0 = 74\ \mu m^2/s$ being the approximate diffusion coefficient in water for Hog1.

The higher value d_H for the diffusion coefficient is so large that the molecules have time to move around the entire cell faster than the reactions produce any significant changes in concentration. This means that the proteins will be evenly spread across the cell at all times. This is seen in figure 2.8 for two of the components in the pathway (MKK and MAPKPP). The figure shows concentrations for times between 0 and 50 min on a line through the center of the cell. One sees that the concentration is the same in the center of the cell as at the edge. One can also see that the oscillations are not as large as for the original model. This is shown more clearly in figures 2.4 and 2.5, which should be compared to figure 2.3 for the original model. Here concentrations for all components in the model are shown. Samples are taken at the membrane (figure 2.4) and at the center of the cell (figure 2.5). It is clear that the oscillations have a much smaller amplitude now that we have added diffusion. Furthermore, the oscillations seem to be damped, so that the amplitude decreases with time. This has been confirmed by running longer simulations, where the oscillations slowly fade away. The diffusion acts as a damper for the system.

A similar behavior is seen with the lower value $d = d_L$ for the diffusion coefficient (figures 2.6 and 2.7). The oscillations again have a lower amplitude and are again damped. Here, however, the proteins do not have time to diffuse through the cell before the reactions produce significant changes in phosphorylation levels. This is seen clearly in figure 2.9, where again concentrations of MKK and MAPKPP are shown along a line through the cell center for different times. It is clear that the concentrations are different in the center and at the membrane. However, the oscillations are present and approximately equal in period and amplitude in the center and at the membrane, and also approximately equal to the previous case (with $d = d_H$).

A final example is shown in figure 2.10. There, we show the result of the same model with slow diffusion ($d = d_L$), but in a larger cell with a diameter of about 90 $\mu m^2/s$. Now the phosphorylated proteins do not have time to move very far from the membrane before they are dephosphorylated. Therefore the oscillations occur only near the membrane, while in the center all the kinases are in their inactive (unphoshorylated) state. This of course makes the signaling pathway useless, since its main purpose is to convey the signal of phosphorylations to the cell nucleus, which it fails to do when distances are large, diffusion slow and dephosphorylation reactions comparatively fast.

It is clear that what determines the spatial behavior of the system is a combination of the length scale, the diffusion coefficient and the reaction rate. In order to see this more clearly, we may make the equations non-dimensional. To illustrate this, we take the equation for u_5 (MKKPP), but include only the dephosphorylation reaction with flow v_5. The equation then becomes

$$\frac{\partial u_5}{\partial t} = d\Delta u - \frac{V_5 u_5}{K_5 + u_5}.$$

If we introduce the non-dimensional space variables $(\xi, \eta, \zeta) = h^{-1}(x, y, z)$, the non-dimensional time $\tau = d/h^2 \cdot t$ and the non-dimensional concentration $\nu = u_5/u_0$, with

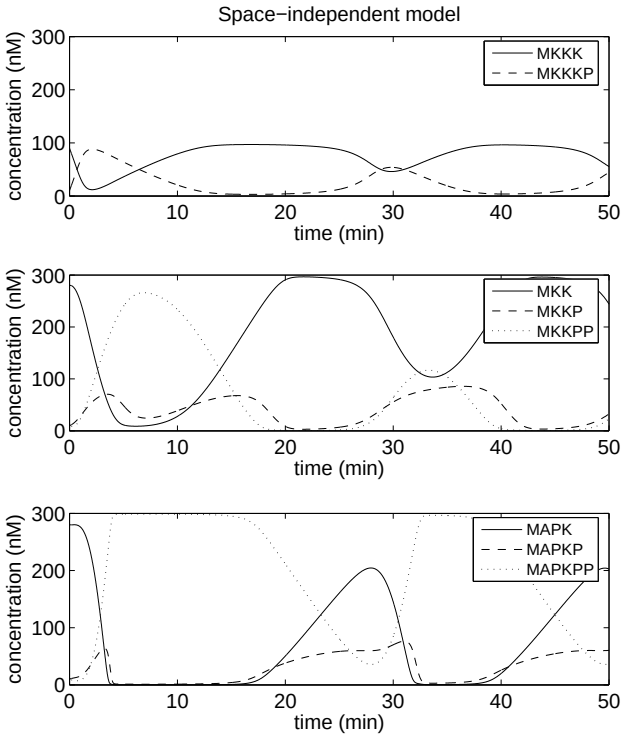

Figure 2.3: The concentrations of the eight components of the original (space-independent) model.

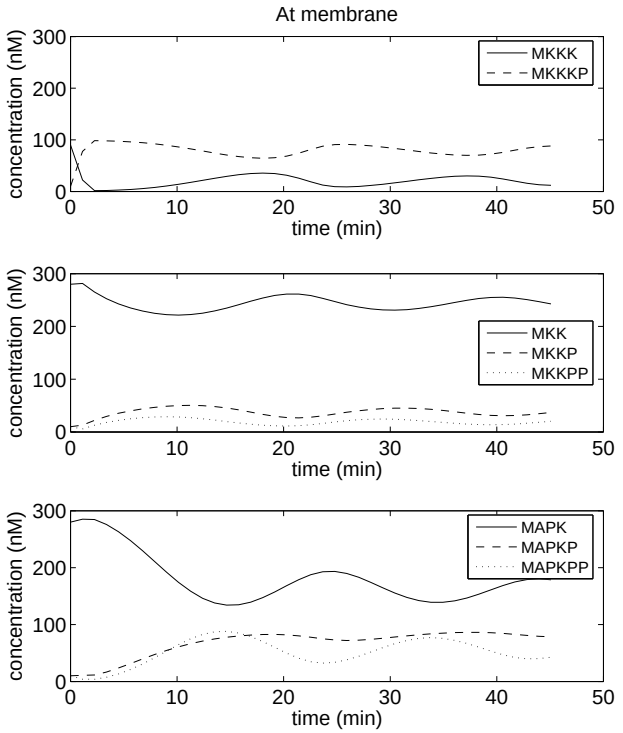

Figure 2.4: Concentrations for the eight components in the space-dependent model with fast diffusion $d = d_H$. The values are taken at the membrane.

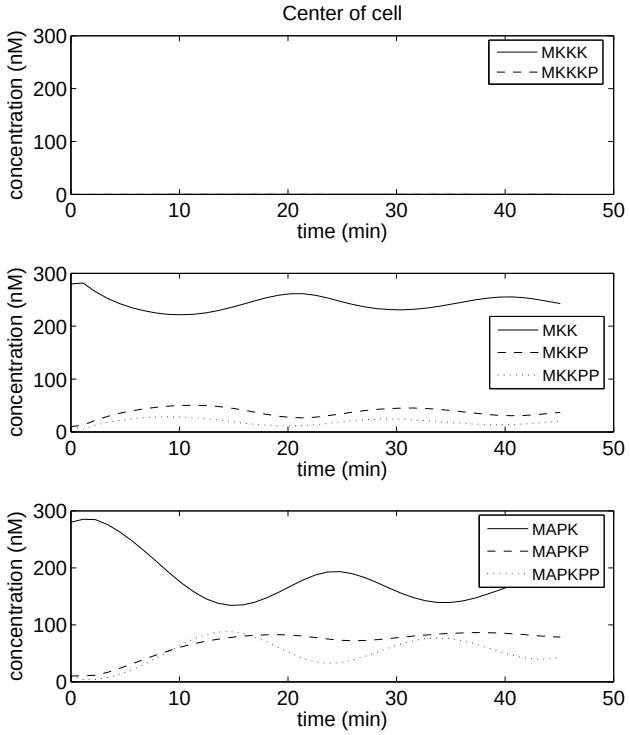

Figure 2.5: Concentrations for the eight components in the space-dependent model with fast diffusion $d = d_H$. The values are taken at the center of the cell.

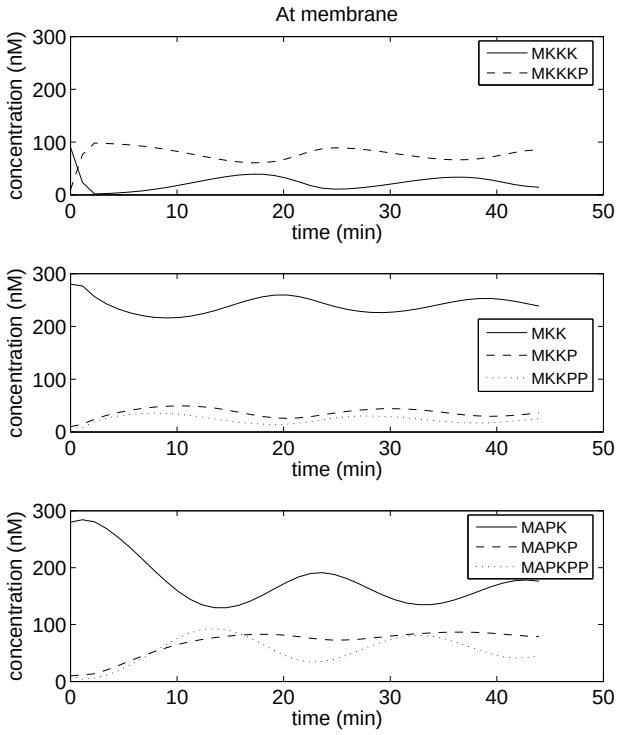

Figure 2.6: Concentrations for the eight components in the space-dependent model with slow diffusion $d = d_L$. The values are taken at the membrane.

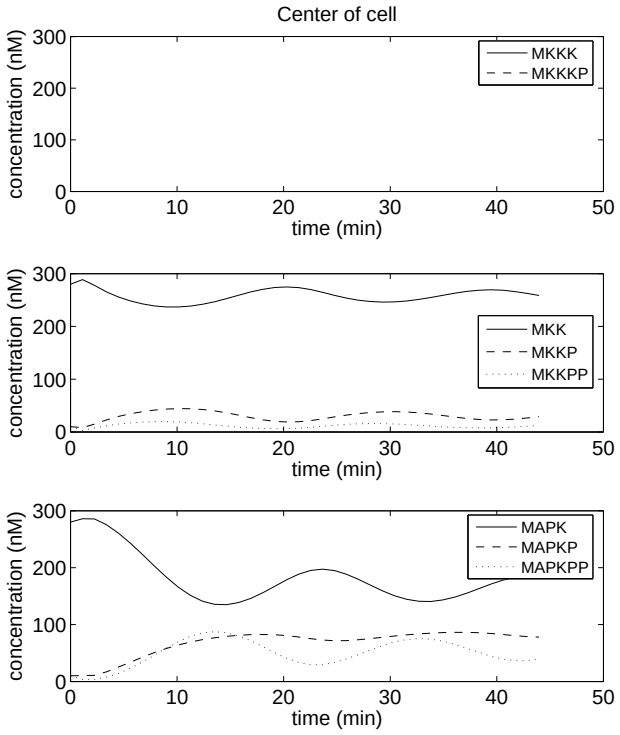

Figure 2.7: Concentrations for the eight components in the space-dependent model with slow diffusion $d = d_L$. The values are taken at the center of the cell.

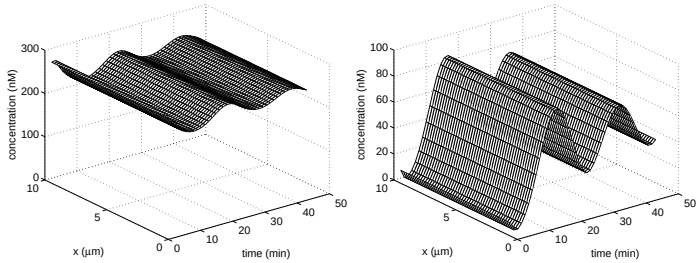

Figure 2.8: Concentrations of MKK (left) and MAPKPP (right) in the model with fast diffusion $d = d_H$. The values are taken on a line through the center of the cell for times up to about 50 minutes. The concentrations are almost equal throughout the cell.

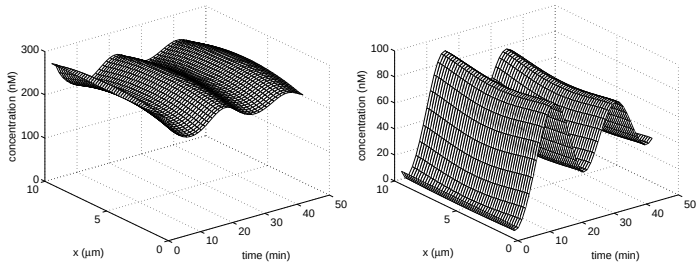

Figure 2.9: Concentrations of MKK (left) and MAPKPP (right) in the model with slow diffusion $d = d_L$. The values are taken on a line through the center of the cell for times up to about 50 minutes. The concentration gradient is clearly visible.

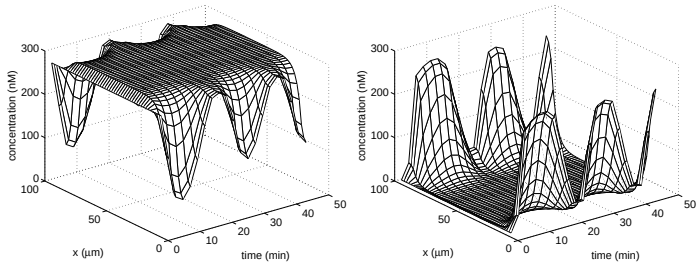

Figure 2.10: Again, concentrations of MKK and MAPKPP with slow diffusion $d = d_L$, but this time in a cell with diameter 90 µm. The values are taken on a line through center. Here, the phosphorylated proteins do not have time to diffuse far before the phosphate is removed, so the whole oscillation takes place at the membrane.

scaling coefficients h, k and u_0 with dimensions $[h] = \text{m}$, $[k] = \text{s}$ and $[u_0] = \text{M}$, we get

$$\frac{\partial \nu}{\partial \tau} = \Delta_\xi \nu - \frac{h^2 V_5}{d\, u_0} \frac{\nu}{K_5/u_0 + \nu},$$

with Δ_ξ denoting the Laplacian in the (ξ, η, ζ)-variables. The typical size of the diffusion term is now 1, while the typical size of the reaction term is $\lambda = h^2 V_5/(d\, u_0)$. This is the quantity that determines the influence of the reaction compared to the influence of diffusion. The values of the λ for the three examples discussed here are summarized in table 2.2. We see that in the first case $\lambda \ll 1$, indicating that reactions are slow compared

Example	h (µm)	d (µm^2/s)	V_5 (nM/s)	u_0 (nM)	λ
1	10	7.39	0.75	300	$3.38 \cdot 10^{-2}$
2	10	0.0739	0.75	300	3.38
3	100	0.0739	0.75	300	$3.38 \cdot 10^{2}$

Table 2.2: The ratio λ for the three examples.

to diffusion, so that molecules have time to travel across the cell before changing their phosphorylation state. This leads to homogeneous mixing of the diffusing components and no concentration gradients are observed. This is exactly what we see in figure 2.8 for the first example. For the second example, we have $\lambda \approx 1$ and diffusion and reaction are comparable, meaning that mixing of components is not complete. Thus we expect to see some concentration gradients, but also some diffusion of reaction products through the cell, which is what figure 2.9 shows. Finally, in the third example, $\lambda \gg 1$, meaning that reactions are much faster than diffusion, leading to highly localized reactions with small exchange of material. This is seen in figure 2.10.

2.5. Discussion

Much could be said about the Kholodenko model and whether oscillations really do occur in signaling pathways in real cells. One of the weakest points of the model is perhaps that the stimulus that activates the pathway is assumed to be "on" all the time, only being inhibited temporarily when the level of MAPKPP is high. This assumes that the cell never really responds to the signal, which is pretty absurd, since the purpose of the signal is to cause the cell to respond and adapt to new conditions. Once the cell has adapted, the signal must cease, or the cell will probably die or at least spend all its energy on useless tasks. So, it is maybe not so probable that we will observe oscillations in signaling pathways in real cells.

However, this numerical experiment has shown a number of other things of more general interest, which are easy to observe because oscillations is an effect that is easily studied. First, when we include the spatial distribution of proteins, we see that the oscillations are severely damped compared to the original model, almost independent of the diffusion coefficient as long as it is not too small. This shows that by neglecting diffusion and spatial distribution when modeling, one may overlook important aspects and draw false conclusions about the behavior of the system. Of course, the principal behavior is in large determined by the space-independent reaction terms, but our simulations show that the amplitude of the effect may be diminished significantly by the addition of diffusion. Furthermore, in our examples, the oscillations are not sustained, but seem to diminish with time, indicating that the system is damped by the diffusion. This is a different type of behavior than the sustained oscillations and is also an important thing to keep in mind when doing space-independent modeling of biochemical processes. The reason for this dependence on diffusion is that the problem is directly space-dependent, since we know that the first reactions take place only at the membrane, while the proteins at the end of the chain may move about freely. Thus, purely time-dependent modeling can be expected to produce errors for all signaling pathways, since these are space-dependent by nature, while for example space-independent metabolic models may be assumed to be more correct, since the space-dependence is not so obvious in that case.

There is of course a reason for not using full spatial modeling, in particular three-dimensional modeling, since the solving of systems of PDEs take so much more time than solving a system of ODEs. The images shown here are results of simulations that took 10 minutes or more. This should be compared to fractions of a second for solving the space-independent model. The amount of time needed to solve the space-dependent equations makes it impossible to use for example parameter fitting algorithms, since such algorithms require a large number of simulations with different parameter settings. Therefore, space-independent ODE models are of great importance, but one should be aware that one is neglecting something and that it may be worthwhile to see what happens in a space-dependent model.

3. Spatial modeling of the HOG pathway in yeast

3.1. The HOG pathway

One rather well-studied MAPK signaling pathway is the Hog1-pathway in the yeast *Saccharomyces cerevisiae*, where HOG stands for High Osmolarity Glycerol and the MAP kinase Hog1 is the last enzyme in the signaling chain, whose purpose is to sense that the osmotic pressure on the cell membrane increases and produce the appropriate response (see [4]). That is, if solutes (e.g. salt) are added to the solution outside the cell, the process called osmosis will strive to level out the difference in solute concentration over the membrane by forcing water to flow out of the cell. This is potentially harmful for the cell, since it then starts to shrink and if that goes on, the cell can not function anymore and eventually it will die.

To avoid this sad fate, the cell has to respond in some way to this new environment. It does so by starting to produce glycerol and accumulating it inside the cell, which evens out the solute concentrations and thereby prevents water from flowing out of the cell. And the link between the sensing of osmotic pressure and the response in the form of glycerol production is the HOG pathway. It is not really well known how the actual sensing of the change in osmotic pressure takes place, but there are enzymes at the cell membrane that are somehow activated, which leads to the activation of Hog1 through a few intermediate kinases. The pathway is shown in figure 3.1 with some of its surrounding components. When Hog1 has been activated, it enters the nucleus and once there it affects transcription of several genes through the transcription factors shown at the bottom of the figure.

A central feature of the signaling pathway is that Hog1 enters the nucleus. This can be viewed in the microscope by genetically adding a Green Fluorescent Protein (GFP) tag to the Hog1 protein, which makes the molecules shine bright green when viewed under ultraviolet light. This is a powerful method to view the localization of proteins in the cell and the results look like figure 3.2. One should be aware, however, that the GFP is a protein of about the same size as Hog1, so that the Hog1-GFP fusion is a much larger protein than wild-type Hog1, which could affect the kinetics and function of the protein. Still, the Hog1-GFP fusion is functional in the sense that it is able to carry out its function in the signaling pathway.

3.2. The model

Figure 3.3 shows the model that we are studying here. It is a model of the center part of the Hog1-pathway, where Hog1 itself is involved. The focus is on the transport of activated Hog1 into the nucleus. For simplicity, we assume that the MAPKK Pbs2 sits at the membrane in its phosphorylated form, ready to phosphorylate Hog1-molecules that come close to the membrane. This assumes that the upper part of the pathway that is not included in the model has reached an equilibrium and that phosphorylation of Hog1 can only take place at the membrane, which is believed to be true. There is also a small amount of spontaneous phosphorylation of Hog1 both in the nucleus and in the cytoplasm.

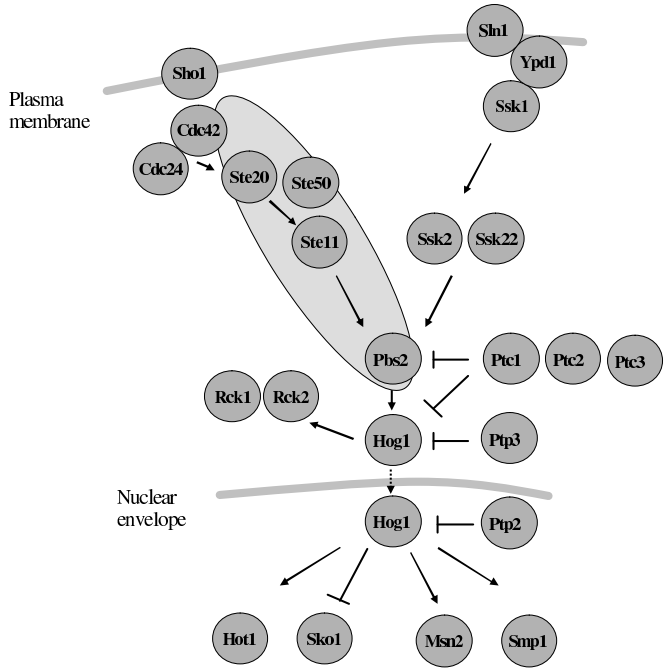

Figure 3.1: A schematic diagram showing the essentials of the HOG pathway, including the two different osmosensing mechanisms at the top, phosphatases on the right, cytoplasmic targets on the left and nuclear targets (transcription factors) at the bottom. Hog1 is seen at the center of it all and it is indicated that upon osmotic stress, it moves from the cytoplasm to the nucleus.

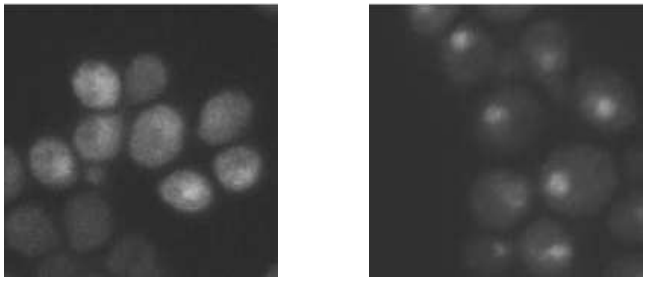

Figure 3.2: Hog1-GFP in wild-type yeast cells, under normal osmotic conditions on the left and after addition of NaCl on the right. The nuclei can be seen as bright spots in the right image, indicating that Hog1 has entered the nucleus. The positions of the nuclei may be verified by staining with a special dye (DAPI) (not shown). The images are generated by Claes Molin.

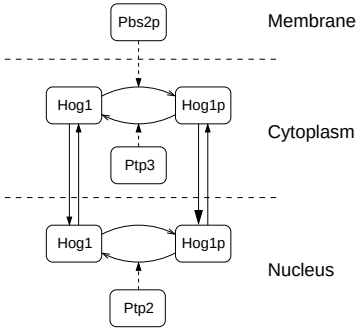

Figure 3.3: The model for the Hog1-pathway. The MAPKK Pbs2 is assumed fixed at the membrane in its phosphorylated form, meaning that the pathway is constantly active. The phosphatases Ptp2 and Ptp3 are distributed evenly throughout the nucleus and cytoplasm respectively. Hog1 is free to diffuse through the cell, but at the nuclear membrane, the transport is regulated. Unphosphorylated Hog1 is transported in and out of the nucleus at equal rates, while phosphorylated Hog1 (Hog1p) is transported into the nucleus at a higher rate than it is transported out.

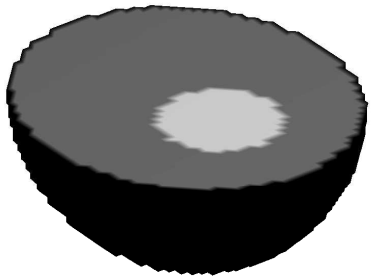

Figure 3.4: A cross section of the cell used in the model. The nucleus is shown as the bright spot.

The two phosphatases in the model, Ptp2 and Ptp3 are assumed to be evenly distributed in their respective domains, the nucleus and the cytoplasm. They are responsible for the dephosphorylation of Hog1. The only diffusing components of the model are thus Hog1 and Hog1p, which are free to move in the cytoplasm and in the nucleus, but not between the two compartments. The transport through the nuclear membrane is regulated so that unphosphorylated Hog1 is transported in and out of the nucleus at equal rates. Phosphorylated Hog1 (Hog1p) is transported out of the nucleus at the same rate, but transported into the nucleus at a much higher rate. The geometry used for the model is a simplified three-dimensional cell with an off-center spherical nucleus and a slightly ellipsoidal cell membrane. No other internal cell structure is included. A cross section of the model cell is shown in figure 3.4.

The model is of course a very simplified one. We disregard the fact that the phosphorylation of Hog1 takes place in two steps, so that the active form is double-phosphorylated. We also ignore the upper part of the pathway, the osmosensing mechanism and the phosphorylation of Pbs2. Also, the targets of Hog1 are not included in the model and there is no feedback loop to turn off the signaling pathway. The transport of proteins through the nuclear membrane is also simplified. Here, we use the assumption that the flux through the membrane is proportional to the concentration of proteins, which is of course a simplification of the rather complex transport system that shuttles the proteins in and out of the nucleus. And finally, the inner structure of the cell is not included other than in a reduction of the diffusion constant because of the obstacles.

We denote the concentrations of unphosphorylated and phosphorylated Hog1 by u_H and u_P, respectively, the nucleus by Ω_N, the cytoplasm by Ω_C and the nuclear and plasma membranes Γ_N and Γ_C. The model then leads to the following partial differential equations

$$\begin{aligned}
\partial u_H/\partial t &= d_H \Delta u_H - f_C(u_H, u_P), && \text{in } \Omega_C,\\
\partial u_P/\partial t &= d_P \Delta u_P + f_C(u_H, u_P), && \text{in } \Omega_C,\\
\partial u_H/\partial t &= d_H \Delta u_H - f_N(u_H, u_P), && \text{in } \Omega_N,\\
\partial u_P/\partial t &= d_P \Delta u_P + f_N(u_H, u_P), && \text{in } \Omega_N,\\
\partial u_H/\partial n &= 0, && \text{on } \Gamma_C\\
\partial u_P/\partial n &= 0, && \text{on } \Gamma_C\\
d_H \partial u_H/\partial n &= a_H u_H^{(N)} - b_H u_H^{(C)}, && \text{on } \Gamma_N\\
d_P \partial u_P/\partial n &= a_P u_P^{(N)} - b_P u_P^{(C)}, && \text{on } \Gamma_N
\end{aligned}$$

with

$$\begin{aligned}
f_C(u_H, u_P) &= \frac{V_{Pbs2} u_H}{K_{Pbs2} + u_H} - \frac{V_{Ptp3} u_P}{K_{Ptp3} + u_P} + k_{spC} u_H\\
f_N(u_H, u_P) &= -\frac{V_{Ptp2} u_P}{K_{Ptp2} + u_P} + k_{spN} u_H.
\end{aligned}$$

Here, d_H and d_P are diffusion constants, which we assume to be equal. The four first equations are diffusion-reaction equations with one diffusion term and one reaction term, the two first describe the cytoplasm and the two last the nucleus. The reaction terms make use of Michaelis-Menten kinetics (see e.g. [2, chapter 4]) for the enzyme reactions, which is a standard way of modeling such reactions. Next follows boundary conditions at the plasma membrane, which say that the flux through the membrane is zero, i.e. that no protein molecules may leave the cell. The two last rows are boundary conditions at the nuclear membrane, both saying that the flux out through the membrane equals a constant times the nuclear concentration at the membrane minus a constant times the cytoplasmic concentration on the outside of the membrane. We also need initial conditions that describe the concentrations at time $t = 0$. These are shown in figure 3.5 and are chosen so that the system is near its equilibrium. Most of the Hog1 molecules are in the unphosphorylated state and are evenly distributed between nucleus and cytoplasm. A small fraction of the molecules are phosphorylated and these have a higher concentration in the nucleus than in the cytoplasm.

Table 3.1 shows the parameter values used for the simulations. The diffusion coefficients are taken to be $d_0/1000$, where d_0 is the diffusion coefficient of Hog1 in water, estimated from equation (2.1). The reaction coefficients are chosen in the same range as for the Kholodenko model, but their exact values are quite arbitrarily chosen to get a result which resembles the *in vivo* behavior. The parameters a and b describing the efficiency of the nuclear transport are also rather arbitrarily chosen to get a reasonable result and are not based on experiments. For comparison, one simulation was also done with faster diffusion ($d_H = d_P = d_0/10$).

d_H	$0.074\mu\text{m}^2/\text{s}$	d_P	$0.074\mu\text{m}^2/\text{s}$
a_H	$1\ d_H\ \mu\text{m}/\text{s}$	a_P	$1\ d_P\ \mu\text{m}/\text{s}$
b_H	$1\ d_H\ \mu\text{m}/\text{s}$	b_P	$3\ d_P\ \mu\text{m}/\text{s}$
V_{Pbs2}	1.0 nM/s	K_{Pbs2}	50.0 nM
V_{Ptp3}	0.2 nM/s	K_{Ptp3}	15.0 nM
V_{Ptp2}	0.2 nM/s	K_{Ptp2}	15.0 nM
k_{spC}	$0.0002\ \text{s}^{-1}$	k_{spN}	$0.0002\ \text{s}^{-1}$

Table 3.1: Parameter values

The equations were solved using the Immersed Interface Method and finite differences on uniform three-dimensional grids with $54 \times 54 \times 54$ nodes. The method is described in Part II of this thesis. The final simulations shown here, involving some 200 time steps, took up to 30 minutes to complete on a standard computer.

3.3. Results

Figure 3.5 shows the initial conditions used in the 3D simulations. The images only show the concentrations in a slice through the center of the cell, but the concentrations are assumed to be uniform throughout the nucleus and the cytoplasm. The values chosen are near the equilibrium of the system when the pathway is not activated. The exact values used are 100 nM for Hog1 and 1.5 nM and 3.2 nM for Hog1p in cytoplasm and nucleus respectively, but changing these values slightly does not alter the behavior of the system much, so the exact values are not very important.

Then at time $t = 0$, the signaling is turned on, in this model by activating Pbs2, that is setting V_{Pbs2} to a non-zero value. In figure 3.6 we see the simulated response as it would appear in the fluorescence microscope, namely the total concentration of Hog1 ([Hog1] + [Hog1p]) at the start and the end of the simulation. The simulation ends at 22 minutes when the concentrations are approaching the steady-state levels with the signal turned on. In a real cell the level of phosphorylation would again turn down when the cell started to adapt, but since this adaptation is not included in the model, the concentrations just approach a steady state with high concentration in the nucleus, mostly containing phosphorylated Hog1 and lower concentrations outside.

To see the time evolution of the system, we show in figure 3.7 concentrations along a line through the center of the cell for all times and for Hog1 and Hog1p separately. We see that the levels of Hog1p are gradually increasing and that Hog1p is transported into the nucleus, while unphosphorylated Hog1 is moved out of the nucleus as the levels become lower in the cytoplasm when Hog1 is phosphorylated. One also notes that there are concentration gradients in the cytoplasm, arising from the fact that the diffusion is not fast enough to even out the concentrations at the same rate that the reactions change them. The dimensionless parameter λ defined in section 2.4 is here

$$\lambda = \frac{h^2 V_{Pbs2}}{d\, u_0} \approx \frac{(10^{-5})^2 \cdot 1.0}{7.4 \cdot 10^{-14} \cdot 100} \approx 13.5,$$

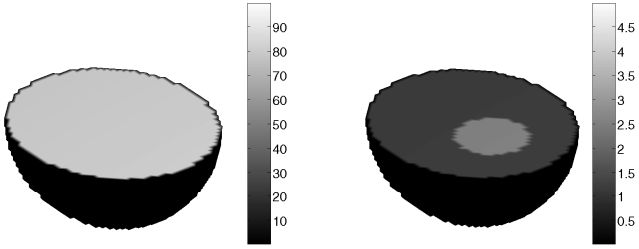

Figure 3.5: Initial conditions for the simulation in 3D, with unphosphorylated Hog1 on the left and phosphorylated Hog1 (Hog1p) on the right. The images show a slice through the center of the cell, with the nucleus visible in the right image. Note the different scales on the two images.

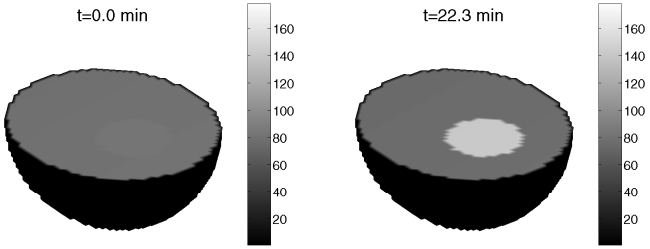

Figure 3.6: The total Hog1 concentration ([Hog1] + [Hog1p]) at the start (left) and end (right) of simulation. These images are comparable to the ones seen in the fluorescence microscope. The ragged edges are artifacts arising from the space discretisation.

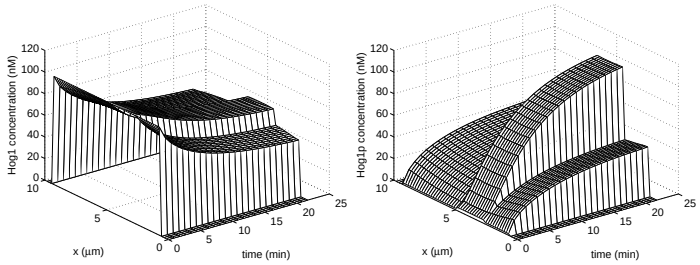

Figure 3.7: Concentrations of Hog1 (left) and Hog1p (right) for slow diffusion. The values are taken along a line through the center of the cell and the nucleus. The nuclear relocation of Hog1p is clearly seen. There is a slight concentration gradient in the cytoplasm.

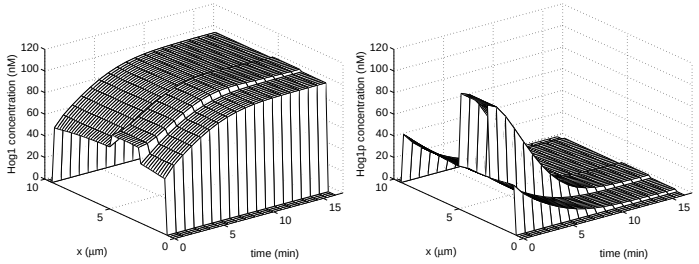

Figure 3.8: Concentrations of Hog1 (left) and Hog1p (right) for slow diffusion, when the signal is turned off after Hog1p has accumulated in the nucleus. The concentrations return to their original state.

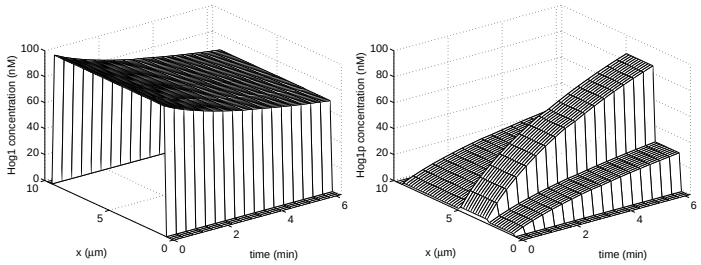

Figure 3.9: Concentrations of Hog1 (left) and Hog1p (right) for fast diffusion. Here the concentration gradient is no longer present and the response is slightly faster than in the example with slow diffusion.

indicating that reactions are slightly faster than diffusion, which should produce gradients in the concentration levels.

In figure 3.8, we also show the return to the original state when the signaling pathway is turned off. Starting at the final levels of the previous simulation, and setting $V_{Pbs2} = 0$, we see that the system returns to its original state with low levels of phosphorylated Hog1 and almost equal concentrations in nucleus and cytoplasm.

Finally, we also show a simulation with a larger diffusion coefficient ($d_H = d_P = d_0/10 \approx 7.4\mu m^2/s$). The results are shown in figure 3.9. Now, diffusion is fast enough to level out the concentrations, so that they are uniform in nucleus and cytoplasm, respectively. The response in the nucleus is also slightly faster than in the slow diffusion simulation.

3.4. Discussion

The results shown above show that it is possible to reproduce in simulations the nuclear relocation of Hog1 that is observed when yeast cells are exposed to osmotic stress. The model used is a very simple one, including only two diffusing components and only the final steps of the phosphorylation chain. The results should therefore not be used to draw far-reaching conclusions about the actual behavior of the cell.

What we see, however, is that only with the small diffusion coefficient $d_0/1000$ do we observe concentration gradients and interesting spatial effects. This value for the diffusion coefficient is much smaller than the values observed for similar proteins inside living cells (up to 10-fold reduction compared to diffusion in water [10]). For the larger diffusion coefficient $d_0/10$, which is more in agreement with observed values, concentrations are practically constant in each of the compartments. This indicates that diffusion is fast compared to the reactions involved, so that full spatial modeling with PDEs might not be of crucial importance for the simple model with the parameter values used here. However,

the behavior of the system is very much dependent on the enzyme rates, especially for the phosphatases Ptp2 and Ptp3. The values for the reaction rates used here are not based on direct measurements, but are taken to be near the values in the Kholodenko model, which are based on *in vitro* measurements for MAP kinases in mammalian cells. Those values may not be correct for the reactions in our model. So, we can not determine from our simple model whether the explicit modeling of spatial effects shown here is of crucial importance. In any case, one must remember that cell signaling is a spatial phenomenon, so that spatial features should be taken into account, in one way or another. What we have developed is a method to simulate full spatial models that could be used to compare the output with results from simpler, less computationally intense methods.

In the end, what decides whether a biological model is good or bad, is if it compares well with measurements of the real phenomenon in live cells. So, to be certain about what type of models are needed, one should compare the results to precise measurements. This, however, is not an easy task. The types of measurements available at present to determine spatial distributions of proteins inside the cell are fluorescence microscopy experiments, yielding images such as those in figure 3.2. In order to compare these to the results of the computations made here, one would like to follow a single cell and take images at intervals of seconds or less, which is very hard to do. Furthermore, one would ideally like to be able to view phosphorylated and unphosphorylated Hog1 separately, which is not possible at present.

One may also discuss if a diffusion PDE is an appropriate model for the transport of proteins inside the cell. First of all, the number of molecules is relatively small. The estimated number of Hog1 molecules in the yeast cell is about 7000 [3], giving a concentration of around 50 nM. Since the diffusion equation is obtained as the limit when the number of molecules tend to infinity, it may not be an accurate model. The random fluctuations arising from the small number of molecules may also influence the reactions. However, as long as there is no data to compare to, it is hard to decide between models. But it would be interesting to investigate the differences between a PDE model and a stochastic model for the signaling pathway.

Another issue related to diffusion is that the cell is full of obstacles, consisting of organelles, cytoskeleton networks, large protein complexes etc. This makes the assumptions underlying the diffusion model invalid. It has been shown (see [11], [10], [7]) that these obstacles do not only cause a reduction in the diffusion coefficient, but may give rise to the phenomenon called anomalous (sub)diffusion. For anomalous diffusion, the mean square displacement of the molecules is no longer proportional to time ($\langle x(t)^2 \rangle \propto t$), but to some power $\alpha < 1$ of time ($\langle x(t)^2 \rangle \propto t^\alpha$). This may give rise to new interesting phenomena and is also a subject that would be interesting to study.

References

[1] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter. *Molecular Biology of the Cell*. Garland Science, 4th edition, 2002.

[2] C. P. Fall, E. S. Marland, J. M. Wagner, and J. J. Tyson, editors. *Computational Cell Biology*. Springer, 2002.

[3] S. Ghaemmaghami, W.-K. Huh, K. Bower, R. W. Howson, A. Belle, N. Dephoure, E. K. O'Shea, and J. S. Weissman. Global analysis of protein expression in yeast. *Nature*, 425:737 – 741, Oct 2003.

[4] S. Hohmann. Osmotic stress signaling and osmoadaptation in yeasts. *Microbiol Mol Biol Rev*, 66(2):300–372, 2002.

[5] C.-Y. F. Huang and J. E. Ferrell, Jr. Ultrasensitivity in the mitogen-activated protein kinase cascade. *Proc. Natl. Acad. Sci. USA*, sept 1996.

[6] B. N. Kholodenko. Negative feedback and ultrasensitivity can bring about oscillations in the mitogen-activated protein kinase cascades. *European Journal of Biochemistry*, 267:1583–1588, 2000.

[7] M. J. Saxton. Anomalous diffusion due to obstacles: a Monte Carlo study. *Biophys J.*, 66(2:1):394–401, 1994.

[8] I. Swameye, T. G. Müller, J. Timmer, O. Sandra, and U. Klingmüller. Identification of nucleocytoplasmic cycling as a remote sensor in cellular signaling by databased modeling. *PNAS*, 100(3):1028–33, 2003.

[9] G. A. Truskey, F. Yuan, and D. F. Katz. *Transport Phenomena in Biological Systems: A Textbook for Biomedical Engineers*. Prentice Hall, 2003.

[10] M. Wachsmuth, W. Waldeck, and J. Langowski. Anomalous diffusion of fluorescent probes inside living cell nuclei investigated by spatially-resolved fluorescence correlation spectroscopy. *J. Mol. Biol.*, 298:677–689, 2000.

[11] M. Weiss, H. Hashimoto, and T. Nilsson. Anomalous protein diffusion in living cells as seen by fluorescence correlation spectroscopy. *Biophys J.*, 84(6):4043–52, 2003.

[12] H. S. Wiley, S. Y. Shvartsman, and D. A. Lauffenburger. Computational modeling of the EGF-receptor system: a paradigm for systems biology. *Trends Cell Biol.*, 13(1):43–50, 2003.

The Immersed Interface Method on uniform and Boolean grids

Tobias Gebäck

Abstract

The Immersed Interface Method (IIM) is a method which allows the use of finite differences in non-rectangular domains, by immersing interfaces into a rectangular domain which is discretised by a uniform grid. The finite differences near the interface are then corrected using the size and position of the jumps in the solution and its derivatives across the interface.

The method presented here is the Explicit Jump IIM developed by Wiegmann and Bube, but we also present some additional details on how to apply the method in three dimensions, using Robin boundary conditions and to time-dependent problems.

We then apply the IIM to Boolean grids. These are grids that use combinations of a number of Cartesian grids to achieve greater accuracy of approximations, while using fewer grid nodes. The use of IIM on such grids requires some new development of the estimation of jumps on the boundary. We show numerically that the resulting Boolean IIM gives second order error convergence with respect to the smallest step size in the grid, meaning that the required number of nodes needed for a given maximal error is considerably smaller than on uniform Cartesian grids.

Contents

1 The Immersed Interface Method **1**
- 1.1 Introduction . . . 1
- 1.2 EJIIM theory . . . 1
- 1.3 Estimating jumps at boundaries . . . 4
- 1.4 Applying boundary conditions . . . 6
- 1.5 Solving the linear system . . . 9
- 1.6 Other equations . . . 10
- 1.7 Examples . . . 10

2 The Immersed Interface Method on Boolean grids **13**
- 2.1 Introduction . . . 13
- 2.2 Boolean interpolation . . . 13
  - 2.2.1 The algebraic theory . . . 13
  - 2.2.2 Boolean grids . . . 15
  - 2.2.3 Application to finite difference solvers . . . 21
- 2.3 Applying IIM to Boolean grids . . . 22
  - 2.3.1 Boolean approximation of jumps . . . 23
- 2.4 Examples . . . 27
  - 2.4.1 Example 1 – Boolean approximation . . . 27
  - 2.4.2 Example 2 – Finite differences on Boolean grids . . . 28
  - 2.4.3 Example 3 – IIM on Boolean grids . . . 31

References **34**

1. The Immersed Interface Method

1.1. Introduction

The use of finite difference methods for solving partial differential equations has a few advantages, mainly that they are easy to implement and that they may be easily and quickly solved using Fast Fourier Transform (FFT) methods. The drawbacks are that these methods demand a rectangular domain with a uniform grid and also that it is not straightforward to obtain accuracy estimates and convergence results.

The immersed interface method overcomes the first of these problems, in that it allows the solution and its derivatives to be discontinuous along interfaces. This makes it possible to immerse a boundary into a rectangular grid, let the solution be zero outside the boundary and apply boundary conditions to the solution along the boundary. The finite differences near the boundary are then corrected using the jumps in the solution at the boundary, so that the differences remain valid even though the solution is discontinuous. Therefore, the method makes it unnecessary to spend time on making grids adapted to the geometry and also allows the use of FFT-based methods even though the domain is not rectangular.

The ideas behind the immersed interface method were first conceived by Peskin [13] and used in computations on heart flows, with moving boundaries. The method was further developed by LeVeque and Li [11], [12] and used with finite differences on Cartesian grids. Finally, Wiegmann and Bube [16], [15] gave the method a clearer formulation and extended it to more general problems. They call their method the "explicit jump immersed interface method" or EJIIM, and that is the method we will be concerned with here.

We will first describe the general idea behind the EJIIM and then proceed to the problem of implementing the method. This will include treatment of three-dimensional problems and time-dependent problems, which is not included in the original article. We will also discuss some additional boundary conditions which were not discussed by Wiegmann and Bube.

This presentation contains no proofs of convergence or error estimates. Wiegmann and Bube provide proofs of convergence for the method in one dimension and also in two dimensions for the special case when the jumps at the boundary are known beforehand. Other proofs or error estimates are not known and here we only confirm numerical convergence.

1.2. EJIIM theory

The basic idea of the IIM is that standard finite differences, such as

$$u_{xx}(x_i) = \frac{u(x_i+h) - 2u(x_i) + u(x_i-h)}{h^2} + O(h^2) \tag{1.1}$$

are not valid for non-smooth functions, since they are based on Taylor-expansions. However, they may be corrected using the size and position of the discontinuities in u and its derivatives. Let us denote the jump in the m:th derivative in $u : \mathbb{R} \to \mathbb{R}$ at a point $\alpha \in \mathbb{R}$

by

$$[u^{(m)}]_\alpha = \lim_{x\to\alpha^+} u^{(m)}(x) - \lim_{x\to\alpha^-} u^{(m)}(x),$$

where, of course, $u^{(0)} = u$.

Following [16], we give two lemmas that contain the essence of the EJIIM. The proofs are essentially exercises in the use of Taylor expansions and we refer the reader to the original article [16].

Lemma 1.1 (cf. [16, lemma 1]) *Let $h > 0$ and assume $u^- \in C^{l+1}([\alpha-h, \alpha])$ and $u^+ \in C^{l+1}([\alpha, \alpha+h])$. Then let*

$$u(x) = \begin{cases} u^-(x) & \text{for } x \le \alpha, \\ u^+(x) & \text{for } x > \alpha. \end{cases}$$

For $x \in (\alpha-h, \alpha)$, we then have

$$u(x+h) = \sum_{k=0}^{l} \frac{h^k}{k!} u^{(k)}(x) + \sum_{k=0}^{l} \frac{(x+h-\alpha)^k}{k!} [u^{(k)}]_\alpha + O(h^{l+1}),$$

and

$$u(x) = \sum_{k=0}^{l} \frac{(-h)^k}{k!} u^{(k)}(x+h) - \sum_{k=0}^{l} \frac{(x-\alpha)^k}{k!} [u^{(k)}]_\alpha + O(h^{l+1}),$$

Using this lemma in the expressions for finite differences on a uniform 1d-grid $\{x_j\}$ with grid spacing h, we get the next lemma.

Lemma 1.2 (cf. [16, lemma 3]) *Let $x_j \le \alpha < x_{j+1}$ and suppose $u \in C^4([x_j-h, \alpha)) \cap C^4((\alpha, x_j+h])$, with derivatives extending continuously to α. Then the following approximations hold:*

$$u_x(x_j) = \frac{u(x_{j+1}) - u(x_{j-1})}{2h} - \frac{1}{2h} \sum_{m=0}^{2} \frac{(x_{j+1}-\alpha)^m}{m!} [u^{(m)}]_\alpha + O(h^2), \tag{1.2}$$

$$u_x(x_{j+1}) = \frac{u(x_{j+2}) - u(x_j)}{2h} - \frac{1}{2h} \sum_{m=0}^{2} \frac{(x_j-\alpha)^m}{m!} [u^{(m)}]_\alpha + O(h^2), \tag{1.3}$$

$$u_{xx}(x_j) = \frac{u(x_{j+1}) - 2u(x_j) + u(x_{j-1})}{h^2} - \frac{1}{h^2} \sum_{m=0}^{3} \frac{(x_{j+1}-\alpha)^m}{m!} [u^{(m)}]_\alpha + O(h^2), \tag{1.4}$$

$$u_{xx}(x_{j+1}) = \frac{u(x_{j+2}) - 2u(x_{j+1}) + u(x_j)}{h^2} + \frac{1}{h^2} \sum_{m=0}^{3} \frac{(x_j-\alpha)^m}{m!} [u^{(m)}]_\alpha + O(h^2). \tag{1.5}$$

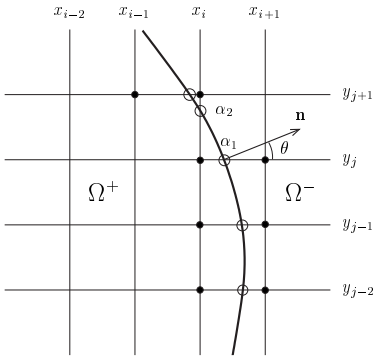

Figure 1.1: Immersed boundary in 2D. The immersed boundary intersects the grid at points $\alpha_1 = (x_{\alpha_1}, y_{\alpha_1})$ and $\alpha_2 = (x_{\alpha_2}, y_{\alpha_2})$. Interface intersection points (IIPs) are marked by '$\circ$' and anchor points are marked by '$\bullet$'.

The application of these results to higher dimensions is now straightforward. In the situation in figure 1.1, for example, the Laplacian at (x_i, y_j) would be approximated as

$$\begin{aligned}\Delta u(x_i, y_j) = &\frac{u(x_{i+1}, y_j) + u(x_{i-1}, y_j) + u(x_i, y_{j+1}) + u(x_i, y_{j-1}) - 4u(x_i, y_j)}{h^2} \\ &+ \frac{1}{h^2}\sum_{m=0}^{3}\frac{(x_{i+1} - x_{\alpha_1})^m}{m!}[u_x^{(m)}]_{\alpha_1} + \frac{1}{h^2}\sum_{m=0}^{3}\frac{(y_{j+1} - y_{\alpha_2})^m}{m!}[u_y^{(m)}]_{\alpha_2} + O(h^2).\end{aligned} \tag{1.6}$$

Here,

$$[u^{(m)}]_{\alpha_k} = u^{(m),+}(\alpha_k) - u^{(m),-}(\alpha_k) = \lim_{\substack{(x,y)\to\alpha_k \\ (x,y)\in\Omega^+}} u^{(m)}(x,y) - \lim_{\substack{(x,y)\to\alpha_k \\ (x,y)\in\Omega^-}} u^{(m)}(x,y),$$

with $\alpha_k = (x_{\alpha_k}, y_{\alpha_k})$, $k = 1, 2$, which in this case gives a different sign for the jump compared to the previous definition (for the one-dimensional case), accounting for the plus signs for the corrections in (1.6), while there is a minus sign in (1.4). This new definition makes it unnecessary to keep track of coordinate directions at the interface when adding corrections.

This means that on a uniform grid with an immersed boundary, the discrete Laplacian may be calculated as

$$\Delta_h U + \Psi C, \tag{1.7}$$

where Δ_h is the standard $N \times N$ finite difference matrix for the Laplacian, U is the vector of function values at the grid points, Ψ contains the coefficients for the correction terms from (1.6) and $C = ([u]_{\alpha_1}, [u^{(1)}]_{\alpha_1}, [u^{(2)}]_{\alpha_1}, \ldots)^T$ is a vector containing the jumps of u and its derivatives at the points where the interface intersects the grid. So (1.6) corresponds to one row of (1.7).

We will now consider the problem of solving the Poisson equation

$$\Delta u(x) = f(x) \qquad x \in \Omega^+, \tag{1.8}$$

where Ω^+ is a domain in $\mathbb{R}^2$ or $\mathbb{R}^3$. We wish to apply boundary conditions of the following types

$$\begin{aligned}[u](x) &= g(x), \\ \left[\frac{\partial u}{\partial n}\right](x) &= h(x), \\ \left[\frac{\partial u}{\partial n}\right](x) &= au^+(x) - bu^-(x),\end{aligned} \tag{1.9}$$

for $x \in \partial\Omega^+$, that is Dirichlet, Neumann or Robin boundary conditions.

In order to apply the EJIIM, we let the domain Ω^+ be embedded in a rectangular area discretised by a uniform grid and let Ω^- denote the domain outside the interface $\partial\Omega^+$. Setting $f(x) = 0$ in Ω^- and thus $u^- = 0$, the above boundary conditions become the regular Dirichlet, Neumann and Robin boundary conditions for u^+ in Ω^+. The corresponding linear system of equations becomes

$$\Delta_h U + \Psi C = F.$$

The problem here is that the jumps C are unknown, so we need to find an additional relation that specifies these jumps. Some of the jumps are determined by the boundary conditions at $\Gamma = \partial\Omega^+$, while others must be determined in another way. This is done by creating an interpolation matrix D^T, which, given the function values, estimates the jumps at the boundary. Thus the entire equation system may be written

$$\begin{aligned}\Delta_h U + \Psi C &= F_1, \\ C &= D^T U + F_2,\end{aligned} \tag{1.10}$$

where F_1 contains function values of the right hand side f and F_2 contains the known jumps derived from boundary conditions.

1.3. Estimating jumps at boundaries

The jumps at the boundary are estimated by Lagrange interpolation, meaning that for each interface intersection point (IIP), i.e. for each point where the interface intersects the grid (see figure 1.1), we select a number of grid nodes on one side of the boundary and calculate the interpolating polynomial of degree d, say. This polynomial and its derivatives

are then evaluated at the boundary. Doing the same from the other side of the interface, we may take the difference and get an estimate of the jump at the IIP, given function values at the grid nodes. The number of grid nodes needed to construct a polynomial of degree d in n dimensions is $\binom{n+d}{d}$.

As an example, let us consider a second order polynomial in two dimensions (i.e. $d = n = 2$). We select $\binom{4}{2} = 6$ grid nodes $p_i = (x_i, y_i)$, $i = 1, \ldots, 6$ with corresponding function values $u_1, \ldots, u_6$ and set up the equation system

$$\begin{bmatrix} 1 & x_1 & y_1 & x_1^2 & x_1y_1 & y_1^2 \\ 1 & x_2 & y_2 & x_2^2 & x_2y_2 & y_2^2 \\ 1 & x_3 & y_3 & x_3^2 & x_3y_3 & y_3^2 \\ 1 & x_4 & y_4 & x_4^2 & x_4y_4 & y_4^2 \\ 1 & x_5 & y_5 & x_5^2 & x_5y_5 & y_5^2 \\ 1 & x_6 & y_6 & x_6^2 & x_6y_6 & y_6^2 \end{bmatrix} \begin{bmatrix} a_{00} \\ a_{10} \\ a_{01} \\ a_{20} \\ a_{11} \\ a_{02} \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix} \tag{1.11}$$

in order to construct the polynomial

$$\mathcal{P}_2(x, y) = a_{00} + a_{10}x + a_{01}y + a_{20}x^2 + a_{11}xy + a_{02}y^2.$$

We wish to solve this system for the coefficients a_{jk}, which leads to the requirement that the determinant of the 6×6-matrix (the so called Vandermonde determinant) must be non-zero. This in turn leads to restrictions on the choice of the grid nodes p_i. These restrictions were studied by Chui and Lai [4] for arbitrary n and d. They give a sufficient criterion for selection of nodes so that the Vandermonde determinant is non-zero, which they call *Node Configuration A*.

Definition 1.1 (Node Configuration A in $\mathbb{R}$) *Any set of distinct points in $\mathbb{R}$ satisfies Node Configuration A in $\mathbb{R}$.*

Definition 1.2 (Node Configuration A in $\mathbb{R}^n$) *Let $\hat{X}_d^n = \{x_1, \ldots, x_{N_d^n}\}$ be a set of $N_d^n = \binom{n+d}{d}$ distinct points in $\mathbb{R}^n$. $\hat{X}_d^n$ satisfies Node Configuration A in $\mathbb{R}^n$ if there exist $d+1$ hyperplanes K_i^n, $i = 0, \ldots, d$ with*

$$x_{N_{d-1}^n+1}, \ldots, x_{N_d^n} \in K_d^n$$

and

$$x_{N_{j-1}^n+1}, \ldots, x_{N_j^n} \in K_j^n \setminus (K_{j+1}^n \cup \ldots \cup K_d^n)$$

for $j = 0, \ldots, d-1$, and such that each set of points

$$\hat{X}_j^{n-1} = \{x_{N_{j-1}^n+1}, \ldots, x_{N_j^n}\}, \qquad 0 \le j \le d,$$

viewed as points in $\mathbb{R}^{n-1}$ satisfies Node Configuration A in $\mathbb{R}^{n-1}$.

Theorem 1.3 (cf. [4, theorem 4]) *If $\hat{X}_d^n$, with $n \ge 1$ and $d \ge 0$, satisfies Node Configuration A, then the corresponding Vandermonde determinant is non-zero.*

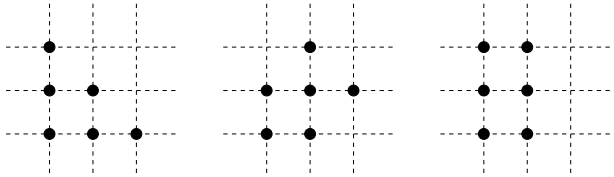

Figure 1.2: Examples of node configurations in two dimensions. The two on the left satisfy *Node Configuration A*, while the one on the right does not.

In the definition we use the convention $N_{-1}^n = 0$ for all n. Also note that $(N_j^n - N_{j-1}^n) = N_j^{n-1}$, so that the $(n-1)$-dimensional hyperplane K_j^n contains N_j^{n-1} points, which by an orthogonal transformation may be regarded as points in $\mathbb{R}^{n-1}$ for which the node configuration problem could again be posed. Therefore *Node Configuration A* is well defined.

The recursive definition of the node configuration may seem hard to interpret in all its generality, so let us see what it means in our above example with $n = 2$ and $d = 2$. In that case, the hyperplanes become lines, and the definition says that we should be able to select $d + 1 = 3$ lines, such that $N_2^1 = \binom{1+2}{2} = 3$ points lie on the first one, $N_1^1 = \binom{1+1}{1} = 2$ on the second (but not on the first) and $N_0^1 = \binom{1+0}{0} = 1$ on the third. Furthermore the set of points on each of these lines should satisfy *Node Configuration A*, which they do automatically, since they are on one-dimensional sets. Two valid node configurations and one invalid are shown in figure 1.2.

In three dimensions, the hyperplanes become planes, and for a second order polynomial, we need to be able to choose 3 planes, containing 6, 3 and 1 nodes respectively, making $\binom{3+2}{2} = 10$ nodes in total. And on each of the planes, the nodes should satisfy *Node Configuration A* in two dimensions.

1.4. Applying boundary conditions

Having seen how to create interpolating polynomials, we return to the problem of estimating the jumps in the vector C in (1.10), i.e. we need to determine the matrix D^T and the vector F_2 in the equation

$$C = D^T U + F_2.$$

Here, we do this in three dimensions, since the two-dimensional problem is presented in [16], while three-dimensional problems are not treated. In order to determine D^T, we first find all the interface intersection points (IIPs). For each of these points, there is a coordinate direction $X_j = x, y$ or z, which is the direction of the grid line on which the

IIP is located. Along this grid line, there are two anchor points, one on each side of the interface, which are the grid nodes closest to the IIP and whose finite difference stencils are affected by the discontinuity at the IIP (see figure 1.1). We denote these p_+ and p_-.

Now, for each IIP α_j, we see from lemma 1.2 that we need to estimate the jumps

$$[u]_j = u^+(\alpha_j) - u^-(\alpha_j), \quad [u_{X_j}]_j = u^+_{X_j}(\alpha_j) - u^-_{X_j}(\alpha_j), \quad [u_{X_jX_j}]_j = u^+_{X_jX_j}(\alpha_j) - u^-_{X_jX_j}(\alpha_j)$$

in order to get accurate finite difference approximations of order $O(h)$. We will see from numerical results that it is in fact enough to have $O(h)$ approximations at the boundary in order to get overall $O(h^2)$-convergence. This behavior is confirmed for all IIM methods and may be loosely motivated by the fact that the boundary is a lower-dimensional set so that in three dimensions, for example, the number of points with $O(h)$ errors is $O(h^{-2})$ rather than $O(h^{-3})$. One may also use $O(h^2)$ estimates at the boundary. This still gives overall $O(h^2)$ errors for the solution, although the actual errors may be smaller (see [16, example 1]).

In order to approximate the jumps, we select stencils of grid nodes around each of the anchor points according to *Node Configuration A* in the previous section and define matrices $\mathcal{P}_{j+}$ and $\mathcal{P}_{j-}$ in analogy with the matrix in (1.11), using coordinates relative to p_+ and p_-, respectively. We also define restriction matrices $\mathcal{R}_{j+}$ and $\mathcal{R}_{j-}$, consisting of only 1's and 0's and serving only to select and reorder the grid nodes used in the stencils. In this way, we see that $\mathcal{P}^{-1}_{j+}\mathcal{R}_{j+}U$ is a vector containing the coefficients of an interpolating polynomial near α_j on the +-side of the interface. Using this we may estimate

$$\begin{bmatrix} u^+(\alpha_j) \\ u^+_x(\alpha_j) \\ u^+_{xx}(\alpha_j) \end{bmatrix} \approx \begin{bmatrix} 1 & h^+_x & h^+_y & h^+_z & (h^+_x)^2 & (h^+_y)^2 & (h^+_z)^2 & h^+_x h^+_y & h^+_x h^+_z & h^+_y h^+_z \\ 0 & 1 & 0 & 0 & 2h^+_x & 0 & 0 & h^+_y & h^+_z & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \mathcal{P}^{-1}_{j+}\mathcal{R}_{j+}U,$$

where we have set $h^+_x = x_{\alpha_j} - x_{p_+}$, etc, and with similar expressions for derivatives in other coordinate directions. Naming the matrix of coefficients on the left $\mathcal{Q}_{j+}$, we may now write

$$\begin{bmatrix} u^+(\alpha_j) \\ u^+_x(\alpha_j) \\ \vdots \\ u^-_z(\alpha_j) \\ u^-_{zz}(\alpha_j) \end{bmatrix} \approx \begin{bmatrix} \mathcal{Q}_{j+} & 0 \\ 0 & \mathcal{Q}_{j-} \end{bmatrix} \begin{bmatrix} \mathcal{P}^{-1}_{j+} & 0 \\ 0 & \mathcal{P}^{-1}_{j-} \end{bmatrix} \begin{bmatrix} \mathcal{R}_{j+} \\ \mathcal{R}_{j-} \end{bmatrix} U \equiv \mathcal{Q}_j\mathcal{P}_j^{-1}\mathcal{R}_j U. \tag{1.12}$$

It is now time to incorporate the boundary conditions (1.9) into the equation for C. This is done by creating a matrix $\mathcal{L}_j$ and a vector $F_{2,j}$. The vector $F_{2,j}$ contains the known jump information derived from the boundary conditions, while the matrix $\mathcal{L}_j$ defines linear combinations of estimated function values at α_j giving the contributions to the jumps that have to be estimated from the solution U.

In the case where Ω^+ is immersed into a rectangular domain and $u = 0$ outside Ω^+, then of course $u^- = 0$ everywhere along with its derivatives, so it is not necessary to estimate

$u^-(\alpha_j)$. However, by describing the method in this more general context, it may also be applied when the interface is an interface between two domains where u is nonzero.

In the case of Dirichlet boundary conditions, this is very simple. Suppose we have a boundary condition that says $[u] = g(x)$ on the interface. Then, if for example the grid intersects the interface in the x-direction at α_j, we write

$$\begin{bmatrix} [u]_j \\ [u_x]_j \\ [u_{xx}]_j \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 & -1 & \cdots & 0 \end{bmatrix} \mathcal{Q}_j\mathcal{P}_j^{-1}\mathcal{R}_j U + \begin{bmatrix} g(\alpha_j) \\ 0 \\ 0 \end{bmatrix}, \tag{1.13}$$

where $F_{2,j}$ is the vector on the right and $\mathcal{L}_j$ is the leftmost matrix, so that we get

$$\begin{bmatrix} [u]_j \\ [u_x]_j \\ [u_{xx}]_j \end{bmatrix} = \mathcal{L}_j\mathcal{Q}_j\mathcal{P}_j^{-1}\mathcal{R}_j U + F_{2,j} = \begin{bmatrix} g(\alpha_j) \\ u^+_x(\alpha_j) - u^-_x(\alpha_j) \\ u^+_{xx}(\alpha_j) - u^-_{xx}(\alpha_j) \end{bmatrix}.$$

Now setting $D_j^T = \mathcal{L}_j\mathcal{Q}_j\mathcal{P}_j^{-1}\mathcal{R}_j$, we have

$$C_j = D_j^T U + F_{2,j}. \tag{1.14}$$

By just stacking the contributions from all IIPs, we arrive at (1.10).

For other boundary conditions, things are more complicated. Consider for example Neumann conditions, $\partial u/\partial n = h(x)$ on the interface. Then we are interested in derivatives in the coordinate directions, while the boundary condition is given in the normal direction of the interface. We define the normal direction through the two angles (θ, φ), $0 \le \theta \le \pi$, $0 \le \varphi < 2\pi$, as in spherical coordinates and introduce a new Cartesian coordinate system (ξ, η, ζ) which is oriented so that the ξ-axis points in the normal direction, the η-axis is perpendicular to the ξ-axis and lies in the (x, y)-plane, and the ζ-axis is perpendicular to the other two and has positive z-component. This coordinate tranformation yields

$$\begin{bmatrix} [u_x]_j \\ [u_y]_j \\ [u_z]_j \end{bmatrix} = \begin{bmatrix} \sin\theta\cos\varphi & -\sin\varphi & -\cos\theta\cos\varphi \\ \sin\theta\sin\varphi & \cos\varphi & -\cos\theta\sin\varphi \\ \cos\theta & 0 & \sin\theta \end{bmatrix} \begin{bmatrix} [u_\xi]_j \\ [u_\eta]_j \\ [u_\zeta]_j \end{bmatrix} \tag{1.15}$$

$$\begin{bmatrix} [u_\xi]_j \\ [u_\eta]_j \\ [u_\zeta]_j \end{bmatrix} = \begin{bmatrix} \sin\theta\cos\varphi & \sin\theta\sin\varphi & \cos\theta \\ -\sin\varphi & \cos\varphi & 0 \\ -\cos\theta\cos\varphi & -\cos\theta\sin\varphi & \sin\theta \end{bmatrix} \begin{bmatrix} [u_x]_j \\ [u_y]_j \\ [u_z]_j \end{bmatrix} \tag{1.16}$$

By the boundary condition, we know that $[u_\xi]_j = h(\alpha_j)$, so using (1.15) we get $[u_x]_j$, $[u_y]_j$ and $[u_z]_j$ expressed in $h(\alpha_j)$, $[u_\eta]_j$ and $[u_\zeta]_j$, which, using (1.16) may be expressed in the quantities $u^+_x(\alpha_j) - u^-_x(\alpha_j)$, $u^+_y(\alpha_j) - u^-_y(\alpha_j)$ and $u^+_z(\alpha_j) - u^-_z(\alpha_j)$, which we get from (1.12). Thus, we get for example

$$\begin{aligned}[u_x]_j = {} & \sin\theta\cos\varphi\, h(\alpha_j) + (\sin^2\varphi + \cos^2\theta\cos^2\varphi)(u^+_x(\alpha_j) - u^-_x(\alpha_j)) \\ & + \sin\varphi\cos\varphi\,(\cos^2\theta - 1)(u^+_y(\alpha_j) - u^-_y(\alpha_j)) - \cos\theta\sin\theta\cos\varphi\,(u^+_z(\alpha_j) - u^-_z(\alpha_j)),\end{aligned}$$

where the first term is known and therefore included in $F_{2,j}$, while the coefficients of the second term are included in $\mathcal{L}_j$, so that we get (1.14) once more. The expressions for $[u_y]_j$ and $[u_z]_j$ are derived similarly, while $[u]_j = u^+(\alpha_j) - u^-(\alpha_j)$ and $[u_{X_jX_j}]_j = u^+_{X_jX_j}(\alpha_j) - u^-_{X_jX_j}(\alpha_j)$ for $i = 1, 2, 3$.

Finally, we consider Robin boundary conditions, i.e. $\partial u/\partial n = au^+ - bu^-$. Using the above notation, this leads to the expression

$$u^+_\xi = u^-_\xi = au^+ - bu^- = a[u] - (b-a)u^-,$$

which gives us

$$[u]_j = \frac{1}{a}u^+_\xi(\alpha_j) + (\frac{b}{a} - 1)u^-(\alpha_j).$$

Furthermore, we need to impose the implicit condition that $u^+_\xi = u^-_\xi$, which we impose by setting $h(\alpha_j) = 0$ in the Neumann case above. In summary, this leads to the above expression for $[u]_j$, expressions similar to the Neumann case (but with $F_{2,j} = 0$) for $[u_{X_j}]_j$ and the same expressions as before for $[u_{X_jX_j}]_j$.

This way, we construct the sparse matrix D^T and the vector F_2 by stacking the contributions from each IIP. If we have multiple interfaces with different boundary conditions, we just include the IIPs from all the interfaces and use the corresponding boundary condition for each IIP according to the above treatment.

Finally, we need to construct the sparse matrix Ψ in (1.10). This matrix contains the coefficients of the jumps in (1.2) to (1.5), placed at appropriate positions, so that the corrections affect the corresponding anchor points. Since the corrections are additive, the Ψ-matrix is easy to construct.

We also remark that it is possible to apply other types of boundary conditions, such as in composite material problems illustrated in [16], by the same method as shown here.

1.5. Solving the linear system

Through the previous sections, we have arrived at the system of equations

$$\begin{aligned} \Delta_h U + \Psi C &= F_1, \\ C &= D^T U + F_2, \end{aligned}$$

which is (1.10). Solving for U in the first equation and inserting this into the second, we get

$$\begin{aligned} U &= \Delta_h^{-1}(-\Psi C + F_1), \\ (I + D^T\Delta_h^{-1}\Psi)C &= D^T\Delta_h^{-1}F_1 + F_2. \end{aligned}$$

Here, the second equation is a 'small' system of equations for C with $3N_{IIP}$ rows (where N_{IIP} is the number of IIPs). It may be solved using iterative methods to avoid forming the matrix on the left explicitly, which would be unfeasibly memory-consuming for three-dimensional problems, since the matrix is not sparse. We use the BiCGStab method (see

e.g. [10], [2]) to solve the system and since we then only need to be able to form the matrix-vector product $(I + D^T\Delta_h^{-1}\Psi)C$, our only remaining problem is to apply Δ_h^{-1}. Since our domain is embedded in a rectangular uniform grid, we may use the Fast Fourier Transform to achieve this efficiently (see e.g. [14, chapter 19]). If N is the total number of grid nodes, the FFT is applied in $O(N \log N)$ operations and this is the most time-consuming part of each iteration in the BiCGStab algorithm. We have used the library FFTW [1] to get a fast implementation of the FFT in C.

Having solved for C, all that remains is to compute U from the first equation by applying Δ_h^{-1} once more.

1.6. Other equations

The immersed interface method may also be applied to other types of equations. First of all, it is straightforward to apply it to the heat equation

$$u_t(x,t) - d\Delta u(x,t) = f(x,t), \qquad x \in \Omega^+,\ t \geq 0,$$

with suitable boundary conditions, using for example an implicit Euler scheme in time. In that case we get

$$\begin{aligned} (I/\Delta t - d\Delta_h)U_{n+1} - d\Psi C_{n+1} &= U_n/\Delta t + F_1, \\ C_{n+1} &= D^T U_{n+1} + F_2, \end{aligned}$$

and

$$\begin{aligned} U_{n+1} &= (I/\Delta t - d\Delta_h)^{-1}(d\Psi C_{n+1} + U_n/\Delta t + F_1), \\ (I - D^T(I/\Delta t - d\Delta_h)^{-1}d\Psi)C_{n+1} &= D^T(I/\Delta t - d\Delta_h)^{-1}(F_1 + U_n/\Delta t) + F_2, \end{aligned}$$

where the operator $(I/\Delta t - d\Delta_h)^{-1}$ may be applied using FFT and the second equation solved using BiCGStab for each time step.

It is also possible to solve equations in multiple domains with boundary conditions between the domains or with different coefficients in the equation in different domains. The technique is exactly as described in the previous sections. See section 1.7 and [16] for examples.

Other combinations of spatial derivatives than the Laplacian may also be considered, although they are not discussed here. It should be clear, however, from the discussion above how the immersed interface method should be applied to these cases.

It is also in principle possible to use the method for moving boundary problems, where one would like to avoid costly grid generation. It would then, however, be necessary to compute the matrices D^T and Ψ at every time step.

1.7. Examples

Our first example is solving the Poisson equation inside an ellipsoid E with half-axes 0.44, 0.3 and 0.3, centered at $(0.5, 0.5, 0.5)$. The equation we solve is given in spherical

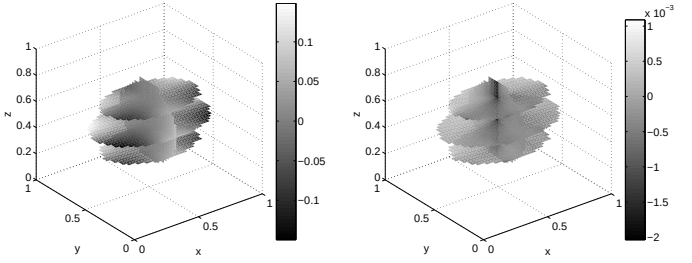

Figure 1.3: EJIIM for the Poisson equation inside an ellipsoid in 3D. The solution was computed on a $40 \times 40 \times 40$ grid and is shown on the left. Errors compared to the exact solution are shown on the right. Values are shown on selected slices through the domain.

coordinates as

$$\begin{aligned} \Delta u(r,\theta,\varphi) &= f(r,\theta,\varphi), & (r,\theta,\varphi) \in E, \\ u(r,\theta,\varphi) &= g(r,\theta,\varphi), & (r,\theta,\varphi) \in \partial E, \end{aligned}$$

where $f(r,\theta,\varphi) = -5\sin(3\varphi)$ and $g(r,\theta,\varphi) = r^2\sin^2\theta\sin(3\varphi)$, which gives the solution $u(r,\theta,\varphi) = r^2\sin^2\theta\sin(3\varphi)$ in E. The solution and the errors are shown in figure 1.3. We see that this non-trivial problem can be solved with reasonable accuracy, even on the rather coarse $40 \times 40 \times 40$ grid used here. The solution took less than a second to compute on a standard computer. The errors are of order $O(h^2)$, which is shown more clearly for another example in section 2.4.3.

The second example shows that more complicated equations and boundary conditions may be solved using the immersed interface method. We solve the heat equation in two dimensions. The domain consists of two concentric circular discs, C_1 and C_2, with C_1 inside C_2. We apply Neumann boundary conditions at ∂C_2 and Robin boundary conditions at ∂C_1. The problem may be written as

$$\begin{aligned} \frac{\partial u}{\partial t}(x,y,t) - \Delta u(x,y,t) &= 0, & (x,y) \in C_1 \cup C_2,\ t \geq 0, \\ \frac{\partial u}{\partial n}(x,y,t) &= 0, & (x,y) \in \partial C_2,\ t \geq 0, \\ \frac{\partial u}{\partial n}(x,y,t) &= 10u_2(x,y) - 3u_1(x,y), & (x,y) \in \partial C_1,\ t \geq 0, \\ u(x,y,0) &= u_0(x,y), & (x,y) \in C_1 \cup C_2. \end{aligned}$$

Here, $u_0(x,y)$ is a bell-shaped function inside C_1 shown at the top of figure 1.4. u_2 and u_1 are the values of u at ∂C_1, on the outer and inner side of the boundary.

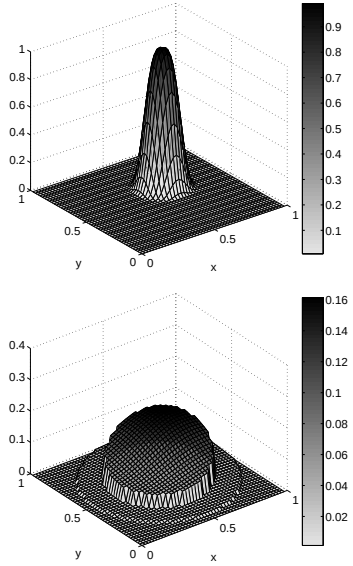

Figure 1.4: EJIIM for the heat equation inside two concentric discs. The initial data is shown at the top and the solution at $t = 0.03$ below. The domain boundaries are clearly visible in the second figure.

The method used is the one described in section 1.6 with implicit Euler time stepping. The boundary conditions are implemented as described earlier.

Obviously, the exact solution is not known for this problem, so we have nothing to compare to. However, the example shows that it is possible to solve multiple boundary, time-dependent problems with a range of boundary conditions using the immersed interface method. Here, for simplicity, we have used only circular and elliptical domains, but in the method itself there are no limitations on the shape of the domain. Further examples are given by Wiegmann and Bube [16] and in part I of this thesis.

2. The Immersed Interface Method on Boolean grids

2.1. Introduction

There is a constant desire to make computations as fast and efficient as possible, in order to be able to solve larger problems with higher accuracy, or just to minimize the time spent waiting for the computer to carry out the calculations. The Boolean grids presented here provide one method to make computations faster by using less data, while still obtaining the same accuracy in the calculations.

The Boolean methods were originally developed in the 1960's in order to represent surfaces used in computer-aided design (CAD). They were first used by Coons [5] to create interpolatory surfaces, coinciding with prescribed values on the boundary of the unit square. The theory was then developed in a series of articles by Gordon [8], [7], resulting in an abstract theory of commutative projectors which is presented here in section 2.2.1. There is also a book by Delvos [6], where the methods are presented in some detail.

As we will see, it is straightforward to use finite differences on Boolean grids and one may use FFT-based methods to solve PDEs on such grids, gaining several orders of magnitude in the number of points needed for a given accuracy. However, just as for uniform grids, these methods can only be applied to rectangular regions. Therefore it is interesting to apply the Immersed Interface Method from the previous section to Boolean grids in order to get a similar decrease in computational time even for problems in irregularly shaped domains.

The outline of our presentation is as follows. In section 2.2, we present the abstract theory of Boolean interpolation and construct Boolean grids on which we apply finite differences and the immersed interface method. In section 2.3, we discuss how to extend the immersed interface method to work on the Boolean grids and finally, in section 2.4, we give some numerical examples of the use of Boolean grids for interpolation and equation solving.

As for the IIM on uniform grids, we do not have any proofs of convergence or error estimates for the IIM on Boolean grids. We only confirm numerical convergence and superiority to the uniform IIM.

2.2. Boolean interpolation

2.2.1. The algebraic theory

The following presentation is taken from Gordon [7] and is an abstract algebraic approach to approximation theory, giving a motivation for the use of Boolean approximations.

We consider an arbitrary function space $\mathcal{F}$. On this space, we define M projectors P_j, $j = 1, \ldots, M$, meaning that $P_j : \mathcal{F} \to \Phi_j$ is a linear transformation with the property $P_jP_j = P_j$. Here Φ_j is a subspace of $\mathcal{F}$ for $j = 1, \ldots, M$.

A function $\pi = P_jf \in \Phi_j$ is called the approximation of $f \in \mathcal{F}$, and the function $f - \pi$ is called the remainder.

We define multiplication and addition of projectors in the natural way and note that the associative and distributive rules hold for projectors defined on the same domain $\mathcal{F}$. Furthermore, we assume that multiplication is commutative, i.e.

$$P_jP_k = P_kP_j \quad \text{for all } j, k = 1, \ldots, M.$$

It is obvious that the product of two commutative projectors, $A = P_jP_k$, is again a projector, since $AA = A$. This is not true, however, for the sum of two projectors, since

$$(P_j + P_k)(P_j + P_k) = P_jP_j + P_jP_k + P_kP_j + P_kP_k = P_j + P_k + 2P_jP_k \neq P_j + P_k.$$

Therefore, we introduce the *Boolean addition*, denoted by $\oplus$ and defined by

$$P_j \oplus P_k = P_j + P_k - P_jP_k. \tag{2.1}$$

It is easy to check that $P_j \oplus P_k$ is again a projector.

We also need to compare projectors in order to decide which are better than others. To this end, we introduce the ordering relation $\leq$ defined by

$$P_j \leq P_k \Leftrightarrow P_jP_k = P_j, \tag{2.2}$$

that is, if $P_j \leq P_k$, P_j removes at least as much of the function f as P_k does, or $\Phi_j \subset \Phi_k$.

Now, we may define the space Ψ as the set of all projectors which can be built up as combinations of the P_j, $j = 1, \ldots, M$, under the operations of multiplication and Boolean addition. The set Ψ is now a *distributive lattice* under the partial ordering $\leq$. This means that for all projectors $A, B, C \in \Psi$, the following properties hold (and are easily checked):

i)	reflexivity	$A \leq A$
ii)	anti-symmetry	$A \leq B$ and $B \leq A \Rightarrow A = B$
iii)	transitivity	$A \leq B$ and $B \leq C \Rightarrow A \leq C$
iv)	idempotence	$A \oplus A = A$ and $AA = A$
v)	commutativity	$A \oplus B = B \oplus A$ and $AB = BA$
vi)	associativity	$A(BC) = (AB)C$ and $A \oplus (B \oplus C) = (A \oplus B) \oplus C$
vii)	distributivity	$A(B \oplus C) = AB \oplus AC$ and $A \oplus (BC) = (A \oplus B)(A \oplus C)$
viii)	consistency	$A \leq B \Leftrightarrow AB = A \Leftrightarrow A \oplus B = B$

(2.3)

It is a property of every lattice that any pair $\{A, B\}$ of elements has both a *least upper bound* (denoted sup), that is the least element C such that $C \geq A$ and $C \geq B$, and a *greatest lower bound* (denoted inf), that is the largest element D such that $D \leq A$ and $D \leq B$. These are given explicitly by

$$\sup\{A, B\} = A \oplus B,$$
$$\inf\{A, B\} = AB.$$

It follows that every finite lattice has a unique *maximal element* $\mathcal{M} \in \Psi$, that is an element satisfying $A \leq \mathcal{M}$ for all $A \in \Psi$, and a unique *minimal element* $\mathcal{L}$, for which $\mathcal{L} \leq A$ for all $A \in \Psi$. It is easily seen that

$$\mathcal{M} = \sup\{P_j\}_{j=1}^{M} = P_1 \oplus P_2 \oplus \cdots \oplus P_M,$$
$$\mathcal{L} = \inf\{P_j\}_{j=1}^{M} = P_1 P_2 \cdots P_M.$$

It is also interesting to study the range of the projectors in Ψ. It is clear that the range of $P_j \oplus P_k$ is $\Phi_j \cup \Phi_k$ and that the range of $P_j P_k$ is $\Phi_j \cap \Phi_k$. Thus, the range of $\mathcal{M}$ is the largest space formed from the Φ_j:s, namely $\Phi_1 \cup \cdots \cup \Phi_M$ and the range of $\mathcal{L}$ is the smallest space, $\Phi_1 \cap \cdots \cap \Phi_M$.

Finally, we introduce the *remainder operator* or the complement of a projector A, namely

$$A' \equiv I - A,$$

where I is the identity operator. It is clear that A' is a projector, and that

$$AA' = A'A = 0, \qquad A \oplus A' = I.$$

Now for each P_j, we set $R_j \equiv P_j' = I - P_j$ and note that although in general $R_j \notin \Psi$, the set of combinations of these remainder operators also form a distributive lattice, denoted Ψ'. Also, if we allow the three operators multiplication, Boolean addition and complement to work on the projectors, we generate a Boolean algebra, where additionally de Morgan's laws hold

$$(A \oplus B)' = A'B', \qquad (AB)' = A' \oplus B'.$$

Now, the final statement is that given a commutative set of projectors $\{P_j\}_{j=1}^{M}$, the identity operator, I has a *maximal decomposition*

$$I = \mathcal{M} \oplus \mathcal{M}' = \mathcal{M} + \mathcal{M}' = P_1 \oplus \cdots \oplus P_M + (R_1 R_2 \cdots R_M), \tag{2.4}$$

and a *minimal decomposition*

$$I = \mathcal{L} \oplus \mathcal{L}' = \mathcal{L} + \mathcal{L}' = P_1 P_2 \cdots P_M + (R_1 \oplus \cdots \oplus R_M). \tag{2.5}$$

Here, $R_1 R_2 \cdots R_M = \inf\{R_j\}_{j=1}^{M} \in \Psi'$ and $R_1 \oplus \cdots \oplus R_M = \sup\{R_j\}_{j=1}^{M} \in \Psi'$ as before. This means that by choosing the algebraically maximal projector $\mathcal{M}$, we minimize the remainder and vice versa.

2.2.2. Boolean grids

We now wish to apply the abstract results from the previous section to the problem of creating a grid on which we wish to apply finite difference solvers. This could be done in many ways, depending on how one defines the projectors $P_1, \ldots, P_M$. Here, we only discuss the grids that we actually use and the reason for using those will be clear later on.

First, assume a two-dimensional problem, where we want to approximate the smooth function $f(x, y)$ on a rectangle $\Omega = [a_x, b_x] \times [a_y, b_y]$. We introduce step sizes $h_{1,x} = (b_x - a_x)/(N_{1,x} - 1)$ and $h_{1,y} = (b_y - a_y)/(N_{1,y} - 1)$, interpolation points $\{x_i\}_{i=1}^{N_{1,x}}$, $\{y_i\}_{j=1}^{N_{1,y}}$, with $x_i = a_x + h_{1,x}(i-1)$, $y_j = a_y + h_{1,y}(j-1)$, and define the interpolating projectors P_x^1 and P_y^1 by

$$[P_x^1 f](x, y) = \sum_{i=1}^{N_{1,x}} f(x_i, y)\varphi_i(x),$$
$$[P_y^1 f](x, y) = \sum_{j=1}^{N_{1,y}} f(x, y_j)\psi_j(y),$$

where $\varphi_i(x)$ and $\psi_j(y)$ are piecewise linear hat functions, satisfying $\varphi_i(x_k) = \delta_{ik}$ and $\psi_j(y_k) = \delta_{jk}$. Using these two projectors, we may define the algebraically maximal and minimal projectors as

$$\mathcal{M}_{xy} = P_x^1 \oplus P_y^1 = P_x^1 + P_y^1 - P_x^1 P_y^1,$$
$$\mathcal{L}_{xy} = P_x^1 P_y^1.$$

We see that the minimal projector $\mathcal{L}_{xy}$ is the regular interpolation operator, that interpolates $f(x, y)$ only on the nodes (x_i, y_j). The maximal projector, however, is different. It interpolates $f(x, y)$ along the lines $x = x_i$ and $y = y_j$ and since the values at the nodes (x_i, y_j) are included both by P_x^1 and P_y^1, one occurrence is removed by subtracting $P_x^1 P_y^1$.

Since $P_x^1 f$ is piecewise linear in x, we see that $P_x^1 f$ approximates $f(x, y)$ to an accuracy of $O(h_{1,x}^2)$, that is

$$R_x^1 f = (I - P_x^1) f = O(h_{1,x}^2).$$

This obviously holds for P_y^1 too, and we may therefore deduce from (2.4) that the remainder of the maximal projector $\mathcal{M}_{xy}$ is given by

$$\mathcal{M}_{xy}' f = (I - \mathcal{M}_{xy}) f = R_x^1 R_y^1 f = O(h_{1,x}^2 h_{1,y}^2),$$

while for the minimal projector the remainder is

$$\mathcal{L}_{xy}' f = (I - \mathcal{L}_{xy}) f = (R_x^1 \oplus R_y^1) f = (R_x^1 + R_y^1 - R_x^1 R_y^1) f = O(h_{1,x}^2 + h_{1,y}^2).$$

We see that the maximal projector has a fourth order error, while the standard interpolation operator, which is minimal, has a second order error. This can also be easily checked by using Taylor expansions of f. One then sees that for the maximal operator, the second order terms are eliminated by subtracting the term $P_x^1 P_y^1 f$, so that only fourth and higher order terms remain.

There is, however, a difference between $\mathcal{M}_{xy}$ and $\mathcal{L}_{xy}$ in that $\mathcal{M}_{xy} f$ is not fully discretised, since it still has a continuous variable along the lines $x = x_i$ and $y = y_j$. But along these lines, we can afford a finer discretisation. We introduce new (smaller) step sizes $h_{2,x}$

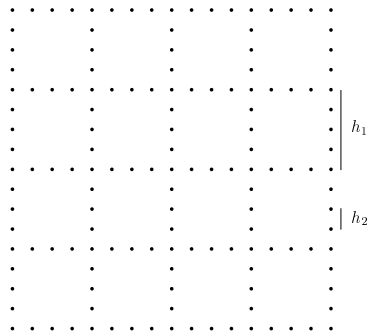

Figure 2.1: The two-dimensional Boolean grid with step sizes h_1 and h_2. The grid consists of three parts, the horizontal lines (G_{21}), the vertical lines (G_{12}) and the coarse grid (G_{11}) consisting of the intersection nodes of these lines. These three are combined as in (2.6) below to create the Boolean approximation.

and $h_{2,y}$ and projectors P_x^2 and P_y^2, identical to P^1 except for the step size. Since the error in $P_x^2 f$ is of order $h_{2,x}^2$, we see that to preserve the $O(h_{1,x}^2 h_{1,y}^2)$ error estimate, we should set

$$h_{2,x} = h_{2,y} = h_{1,x} h_{1,y},$$

or $h_2 = h_1^2$ if the step sizes are equal in the two directions. This results in a grid like the one in figure 2.1. It is important to ensure that the nodes for P^1 are a subset of the nodes for P^2, so that P^1 and P^2 commute and that $P^1 \leq P^2$.

The resulting projector may be expressed in several ways, which are all equal, as can be shown by direct calculation using the rules (2.3) and the fact that $P_x^1 \leq P_x^2$ and $P_y^1 \leq P_y^2$. Denoting the resulting projector by $\mathcal{P}_{xy}$, we have

$$\mathcal{P}_{xy} = P_x^1 P_y^2 \oplus P_x^2 P_y^1 = P_x^2 P_y^2 (P_x^1 \oplus P_y^1) = P_x^2 P_y^1 + P_x^1 P_y^2 - P_x^1 P_y^1. \tag{2.6}$$

The last form shows the three uniform grids which we must combine in order to get the Boolean approximation, namely the grids G_{21} with steps $h_{2,x}$ and $h_{1,y}$, G_{12} with steps $h_{1,x}$ and $h_{2,y}$ and the coarse grid G_{11} with steps $h_{1,x}$ and $h_{1,y}$. Using the second form of $\mathcal{P}_{xy}$ and de Morgan's laws, we may also express the remainder as

$$\mathcal{R}_{xy} = R_x^2 \oplus R_y^2 \oplus R_x^1 R_y^1 = R_x^2 + R_y^2 + R_x^1 R_y^1 - R_x^2 R_y^1 - R_x^1 R_y^2,$$

where we again see that the resulting error will be of order $O(h_2^2 + h_1^4)$, so that a choice of $h_2 = h_1^2$ is optimal.

There is another way to view the error cancellation property of the Boolean grids, as is shown by Bungartz et al. [3]. This view is based on an error splitting for the approximation of the form

$$u^{h_x,h_y} - u = e_x(h_x) + e_y(h_y) + R(h_x, h_y),$$

where e_x only depends on h_x, x and y, e_y depends only on h_y, x and y and $|R(h_x, h_y)| \leq c(h_x h_y)^\nu$ for some constants c and ν independent of x and y. We can now study combinations as the ones above, namely

$$\hat{u}^{h_x,h_y} = u^{\alpha h_x,h_y} + u^{h_x,\alpha h_y} - u^{h_x,h_y}$$

for some $\alpha > 0$. Then, the resulting error is

$$\begin{aligned}\hat{u}^{h_x,h_y} - u &= e_x(\alpha h_x) + e_y(h_y) + R(\alpha h_x, h_y) + e_x(h_x) + e_y(\alpha h_y) + R(h_x, \alpha h_y) \\ &\quad - e_x(h_x) - e_y(h_y) - R(h_x, h_y) \\ &= e_x(\alpha h_x) + e_y(\alpha h_y) + \widehat{R}(h_x, h_y).\end{aligned}$$

Now, if e_x and e_y are the dominating terms and α is small, then we have reduced the error. For example, if $e_x(h_x) = O(h_x^2)$, $e_x(h_y) = O(h_y^2)$ and $\nu = 2$ as is the case for the projections above, and if $\alpha = h_x = h_y$, then $\hat{u}^{h,h} - u = e_x(h^2) + e_y(h^2) + \widehat{R}(h, h) = O(h^4)$, while $u^{h,h} - u = e_x(h) + e_y(h) + \widehat{R}(h, h) = O(h^2)$ exactly as above. This idea may be pursued further, taking combinations of a series of grids to eliminate errors while using as few points as possible. This *combination technique* was introduced by Griebel et al. [9] and is summarized in [3].

Assuming $N_{1,x} = N_{1,y} = N_1$ and $N_{2,x} = N_{2,y} = N_2$, the number of nodes used in the Boolean approximation (2.6) is

$$N_B = 2N_1 N_2 + N_1^2 = O(N_1^3),$$

if we choose N_2 proportional to N_1^2. This should be compared to the case of a standard grid, where the number of points needed to achieve the same order of accuracy is

$$N_S = N^2 = N_2^2 = O(N_1^4),$$

since we must choose N of the same magnitude as N_2 in order to get an error of order h_2^2. We therefore see that by using a Boolean grid, we save one order of magnitude in N_1 while retaining the same accuracy of approximation.

So far for two-dimensional problems. In higher dimensions, one would expect even greater gains and we will now derive an approximation scheme and show that it is indeed so. We consider the same problem in three dimensions. As before, given a smooth function f defined on a box $\Omega = [a_x, b_x] \times [a_y, b_y] \times [a_z, b_z] \in \mathbb{R}^3$, we define the projector P_x^1 by

$$[P_x^1 f](x, y, z) = \sum_{i=1}^{N_x} f(x_i, y, z)\varphi_i(x),$$

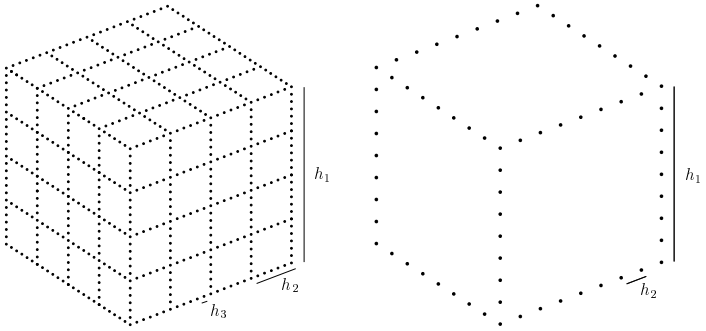

Figure 2.2: The building blocks of Boolean grids in 3D. The maximal grid corresponding to $\mathcal{P}^{\mathcal{M}}_{xyz}$ is shown on the left and the intermediate grid of $\mathcal{P}_{xyz}$ on the right. The maximal grid is created by using a two-dimensional Boolean grid on each of the sides of the cube and is a combination of 13 different grids. The intermediate grid is created by discretising lines in each coordinate direction, and consists of only four different grids.

where φ_i are hat-functions as before, $x_i = a_x + (i-1)h_{1,x}$ and $h_{1,x} = (b_x - a_x)/(N_x - 1)$. We similarly define $P^1_{y'}$ P^1_z, $h_{1,y}$, $h_{1,z}$, N_y and N_z. It is clear that $P^1_x f$ interpolates f on the planes $x = x_i$ and that the error introduced by this approximation is $O(h_x^2)$. Now, to make a full Boolean approximation, we would make an approximation on each of the planes $x = x_i$ using the two-dimensional Boolean grid introduced above, resulting in a projection for this direction given by

$$\mathcal{P}^{\mathcal{M}}_x = (P^2_y P^3_z \oplus P^2_z P^3_y) P^1_x,$$

where we have introduced the projectors P^2_α and P^3_α for $\alpha = y, z$, corresponding to the step sizes $h_{2,\alpha}$ and $h_{3,\alpha}$, respectively. Defining $\mathcal{P}^{\mathcal{M}}_y$ and $\mathcal{P}^{\mathcal{M}}_z$ similarly and taking the Boolean sum, we get the final Boolean projector in three dimensions,

$$\begin{aligned}\mathcal{P}^{\mathcal{M}}_{xyz} &= \mathcal{P}^{\mathcal{M}}_x \oplus \mathcal{P}^{\mathcal{M}}_y \oplus \mathcal{P}^{\mathcal{M}}_z \\ &= (P^2_y P^3_z \oplus P^2_z P^3_y) P^1_x \oplus (P^2_z P^3_x \oplus P^2_x P^3_z) P^1_y \oplus (P^2_x P^3_y \oplus P^2_y P^3_x) P^1_z \\ &= P^{123}_{xyz} \oplus P^{132}_{xyz} \oplus P^{213}_{xyz} \oplus P^{231}_{xyz} \oplus P^{312}_{xyz} \oplus P^{321}_{xyz}, \end{aligned} \tag{2.7}$$

where we have used the notation $P^{ijk}_{xyz} = P^i_x P^j_y P^k_z$. This would result in a grid made up of cubes like the one on the left of figure 2.2. In order to estimate the error of this projection, we study the remainder operator

$$\mathcal{R}^{\mathcal{M}}_{xyz} = I - \mathcal{P}^{\mathcal{M}}_{xyz} = R^{123}_{xyz} R^{132}_{xyz} R^{213}_{xyz} R^{231}_{xyz} R^{312}_{xyz} R^{321}_{xyz},$$

with

$$R^{123}_{xyz} = R^1_x \oplus R^2_y \oplus R^3_z = R^1_x + R^2_y + R^3_z - R^1_x R^2_y - R^1_x R^3_z - R^2_y R^3_z + R^1_x R^2_y R^3_z$$

and so on. Using this in the expression for $\mathcal{R}^{\mathcal{M}}_{xyz}$, and remembering that $R^i_\alpha \geq R^j_\alpha$ if $i \leq j$ for $\alpha = x, y, z$, one could in principle write out the full expression for the remainder using only regular '+' and '−' operations. This expression would include terms of the type $R^1_x R^1_y R^1_z$, $R^2_x R^2_y$ and R^3_x, as well as terms which lead to higher order errors. Thus, assuming for simplicity that the step sizes are equal in all directions (i.e $h_{j,x} = h_{j,y} = h_{j,z} = h_j$ for $j = 1, 2, 3$), and using the fact that $R^j_\alpha f = O(h_j^2)$, we see that

$$\mathcal{R}^{\mathcal{M}}_{xyz} f = O(h_1^6 + h_2^4 + h_3^2),$$

from which we deduce that it is optimal if $h_1^6 = h_2^4 = h_3^2$, i.e. $h_3 = h_2^2 = h_1^3$. In that case, we will get an error of order h_1^6, while the number of points used is

$$N_B = 3N_1(2N_2N_3 - N_2^2) - 3N_1^2 N_3 + N_1^3 = O(N_1^6),$$

while in order to have the same accuracy on a regular grid, we need $N_S = N_3^3 = O(N_1^9)$ grid nodes, i.e. a difference of 3 in the exponent.

This seems very good indeed, but there is a problem with this approach, which is seen if we expand (2.7) into an expression containing only regular '+' and '−' operators. Then we get 13 distinct terms, which is far too many for our purposes, since it requires a lot of overhead calculations as will be seen in the next section. It may also be a problem to actually create and use such Boolean grids, since even a fairly large h_1 will create a very small h_3 and make the number of points needed very large. Therefore, we take a middle way, reducing the number of terms but paying for this by getting a reduction in accuracy, although we still maintain a clear advantage over the regular grid.

So, using the same definition of P^j_α as above, we create the projector

$$\mathcal{P}_{xyz} = P^2_x P^1_y P^1_z \oplus P^2_y P^1_z P^1_x \oplus P^2_z P^1_x P^1_y = P^{211}_{xyz} \oplus P^{121}_{xyz} \oplus P^{112}_{xyz}, \tag{2.8}$$

which corresponds to first projecting the function onto lines $\{x = x_i,\ y = y_j\}$, which are spaced with the step size h_1, then discretising these lines with step size h_2 and finally take the Boolean sum of the three directions. This gives a grid like the one on the right of figure 2.2. Expanding $\mathcal{P}_{xyz}$ using the algebraic rules, we see that

$$\mathcal{P}_{xyz} = P^{211}_{xyz} + P^{121}_{xyz} + P^{112}_{xyz} - 2P^{111}_{xyz}, \tag{2.9}$$

giving us just four terms. The remainder becomes

$$\mathcal{R}_{xyz} = R^{211}_{xyz} R^{121}_{xyz} R^{112}_{xyz},$$

with

$$R^{211}_{xyz} = R^2_x \oplus R^1_y \oplus R^1_z = R^2_x + R^1_y + R^1_z - R^2_x R^1_y - R^2_x R^1_z - R^1_y R^1_z + R^2_x R^1_y R^1_z,$$

which means that $\mathcal{R}_{xyz}$ includes terms of the type R_x^2 and $R_x^1 R_y^1$ as well as higher order terms. Therefore,

$$\mathcal{R}_{xyz} f = O(h_1^4 + h_2^2),$$

and we see that a choice of $h_2 = h_1^2$ gives the optimal $O(h_1^4)$ error. The number of points used is now

$$N_B = 3N_1^2 N_2 - 2N_1^3 = O(N_1^4),$$

while for a regular grid $N_S = N_2^3 = O(N_1^6)$ for the same accuracy.

2.2.3. Application to finite difference solvers

It is clear from the previous section that if we can create some approximation method with second order errors in each direction, we can use the Boolean grids defined there to improve accuracy, while using few grid nodes (as compared to standard grids). Specifically, we want to use these grids for finite difference approximations of derivatives, in particular the Laplacian in two or three dimensions.

Starting in two dimensions, we therefore define the standard finite difference approximation of the second derivative in x by

$$D_{x,h}^2 u(x_i, y) = \frac{u(x_{i+1}, y) - 2u(x_i, y) + u(x_{i-1}, y)}{h^2},$$

which is second order accurate, that is, the error in the approximation is $O(h^2)$. $D_{y,h}^2$ is defined similarly. The discrete Laplacian is then given by

$$\Delta_{h_x,h_y} u(x_i, y_j) = D_{x,h_x}^2 u(x_i, y_j) + D_{y,h_y}^2 u(x_i, y_j).$$

Assuming for simplicity that the step sizes are equal in the two directions ($h_{1,x} = h_{1,y} = h_1$ and $h_{2,x} = h_{2,y} = h_2$), we may then investigate the Boolean approximation of the Laplacian, which according to (2.6) should be defined as

$$\Delta_{h_1,h_2}^B u(x, y) = \Delta_{h_2,h_1} u(x, y) + \Delta_{h_1,h_2} u(x, y) - \Delta_{h_1,h_1} u(x, y).$$

The accuracy of this approximation is then $O(h_1^4 + h_2^2)$, i.e. $O(h_1^4)$ if we choose $h_2 = h_1^2$. It is again easy to see why the errors cancel out. The two first approximations have one fine direction and one coarse. The derivatives in the coarse direction are cancelled by subtraction of the last term, which is coarse in both directions. It should be emphasized that the $O(h_1^4)$-accuracy holds not only on the Boolean grid in figure 2.1, but at any point inside the domain, as long as we use the Boolean combination of the three components. That is, having calculated the three contributions $\Delta_{h_2,h_1} u$, $\Delta_{h_1,h_2} u$ and $\Delta_{h_1,h_1} u$ on the three grids, we may linearly interpolate on each of these and combine these interpolations by the Boolean scheme to get the value at any point. These interpolations introduce new error terms, and the error is in general larger, but the order of the error is the same as before.

In three dimensions, things are very much the same. We use the intermediate scheme defined by (2.8) on the grid on the right of figure 2.2, which consists of four grids. Three grids are fine in one direction and coarse in the other two, while the last grid is coarse in all directions. Again, using the appropriate Boolean scheme given by (2.9), we set

$$\begin{aligned}\Delta_{h_1,h_2}^B u(x, y, z) = {} & \Delta_{h_2,h_1,h_1} u(x, y, z) + \Delta_{h_1,h_2,h_1} u(x, y, z) \\ & + \Delta_{h_1,h_1,h_2} u(x, y, z) - 2\Delta_{h_1,h_1,h_1} u(x, y, z),\end{aligned}$$

to get an $O(h_1^4 + h_2^2)$-approximation.

Solving for example the Poisson equation $\Delta u = f$ using finite differences on Boolean grids is now straightforward. One simply solves the problem on each of the component grids and then combines the solutions using the Boolean scheme. This is shown in example 2 in section 2.4.2.

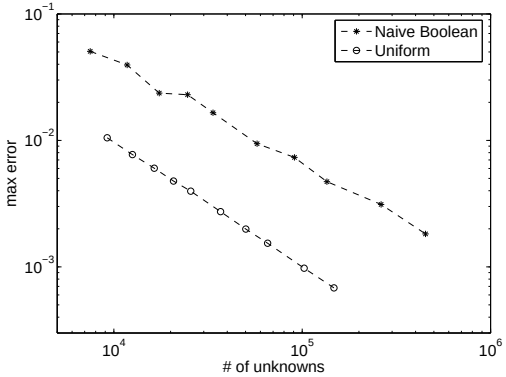

Figure 2.3: Comparison of different types of Boolean immersed interface methods applied to the Poisson equation inside a circle in $\mathbb{R}^2$, where the solution is a sixth degree polynomial. The errors for the naive scheme, where three complete solutions by IIM are combined by the Boolean scheme, are denoted by '$*$'. The errors for IIM on uniform grids are denoted by '$\circ$'. It is seen clearly that in the naive method, the errors are not eliminated in the way we want.

2.3. Applying IIM to Boolean grids

Having successfully applied finite differences on the Boolean grids, we now try to implement the Immersed Interface Method as described in section 1 on these grids in order to cope with domains that are not rectangular.

From the previous section, it might be suggested that one could use the Immersed Interface Method straight away on the component grids and then combine them using the Boolean scheme to get improved results. This does not work, however, since when we introduce jumps in the finite differences, the errors no longer behave in the correct way to be eliminated by the Boolean scheme. This is seen in figure 2.3.

Therefore, we need a slightly more elaborate method. The reason for not getting the desired elimination of errors is clearly the estimation of the jumps at the boundaries, since the results in section 2.4.2 show that finite differences work fine on Boolean grids. So, instead of approximating the jumps on each grid individually, we use the Boolean interpolation to get the jumps from the solution. That is, we must create a matrix D_B^T implementing this Boolean interpolation so that

$$C = D_B^T U + F_2,$$

with $U = [U_{11}^T \; U_{21}^T \; U_{12}^T]^T$ containing the solution on the three grids (supposing we are solving a two-dimensional problem). Here U_{ij} is the solution on the grid created by the projection P_{xy}^{ij}. As before, if we intend to solve the Poisson equation, we also have the equation

$$\Delta_h U + \Psi C = F_1.$$

Here Δ_h works on the three grids separately and Ψ is divided into three blocks corresponding to the grids. As in the uniform case, we also solve for C in the equation

$$(I + D_B^T \Delta_h^{-1} \Psi) C = D_B^T \Delta_h^{-1} F_1 + F_2$$

and then finally compute the solution U as

$$U^* = B\Delta_h^{-1}(-\Psi C + F_1).$$

Here B is a Boolean interpolation matrix, combining the three solutions U_{11}, U_{21} and U_{12} into the final solution U^*, which could be defined on any grid.

The procedure outlined here is our Boolean IIM, resulting in the errors shown in section 2.4.3. The only difference to the uniform IIM described in section 1 is the jump approximation matrix D_B^T and the interpolation matrix B. The interpolation matrix B uses the Boolean schemes (2.6) and (2.9) in two and three dimensions respectively, together with linear interpolation between grid nodes, in order to get the solution U^* at the desired points. The matrix D_B^T is slightly more complicated and we will now discuss how to form it.

2.3.1. Boolean approximation of jumps

Just as in the IIM on uniform grids, we need to approximate the jumps of the solution and its derivatives on the boundary, using the values of the solution on the grid nodes. The boundary conditions are implemented exactly as for the uniform grids (see section 1.4), so what we need to do here is to create equivalents to matrices $\mathcal{Q}_j$, $\mathcal{P}_j$ and $\mathcal{R}_j$ in (1.12)

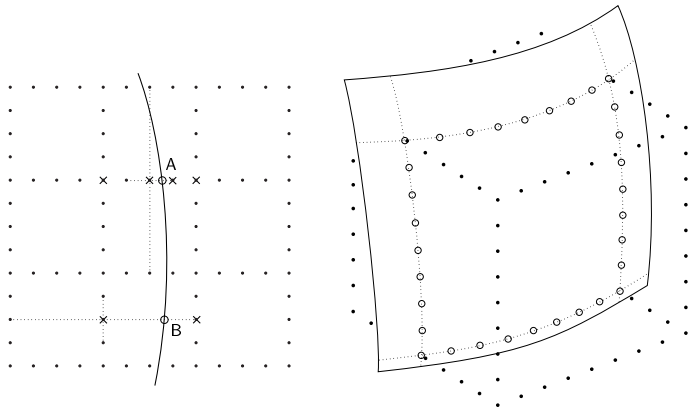

Figure 2.4: Some IIPs on the Boolean grids. The IIPs are marked with circles and the anchor points with crosses. In two dimensions, the IIPs are evenly distributed with spacing h_2. At 'A', the IIP interferes with finite differences both on the fine grid (with step size h_2) and on the coarse grid (with step size h_1). Two of the affected finite difference stencils are indicated by dotted lines. In three dimensions the IIPs are distributed along the edge of the grid cube, giving rise to a two-dimensional Boolean grid pattern.

in order to estimate the values of u and its derivatives at the interface intersection points (IIPs) from both sides of the interface.

So, first of all, we need to find all the IIPs, that is, all the points where the interface intersects the grid lines and thus interferes with the finite differences. Suppose that we want to calculate the Laplacian in two dimensions. Then the IIPs are all the points where the interface intersects the fine grid, as is shown in figure 2.4. That is, we will have as many IIPs as in the case of a uniform grid. In three dimensions, however, the IIPs are distributed on the surface in a two-dimensional Boolean grid, meaning that we have much fewer IIPs using a Boolean grid than a uniform one. This also implies that the vector C will be smaller and thus the entire system of equations will be smaller.

Having found the IIPs, we assign to each of them two (or more) anchor nodes, on both sides of the surface. These are the nodes where the finite difference is affected by the jump in the solution at the IIP. The particular difference taken may be either of size h_1 or size h_2, as is shown in figure 2.4. From (1.4), we see that this gives rise to corrections to the regular second order difference $D_h^2 u$ given by

$$\frac{d^2u}{dx^2}(x_i) = D_h^2 u(x_i) + \frac{1}{h^2}\sum_{m=0}^{3}\frac{(x_i-\alpha)^m}{m!}[u^{(m)}]_\alpha + O(h^2). \tag{2.10}$$

Here, α is the location of the IIP, so that $x_i - \alpha \le h$, and h may be either h_1 or h_2. The question now is how well we need to approximate the jumps and how many derivatives we need. We saw earlier that in the uniform case it is enough to get an approximation of order $O(h)$ on the boundary in order to get overall $O(h^2)$-convergence, and therefore that we may ignore the jumps in the third derivative. In the Boolean case, however, it is not quite clear what we need to do. We want overall $O(h_2^2)$-convergence, but if we ignore the jumps in third derivative in (2.10), we will at some points get errors of order $O(h_1)$, which seems to be too large. Fortunately, our numerical results show that it is in fact sufficient to use jumps in u, u' and u'' and use Boolean interpolation to approximate these to order $O(h_2^3)$, $O(h_2^2)$ and $O(h_2)$, respectively. From (2.10), we then see that we will get truncation errors of order $O(h_1)$ at some nodes (where $h = h_1$) and $O(h_2)$ at others, while the overall error in the solution will be of order $O(h_2^2)$. The explicit reason that this approach works is not known.

How this is done is most easily seen from figure 2.5. Suppose we want to approximate the function values and the two first derivatives in the x-direction at the IIP α_j with coordinates $(x_{\alpha_j}, y_{\alpha_j})$. We use regular Lagrange interpolation on each of the grids, that is, given grid nodes with x-coordinates $x_1, \ldots, x_L$, we set

$$\tilde{u}(x) = \sum_{l=1}^{L} u(x_l)\phi_l(x),$$

with the basis functions

$$\phi_l(x) = \prod_{\substack{k=1\\k\neq l}}^{L}\frac{x-x_k}{x_l-x_k}.$$

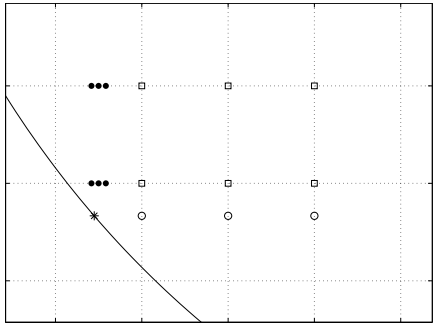

Figure 2.5: Interpolation stencil for jumps at the IIP marked by '$*$'. The nodes in the G_{21} grid are marked by '$\bullet$', in the G_{12} grid by 'o' and in the G_{11} grid by '$\square$'.

To compute derivatives, we just differentiate the basis functions, getting

$$\phi_l'(x) = \sum_{\substack{i=1\\i\neq l}}^{L}\frac{1}{x_l-x_i}\prod_{\substack{k=1\\k\neq l,i}}^{L}\frac{x-x_k}{x_l-x_k},$$

and so on. In this way, we may write for each of the three grids

$$\tilde{u}(x,y) = \sum_{i=1}^{L_x}\sum_{l=1}^{L_y} u(x_i,y_l)\phi_i(x)\phi_l(y).$$

Here, $\tilde{u}$ is of course the projection of u onto the space spanned by the basis functions, so from the theory of section 2.2 we expect that if we combine the three grids according to the Boolean sum, we would eliminate the largest errors. Therefore, we choose grid nodes according to figure 2.5, with (at least) three nodes in the x-direction in order to approximate second derivatives, and (at least) two nodes in the y-direction in order to extrapolate the function to the IIP in that direction.

One could also understand the choice of nodes in the following way. What we really want is a fine (h_2) approximation in the x-direction using the grid G_{21}, which is fine in that direction. But since we must extrapolate in the y-direction, we introduce large (h_1) errors in that direction. These are eliminated by subtracting the approximation on the coarse grid G_{11}, but this in turn adds large (h_1) errors in the x-direction. These are finally eliminated by adding the approximation on the G_{12} grid, which does not introduce new errors in the y-direction.

These principles may be used to approximate any derivative to any order of accuracy, if one uses enough grid nodes for the approximation. For our purpose it is best to use as few grid nodes as possible, since the more nodes we use, the more coupled the resulting system of equations becomes, which makes it harder to solve. Already, the minimal number of 15 nodes shown in figure 2.5 is much more than the six nodes needed on uniform grids. One should also note, however, that if the IIP lies on one of the coarse grid lines (as point A in figure 2.4), it is sufficient to use the three nodes on the fine grid G_{21}, and the Boolean combination is then not needed. But, if one would like to solve the Neumann problem, one would also need y-derivatives, which requires additional nodes in a configuration similar to the one we have studied here.

Anyway, the final scheme may be expressed in matrix form as

$$\begin{bmatrix} u^+(\alpha_j) \\ u^+_x(\alpha_j) \\ u^+_{xx}(\alpha_j) \end{bmatrix} = \mathcal{B} \begin{bmatrix} \mathcal{P}_{11}^{j,+} & 0 & 0 \\ 0 & \mathcal{P}_{21}^{j,+} & 0 \\ 0 & 0 & \mathcal{P}_{12}^{j,+} \end{bmatrix} \mathcal{R}_j^+ U,$$

where $\mathcal{B} = [-I\ I\ I]$ performs the Boolean combination, $\mathcal{R}_j^+$ consists only of 1's and 0's and just reorders and renumbers nodes, while $\mathcal{P}_{ik}^{j,+}$ performs the Lagrange interpolation to evaluate $\tilde{u}(\alpha_j)$ and its derivatives using the grid G_{ik}.

If we apply the same procedure on the other side of the interface (if necessary) to create $\mathcal{P}_{ik}^{j,-}$ and $\mathcal{R}_j^-$, and then create the matrix $\mathcal{L}_j$ exactly as in section 1.4, we may estimate jumps at the boundary as

$$\begin{bmatrix} [u]_j \\ [u_x]_j \\ [u_{xx}]_j \end{bmatrix} = \mathcal{L}_j \mathcal{B} \mathcal{P}_j \mathcal{R}_j U,$$

which should be compared to (1.13). We set $D_{B,j}^T = \mathcal{L}_j \mathcal{B} \mathcal{P}_j \mathcal{R}_j$ and stack the contributions from all the IIPs to get the matrix D_B^T and the system of equations

$$C = D_B^T U + F_2.$$

Being thus able to approximate the jumps, we may use the method outlined earlier to apply the Immersed Interface Method on Boolean grids. For problems in three dimensions, the same procedure is used. There is no need to use a three-dimensional approximation of the jumps, so we can use the stencil shown in figure 2.5 for three-dimensional problems as well.

2.4. Examples

2.4.1. Example 1 – Boolean approximation

In this example, we illustrate simple Boolean interpolation. A sixth degree polynomial in two dimensions is sampled on a uniform grid with 64×64 nodes and on a Boolean grid with $N_1 = 8$ and $N_2 = 64$. Both of these approximations are then linearly interpolated onto a

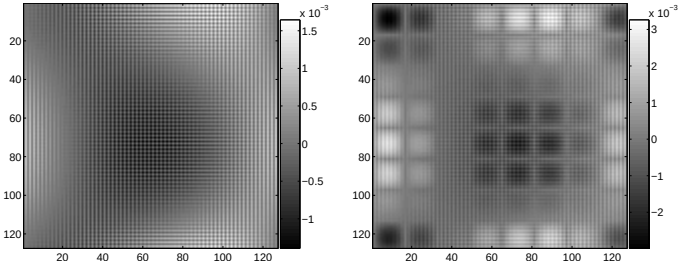

Figure 2.6: Interpolation errors using a uniform grid (left) and a Boolean grid (right). The errors are of about the same size, but the uniform grid has $64 \times 64 = 4096$ nodes, while the Boolean grid uses $2 \times 8^2 \times 8 + 8^2 = 1088$ nodes. Note the patterns of the errors, indicating the grids on which the original sampling was done.

uniform grid with 127×127 nodes. On the Boolean grid, we use the Boolean combination of the three different grids, while in the uniform case a regular linear interpolation is used.

Figure 2.6 shows the resulting errors. In this case, the errors on the Boolean grid are slightly larger, but of approximately the same size as on the uniform grid. However, the number of nodes used in the Boolean grid is $2 \times 8^2 \times 8 + 8^2 = 1088$, while the uniform grid uses $64 \times 64 = 4096$ nodes.

2.4.2. Example 2 – Finite differences on Boolean grids

This example illustrates the use of finite differences on Boolean grids, as explained in section 2.2.3. The Laplacian is applied to the function $f(x) = \sin(2\pi x)\sin(2\pi y)$ on the unit square using finite differences. The exact result is $\Delta f(x) = -8\pi^2 f(x)$. Figure 2.7 shows the result on a Boolean grid with $N_1 = 12$ and $N_2 = 12 \times 8 = 96$. The values on the Boolean grid are interpolated onto a uniform grid with 96×96 nodes and the errors are calculated on this grid. Note that the structure of the Boolean grid is reflected in the error image on the right of figure 2.7. This indicates that the errors are smaller on the Boolean grid than in between grid lines, so that interpolation errors are greater than errors resulting from finite differences on the grid itself.

Figure 2.8 shows the asymptotics of the same approximation compared to approximations on a uniform grid. On the left we see a log-log plot of the errors for the discrete Laplacian applied to $f(x)$ as above, while on the right we see errors for the inverse Laplacian applied to $-8\pi^2 f(x)$ (so that the exact solution is $f(x)$), using zero boundary conditions at the left edge ($x = 0$) and periodic boundary conditions at the other edges. The Boolean grids used have values for N_1 ranging between 24 and 48 and with $N_2 = N_1^2/2$. In the left

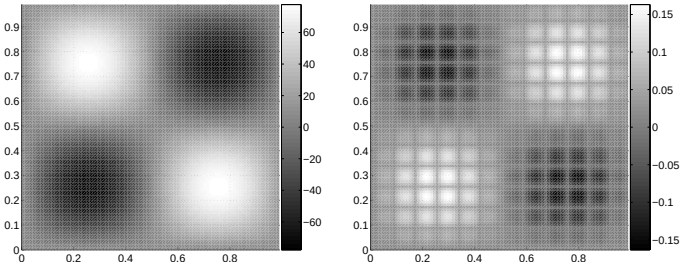

Figure 2.7: Finite differences on a Boolean grid with $N_1 = 12$ and $N_2 = 12 \times 8$. The discrete Laplacian was applied to the function $f(x) = \sin(2\pi x)\sin(2\pi y)$ on a Boolean grid and then interpolated to a full grid. The result is shown on the left with errors on the right. The error image shows that the errors are smaller on the Boolean grid than between grid lines.

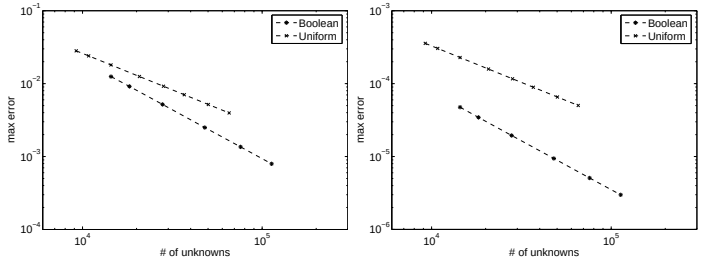

Figure 2.8: Asymptotics for finite difference approximations on two-dimensional uniform and Boolean grids for the forward Laplacian on the left and the inverse Laplacian on the right. The graphs are log-log-plots of errors as function of the number of grid nodes. In both graphs, the slopes of the lines are -1.00 for the uniform grids and -1.34 for the Boolean grids. The Boolean grids use $N_2 = N_1^2/2$ and values of N_1 range from 24 to 48 nodes.

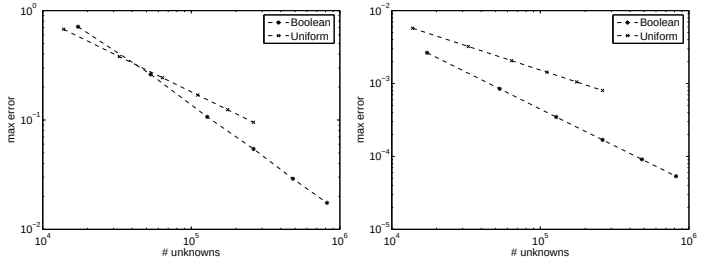

Figure 2.9: Error asymptotics for the discrete Laplacian (left) and its inverse (right) applied using finite differences on uniform and Boolean grids in 3D. The plots are log-log-plots of the number of grid nodes versus the maximum error and the slopes of the lines are near the theoretical values of $-2/3$ for the uniform grids and -1 for the Boolean grids.

figure, the slopes of the lines are -1.000 for the uniform grid and -1.339 for the Boolean grid, while in the right figure, the slopes are -1.000 and -1.344, respectively. This agrees well with the theoretical values, which state that on a uniform grid, the approximation should be of order $O(h^2) = O(N^{-2})$, while the number of unknowns is $O(N^2)$, giving the slope $-2/2 = -1$ in a log-log plot. On the Boolean grid, the number of unknowns is $2N_1N_2 + N_1^2 = O(N_1^3)$ and the accuracy is $O(h_2^2) = O(N_2^{-2}) = O(N_1^{-4})$, since N_2 is proportional to N_1^2. This should give a slope of $-4/3$ in the log-log plot, which agrees very well with the numerical results.

In the last figure, figure 2.9, the asymptotics for the same problem in the unit cube in 3D is shown. Here $f(x) = \sin(2\pi x)\sin(2\pi y)\sin(2\pi z)$ with $\Delta f(x) = -12\pi^2 f(x)$. The figure shows results for the Laplacian on the left and for the inverse Laplacian on the right. The Boolean grid is the intermediate grid with four components. N_1 ranges from 12 to 32 and $N_2 = N_1^2/4$. The slopes of the lines for the uniform grid are both -0.667, which agrees perfectly with the theoretical value derived from $O(h^2) = O(N^{-2})$ accuracy and $O(N^3)$ unknowns. For the Boolean grid, the slope in the figure is -0.965 for the Laplacian and -1.011 for its inverse, which agrees with the theoretical value of -1, arising from the fact that the accuracy is $O(h_2^2) = O(N_1^4)$ and the number of unknowns is $3N_1^2N_2 + N_1^3 = O(N_1^4)$.

The conclusion is that the theory works for standard finite differences and that by using Boolean grids, the number of nodes may be reduced while still retaining the same accuracy. It should be noted that the choices of N_2 here are not necessarily optimal. N_2 is proportional to N_1^2, which it must be, in order to achieve the correct asymptotics. But given a specific N_1, it is not clear beforehand which is the optimal N_2. There is a limit where there is no use increasing N_2 further, because the dominating error terms depend only on N_1. This limit is problem dependent, since the sizes of the error terms depend on

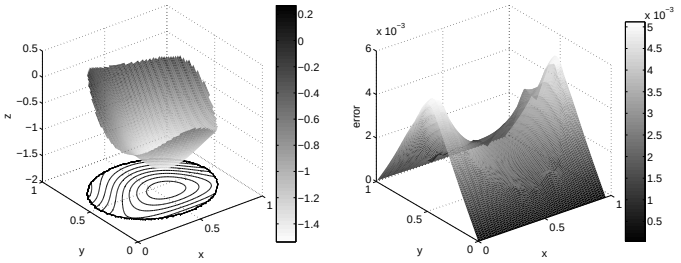

Figure 2.10: IIM on Boolean grid. On the left we see the numerical solution $u(x, y)$ to the Poisson problem in an ellipse computed on a Boolean grid with $N_1 = 24$ and $N_2 = 24 \times 6 = 144$. Level curves are drawn at the bottom, showing clearly the ellipse E. The solution is zero outside the domain. On the right, we see the errors compared to the exact solution on the G_{12} grid with small steps in the y-direction.

the magnitude of the derivatives of the solution f. Choosing optimal values of N_2 for all N_1 will not change the slope of the lines, only the position, that is, it will not change the asymptotics even if it changes the error values.

2.4.3. Example 3 – IIM on Boolean grids

Our final example shows the full immersed interface method on Boolean grids. We solve the Poisson equation with Dirichlet boundary conditions inside an ellipse E with center at $(0.5, 0.5)$ and half-axes 0.44 and 0.38, i.e.

$$\begin{aligned} \Delta u(x, y) &= f(x, y), \qquad (x, y) \in E, \\ u(x, y) &= g(x, y), \qquad (x, y) \in \partial E, \end{aligned}$$

with $f(x, y)$ and $g(x, y)$ chosen so that the solution $u(x, y)$ is a given sixth degree polynomial (the same as in example 1). The solution and the distribution of errors on one of the component grids (G_{12}) are shown in figure 2.10. The results for different grid sizes are listed in table 1. $\|E_N\|_\infty$ denotes the maximum error compared to the exact solution on the Boolean grid, while $\|T_N\|_\infty$ denotes the maximum truncation error, i.e. $T_N = \Delta_h u0 + \Psi C_0 - F_1$, where u_0 and C_0 are the node values and jumps for the exact solution, respectively. The table also lists approximate computational times.

The asymptotics of the errors are shown in figure 2.11. There, asymptotics for a corresponding three-dimensional problem is also shown, that is the Poisson equation inside an ellipsoid with Dirichlet boundary conditions, whose solution is a sixth degree polynomial. The corresponding data is shown in table 2. The numerical asymptotics for the errors

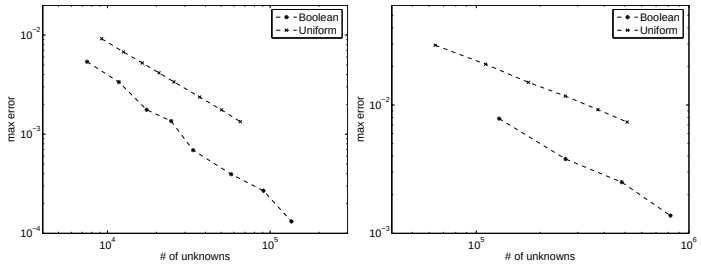

Figure 2.11: Asymptotics of errors for IIM on Boolean grids. Errors for the solution to the Poisson equation inside an ellipse in 2D on the left and inside an ellipsoid in 3D on the right. The results on Boolean grids are compared to results for regular IIM on uniform grids. On the Boolean grids, values of $N_2 = N_1^2/4$ are used throughout. The slopes of the lines are -1.27 for Boolean grids and -0.97 for uniform grids in 2D, while for the 3D problem they are -0.91 and -0.66, respectively. This should be compared to the theoretical values, which for $O(h^2)$-convergence are $-4/3$, -1, -1 and $-2/3$ in the given order (see example 2).

$\|E_N\|$ agree nicely with the theoretical values for $O(h_2^2)$-behavior given in the previous example. As expected, the truncation errors exhibit $O(h_1)$-behavior in both two and three dimensions, but even though these errors are large, the final errors in the solution are small. The computational times listed indicate that the time needed to solve the problem grows slightly faster than the number of unknowns. This is because the number of iterations needed in the BiCGStab algorithm to solve the linear system of equations is not constant, but tend to increase slightly with the number of unknowns. (e.g. from 17 to 21 in the 3D problem).

We conclude that we are able to achieve $O(h_2^2)$ error asymptotics for both two- and three-dimensional problems on the Boolean grids. We also see from the error plots that the number of grid nodes needed to get a specific error is much lower for the Boolean grids than for the uniform ones.

N_1	n_2	N_{tot}	time (s)	$\|E_N\|_\infty$	$\|T_N\|_\infty$
24	6	7488	< 0.1	5.37e-3	2.51
28	7	11760	0.1	3.35e-3	1.49
32	8	17408	0.2	1.76e-3	1.37
36	9	24624	0.3	1.35e-3	1.19
40	10	33600	0.5	6.90e-4	1.34
48	12	57600	0.9	3.94e-4	1.23
56	14	90944	1.4	2.68e-4	1.02
64	16	135168	2.5	1.31e-4	0.86

Table 1: Results for Boolean IIM in two dimensions. $n_2 = N_2/N_1$ and $N_{tot} = N_1^2(2n_2+1)$. See text for details.

N_1	n_2	N_{tot}	time (s)	$\|E_N\|_\infty$	$\|T_N\|_\infty$
20	5	1.28e5	3.1	7.74e-3	2.78
24	6	2.63e5	7.1	3.76e-3	2.49
28	7	4.83e5	13.2	2.50e-3	2.19
32	8	8.19e5	26.6	1.37e-3	1.91

Table 2: Results for Boolean IIM in three dimensions. $n_2 = N_2/N_1$ and $N_{tot} = N_1^3(3n_2+1)$. See text for details.

References

[1] FFTW, `http://www.fftw.org`.

[2] IML++ (Iterative Methods Library), `http://math.nist.gov/iml++/`.

[3] H. Bungartz, M. Griebel, and U. Rüde. Extrapolation, combination, and sparse grid techniques for elliptic boundary value problems. *Comput. Methods Appl. Mech. Engrg.*, 116(1-4):243–252, 1994. ICOSAHOM'92 (Montpellier, 1992).

[4] C. K. Chui and H.-C. Lai. Vandermonde determinant and Lagrange interpolation in $\mathbf{R}^s$. In *Nonlinear and convex analysis (Santa Barbara, Calif., 1985)*, volume 107 of *Lecture Notes in Pure and Appl. Math.*, pages 23–35. Dekker, New York, 1987.

[5] S. A. Coons. Surfaces for computer aided design of space forms. Technical report, Project MAC, Dept. of Mech. Engineering, MIT, 1964. Revised to MAC-TR-41, 1967.

[6] F.-J. Delvos and W. Schempp. *Boolean methods in interpolation and approximation*, volume 230 of *Pitman Research Notes in Mathematics Series*. Longman Scientific & Technical, Harlow, 1989.

[7] W. J. Gordon. Distributive lattices and the approximation of multivariate functions. In *Approximations with Special Emphasis on Spline Functions (Proc. Sympos. Univ. of Wisconsin, Madison, Wis., 1969)*, pages 223–277. Academic Press, New York, 1969.

[8] W. J. Gordon. Blending-function methods of bivariate and multivariate interpolation and approximation. *SIAM J. Numer. Anal.*, 8:158–177, 1971.

[9] M. Griebel, M. Schneider, and C. Zenger. A combination technique for the solution of sparse grid problems. In *Iterative methods in linear algebra (Brussels, 1991)*, pages 263–281. North-Holland, Amsterdam, 1992.

[10] C. T. Kelley. *Iterative methods for linear and nonlinear equations*, volume 16 of *Frontiers in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1995. With separately available software.

[11] R. J. LeVeque and Z. L. Li. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM J. Numer. Anal.*, 31(4):1019–1044, 1994.

[12] Z. Li. *The immersed interface method: a numerical approcah to partial differential equations with interfaces*. PhD thesis, Dept. of Applied Mathematics, University of Washington, Seattle, 1994.

[13] C. S. Peskin. Lectures on mathematical aspects of physiology. In *Mathematical aspects of physiology (Proc. Summer Sem., Univ. Utah, Salt Lake City, Utah, 1980)*, volume 19 of *Lectures in Appl. Math.*, pages 1–107. Amer. Math. Soc., Providence, RI, 1981.

[14] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical recipes in C*. Cambridge University Press, Cambridge, second edition, 1992. The art of scientific computing.

[15] A. Wiegmann. *The explicit jump immersed interface method and interface problems for differential equations*. PhD thesis, Department of Mathematics, University of Washington, Seattle, 1998.

[16] A. Wiegmann and K. P. Bube. The explicit-jump immersed interface method: Finite difference methods for PDEs with piecewise smooth solutions. *SIAM J. Numer. Anal.*, 37(3):827–862, 2000.

Approximation of generalized mean curvature flow with right-angle boundary conditions

Tobias Gebäck

Abstract

In this paper, we prove the convergence of an algorithm for computing the evolution of surfaces, which move at each point with a velocity equal to an increasing function of the mean curvature in that point. Furthermore, the entire evolution is assumed to take place inside a convex domain and wherever the surface intersects the domain boundary, it should do so at a right angle. We show that the approximations given by the algorithm converge to the viscosity solution of the corresponding PDE as the time step tends to zero.

The algorithm presented here is a generalization of the algorithm presented by Ishii and Ishii for regular mean curvature evolution with right-angle boundary conditions and the algorithm by Grzibovskis and Heintz for the case when the velocity equals an increasing function of the mean curvature, without boundary conditions. These algorithms are in turn based on the convolution-thresholding scheme devised by Bence, Merriman and Osher.

Contents

1 Introduction **1**
- 1.1 The BMO-algorithm . . . 2
- 1.2 Right-angle boundary conditions . . . 3
- 1.3 Generalized mean curvature motion . . . 3
- 1.4 Outline . . . 4

2 Viscosity solutions **5**
- 2.1 Introduction . . . 5
- 2.2 Theory of viscosity solutions . . . 5
- 2.3 Boundary conditions . . . 7
- 2.4 Singular equations . . . 8
- 2.5 Comparison . . . 8

3 The algorithm **9**

4 Properties of G_h and $\mathcal{G}_h$ **12**

5 The convergence theorem **23**

6 Examples **33**

References **36**

1. Introduction

Consider a hypersurface Γ_0 in $\mathbb{R}^n$. At each point $x \in \Gamma_0$, assign a velocity v in the normal direction, so that Γ_0 moves at each point with velocity $v(x)\hat{n}(x)$, creating a new hypersurface. Continuing the process, a family $\{\Gamma_t\}_{t\geq 0}$ is created, where the hypersurfaces Γ_t evolve according to the normal velocity $v(x,t)$. If we take $v(x,t) = \kappa$, the mean curvature of Γ_t at x, we get the *mean curvature flow*. Mean curvature flows have been studied since the 1970:s, first by parametric methods from differential geometry, although it was soon clear that for $n \geq 3$ these methods ran into problems even for smooth hypersurfaces Γ_0, as the mean curvature flow could develop singularities (so that the curvature is not defined for some x), see figure 1.1. A method to overcome these problems, was introduced by Brakke [5], who used varifold theory to define weak notions of mean curvature flow (see also [8] for a modernization of these results). This method, however, does not give unique solutions.

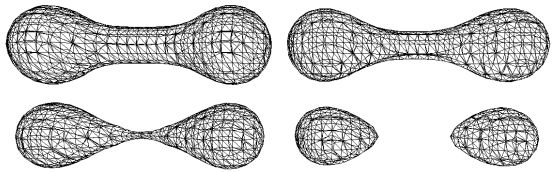

Figure 1.1: The mean curvature evolution of a dumbbell-shaped surface. The surface develops a singularity after a finite time and is split into two. The image was produced by R. Grzibovskis.

Then, following ideas from Osher and Sethian [17], Evans and Spruck [10] developed a new approach to motion by mean curvature in which the hyper-surface Γ_0 is viewed as a level set of a continuous function f, so that

$$\Gamma_0 = \{x \in \mathbb{R}^n \mid f(x) = \lambda\}$$

for some λ. The mean curvature evolution is then studied through the PDE

$$\begin{cases} \dfrac{\partial u}{\partial t} = |\nabla u| \operatorname{div}\left(\dfrac{\nabla u}{|\nabla u|}\right) & \text{in } \mathbb{R}^n \times (0,\infty), \\ u = f & \text{on } \mathbb{R}^n \times \{t = 0\}, \end{cases} \tag{1.1}$$

which ensures that the level sets of u evolve according to their mean curvature, at least as long as $\nabla u \neq 0$. This PDE is nonlinear and degenerate parabolic and has a singularity for $\nabla u = 0$, which makes it rather hard to handle. However, the notion of *viscosity solutions* (see Crandall-Ishii-Lions [7]) provides a well-suited tool to handle such equations and to prove the existence of a unique solution, which was done in [10].

A number of generalizations of this approach have appeared. Existence of a unique solution to more general level set equations has been established by Chen, Giga and Goto [6] and for very general cases, including the case where $v = g(\kappa)$ and g is any non-decreasing, continuous function, by Ishii and Souganidis [16]. The Neumann problem for the mean curvature equation has also been studied. In this case, the whole evolution takes place inside a domain Ω, whose boundary is intersected perpendicularly by the level sets of u. The existence of unique solutions in this case was established by Sato [20] for convex Ω and by Giga and Sato [11] for more general Ω, but less general dependence on the curvature. We will return to this later.

Curvature flows arise naturally in a range of situations, including fast reaction-slow diffusion problems (see [18]) and image processing (see [2]).

In 1992, Bence, Merriman and Osher [4] presented an algorithm to approximate motion by mean curvature using the level set approach. The convergence of this algorithm was proven using different approaches by Evans [9], Barles and Georgelin [3] and Ishii [13]. General thresholding schemes were also studied by Ishii, Pires and Souganidis [15]. Later, two significant generalizations of the algorithm have been developed. In 2002, Ishii and Ishii [14] published an algorithm for mean curvature flow with right-angle boundary conditions, and about at the same time, Grzibovskis and Heintz [12] developed a scheme for motion with normal velocity equal to a (nonlinear) function of the mean curvature (henceforth called generalized mean curvature motion).

This work uses the methods used by Ishii and Ishii [14] to prove convergence of an algorithm for such generalized mean curvature motion with right-angle boundary conditions. In order to understand that algorithm, we will first briefly discuss the previous works.

1.1. The BMO-algorithm

A (slightly generalized) version of the Bence, Merriman and Osher-algorithm (BMO-algorithm) can be described as follows (cf. Ishii [13]). First, fix a radially symmetric convolution kernel, ρ, and define its contraction $\rho^{\sqrt{h}}(x) = h^{-n/2}\rho(x/\sqrt{h})$. Then, given a set $C_0 \subset \mathbb{R}^n$, choose a time-step h and compute the convolution $\widetilde{M}^{C_0}(x,h) = (\rho^{\sqrt{h}} * \chi_{C_0})(x)$. Set

$$C_1 = \{x \in \mathbb{R}^n \mid \widetilde{M}^{C_0}(x,h) \geq \frac{1}{2}\int_{\mathbb{R}^n} \rho^{\sqrt{h}}(y)\, dy\}$$

and continue the process by computing $\widetilde{M}^{C_1}(x,h)$ and defining C_2 and so on. We then end up with a sequence $\{C_k\}_{k\in\mathbb{N}}$ of closed sets in $\mathbb{R}^n$ and we set

$$C_t^h = C_k \quad \text{if } kh \leq t < (k+1)h, \quad t \geq 0.$$

Now, letting $h \to 0$, we obtain a flow of closed subsets in $\mathbb{R}^n$ whose boundaries move with a normal velocity equal to a constant times its mean curvature, where the constant depends only on n and the choice of ρ.

In the original algorithm, $\widetilde{M}^{C_0}(x,t)$ was the solution to the heat equation with initial data χ_{C_0}, which corresponds to the choice of ρ as the Gauss kernel, and which leads to motion by $(n-1)$ times the mean curvature.

1.2. Right-angle boundary conditions

As was already mentioned, the above algorithm was extended by Ishii and Ishii [14] to the case of right-angle boundary conditions. The extension works as follows. Given an open domain $\Omega \subset \mathbb{R}^n$ with C^2-boundary, an initial set C_0, and a convolution kernel ρ, we define

$$M^{C_0}(x,h) = \int_\Omega \rho^{\sqrt{h}}(y-x)\chi_{C_0}(y)\,dy$$

and set

$$C_1 = \{x \in \mathbb{R}^n \mid M^{C_0}(x,h) \geq \frac{1}{2}\int_\Omega \rho^{\sqrt{h}}(y-x)\,dy\},$$

which is the same as before, except that the integrals are taken over Ω instead of $\mathbb{R}^n$. Defining the sequence $\{C_k\}_{k\in\mathbb{N}}$ and C_t^h as above and letting $h \to 0$, we get a flow of sets whose boundary not only moves by a constant times mean curvature but also intersects the boundary of Ω at a right angle, at least in a sense that will be clear later.

1.3. Generalized mean curvature motion

The last extension of the BMO-algorithm we will discuss is the scheme by Grzibovskis and Heintz [12], that lets the boundaries of the sets move with a normal velocity $v = g(\kappa)$, where κ is the mean curvature and $g : \mathbb{R} \to \mathbb{R}$ is an increasing, continuous function.

The algorithm uses two different radially symmetric convolution kernels, ρ_1 and ρ_2 and given a set C we can define

$$\begin{aligned}\widetilde{N}_i^C(x,h) &= \widetilde{M}_i^C(x,h) - \frac{1}{2}\int_{\mathbb{R}^n} \rho_i^{\sqrt{h}}(y-x)\,dy \\ &= \int_{\mathbb{R}^n} \rho_i^{\sqrt{h}}(y-x)\chi_C(y)\,dy - \frac{1}{2}\int_{\mathbb{R}^n} \rho_i^{\sqrt{h}}(y-x)\,dy\end{aligned}$$

for $i = 1, 2$. Now, a crucial part of all the proofs of convergence of these convolution-thresholding algorithms is an expansion of $\widetilde{N}_i^C(x,h)$ in h of the form

$$\widetilde{N}_i^C(x,h) = a_i\sqrt{h}v(x) - b_i\sqrt{h}\kappa(x) + o(h). \tag{1.2}$$

with

$$\begin{aligned}a_i &= \int_{\mathbb{R}^{n-1}} \rho_i(y',0)\,dy', \\ b_i &= \frac{1}{2}\int_{\mathbb{R}^{n-1}} y_1^2\rho_i(y',0)\,dy'.\end{aligned}$$

Clearly, setting $\widetilde{N}_i^C(x,h) = 0$ gives $v = \frac{b_i}{a_i}\kappa + o(\sqrt{h})$, which corresponds to the original BMO-algorithm.

Now, using two convolution kernels, (1.2) gives us two linear equations for v and κ. Solving these, we get

$$\begin{cases} v = \dfrac{1}{\sqrt{h}}\dfrac{b_2\widetilde{N}_1 - b_1\widetilde{N}_2}{d} + o(\sqrt{h}) \\ \kappa = \dfrac{1}{\sqrt{h}}\dfrac{a_2\widetilde{N}_1 - a_1\widetilde{N}_2}{d} + o(\sqrt{h}) \end{cases},$$

where $d = a_1b_2 - a_2b_1$ is the determinant. Thus, since we want to have $v = g(\kappa)$, we define

$$F(N_1,N_2) = v - g(\kappa) = \frac{1}{\sqrt{h}}\frac{b_2N_1 - b_1N_2}{d} - g\left(\frac{1}{\sqrt{h}}\frac{a_2N_1 - a_1N_2}{d}\right) \tag{1.3}$$

and a thresholding scheme

$$C_{k+1} = \{x \in \mathbb{R}^n \mid F(\widetilde{N}_1^{C_k}(x,h), \widetilde{N}_2^{C_k}(x,h)) \geq 0\},$$

for $k \in \mathbb{N}$.

In order for this scheme to converge to the actual generalized mean curvature motion, it turns out that F must satisfy the condition $\partial F/\partial N_i > 0$, $i = 1, 2$, which leads to the restrictions

$$\begin{aligned} &d = a_1b_2 - a_2b_1 > 0, \\ &0 < \frac{b_1}{a_1} < g' < \frac{b_2}{a_2}, \end{aligned} \tag{1.4}$$

on g and ρ_i, saying both that g must have bounded derivative both from above and below, and that given a function g, the convolution kernels must be chosen with some care to fulfill the inequalities. If one would like to use a function g with unbounded or zero derivative, it is possible to use uniform approximations g_ν to g and still get a scheme that converges as h and ν tend to zero.

1.4. Outline

The structure of the following sections is as follows. First, in section 2 we give an introduction to viscosity solutions and give a background to the definition of solution and comparison results that are used in section 5.

Having established the necessary background, the actual treatment begins in section 3, where the algorithm is presented in more detail and all assumptions stated. In section 4, a few crucial lemmas are proven. Then, finally, the last section sums it all up in the proof that the algorithm converges to the solution of the level set equation as the time-step tends to zero.

2. Viscosity solutions

2.1. Introduction

The theory of viscosity solutions was developed during the 1980's by M.G. Crandall, L.C. Evans, H. Ishii, P.-L. Lions and others while seeking solutions to the Hamilton-Jacobi equations. The name viscosity solutions originates from the method of "vanishing viscosity" which was used to solve first-order equations and which was consistent with the new theory being developed. Now, however, viscosity solutions do not generally have much to do with viscosity. The theory provides very general existence and uniqueness results and allows merely continuous functions to be solutions of fully nonlinear second-order equations. An excellent account of the theory may be found in the *"User's guide to viscosity solutions"* by Crandall, Ishii and Lions [7]. Here, we give a short introduction to the theory and introduce some concepts which will be used later on.

2.2. Theory of viscosity solutions

The theory of viscosity solutions applies to equations of the form

$$F(x, u, \nabla u, D^2u) = 0, \tag{2.1}$$

where $x \in \mathbb{R}^n$, $u = u(x)$ is a real-valued function, $\nabla u \in \mathbb{R}^n$ its gradient and $D^2u \in \mathcal{S}(n)$ the matrix of second derivatives of u. $\mathcal{S}(n)$ is the set of real, symmetric $n \times n$ matrices, which is partially ordered by the relation $\leq$, where $Y \leq X$ means $\xi^t Y \xi \leq \xi^t X \xi$ for all $\xi \in \mathbb{R}^n$. We also equip $\mathcal{S}(n)$ with the norm $\|X\| = \max\{|X\xi| \mid \xi \in \mathbb{R}^n, |\xi| = 1\}$. Finally, $F : \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^n \times \mathcal{S}(n) \to \mathbb{R}$ is a function, which can take many forms.

For the theory to apply, we require F to satisfy the monotonicity conditions

$$F(x, r, p, X) \leq F(x, s, p, X) \text{ if } r \leq s \tag{2.2}$$

and

$$F(x, r, p, X) \leq F(x, r, p, Y) \text{ if } Y \leq X, \tag{2.3}$$

where $r, s \in \mathbb{R}$, $x, p \in \mathbb{R}^n$ and $X, Y \in \mathcal{S}(n)$. If (2.3) holds, F is said to be *degenerate elliptic* and if (2.2) also holds, F is *proper*.

Now suppose that F is proper and that $u \in C^2(\mathbb{R}^n)$ is a subsolution to $F = 0$, i.e. solves

$$F(x, u(x), \nabla u(x), D^2u(x)) \leq 0$$

for all $x \in \mathbb{R}^n$. Choose a test function φ that is also C^2, and suppose that $u - \varphi$ has a local maximum at $\hat{x}$. Then we have $\nabla(u-\varphi)(\hat{x}) = 0$ and $D^2(u-\varphi)(\hat{x}) \leq 0$, i.e. $\nabla u(\hat{x}) = \nabla\varphi(\hat{x})$ and $D^2u(\hat{x}) \leq D^2\varphi(\hat{x})$, and by (2.3),

$$F(\hat{x}, u(\hat{x}), \nabla\varphi(\hat{x}), D^2\varphi(\hat{x})) \leq F(\hat{x}, u(\hat{x}), \nabla u(\hat{x}), D^2u(\hat{x})) \leq 0.$$

We have thus replaced the derivatives of u with derivatives of test functions φ, which we choose to be well-behaved (i.e. at least C^2). One could now try to define an arbitrary function u to be a weak or generalized *subsolution* of (2.1) if

$$F(\hat{x}, u(\hat{x}), \nabla\varphi(\hat{x}), D^2\varphi(\hat{x})) \leq 0$$

whenever φ is C^2 and $u - \varphi$ has a local maximum at $\hat{x}$. However, a slightly different definition proves more useful and we therefore note that since $u - \varphi$ has a maximum at $\hat{x}$, $u(x) \leq u(\hat{x}) - \varphi(\hat{x}) + \varphi(x)$ for x near $\hat{x}$, so a Taylor expansion of φ at $\hat{x}$ gives

$$u(x) \leq u(\hat{x}) + \langle p, x - \hat{x}\rangle + \frac{1}{2}\langle X(x-\hat{x}), x-\hat{x}\rangle + o(|x-\hat{x}|^2) \tag{2.4}$$

with $p = \nabla\varphi(\hat{x})$ and $X = D^2\varphi(\hat{x})$. Also, if (2.4) holds for some $(p, X) \in \mathbb{R}^n \times \mathcal{S}(n)$ and $u \in C^2$, then $p = \nabla u(\hat{x})$ and $D^2u(\hat{x}) \leq X$, so that if u solves $F \leq 0$, it follows that $F(\hat{x}, u(\hat{x}), p, X) \leq 0$ whenever (2.4) is true. Letting $\mathcal{O} \subset \mathbb{R}^n$ be locally compact, $u : \mathcal{O} \to \mathbb{R}$ and $\hat{x} \in \mathcal{O}$, we therefore define the second-order "superjet" of u at $\hat{x}$ by

$$J^{2,+}_{\mathcal{O}}u(\hat{x}) = \{(p, X) \mid (2.4) \text{ holds as } x \to \hat{x}, \text{ with } x \in \mathcal{O}\}. \tag{2.5}$$

Reversing the inequality sign in (2.4) gives us the definition of the "subjet" $J^{2,-}_{\mathcal{O}}u(\hat{x})$, or equivalently, $J^{2,-}_{\mathcal{O}}u(\hat{x}) = -J^{2,+}_{\mathcal{O}}(-u)(\hat{x})$. We also state the definitions of the sets of upper and lower semicontinuous functions

$$\begin{aligned} \mathrm{USC}(\mathcal{O}) &= \{u : \mathcal{O} \to [-\infty, \infty) \mid u^{-1}([-\infty, a)) \text{ is open in } \mathcal{O} \text{ for each } a \in \mathbb{R}\} \\ \mathrm{LSC}(\mathcal{O}) &= \{u : \mathcal{O} \to (-\infty, \infty] \mid u^{-1}((a, \infty]) \text{ is open in } \mathcal{O} \text{ for each } a \in \mathbb{R}\} \end{aligned} \tag{2.6}$$

Finally, we are able to define viscosity solutions for the equation (2.1).

Definition 2.1 *Let F be proper and $\mathcal{O} \subset \mathbb{R}^n$. Then $u \in \mathrm{USC}(\mathcal{O})$ is a* subsolution *of $F = 0$ on $\mathcal{O}$ if*

$$F(x, u(x), p, X) \leq 0 \text{ for all } x \in \mathcal{O} \text{ and } (p, X) \in J^{2,+}_{\mathcal{O}}u(x). \tag{2.7}$$

Similarly, $u \in \mathrm{LSC}(\mathcal{O})$ is a supersolution *of $F = 0$ on $\mathcal{O}$ if*

$$F(x, u(x), p, X) \geq 0 \text{ for all } x \in \mathcal{O} \text{ and } (p, X) \in J^{2,-}_{\mathcal{O}}u(x). \tag{2.8}$$

Finally, u is a (viscosity) solution *of $F = 0$ in $\mathcal{O}$ if it is both a subsolution and a supersolution.*

We note that since viscosity solutions are both upper and lower semicontinuous, they are continuous. Also, in view of the discussion above, it may be noted that

$$J^{2,+}_{\mathcal{O}}u(\hat{x}) = \{(\nabla\varphi(\hat{x}), D^2\varphi(\hat{x})) \mid \varphi \in C^2 \text{ and } u - \varphi \text{ has a local maximum at } \hat{x}\},$$

$$J^{2,-}_{\mathcal{O}}u(\hat{x}) = \{(\nabla\varphi(\hat{x}), D^2\varphi(\hat{x})) \mid \varphi \in C^2 \text{ and } u - \varphi \text{ has a local minimum at } \hat{x}\},$$

which may be used to facilitate the use of definition 2.1. We also note that the semijets only depend on the set $\mathcal{O}$ if $\hat{x} \in \partial\mathcal{O}$, so if that is not the case, we may drop the subscript.

We also need to define the closures of the semijets for $x \in \mathcal{O}$ as

$$\overline{J}^{2,\pm}_{\mathcal{O}} u(x) = \{(p, X) \in \mathbb{R}^n \times \mathcal{S}(n) \mid \exists (x_k, p_k, X_k) \in \mathcal{O} \times \mathbb{R}^n \times \mathcal{S}(n) : (p_k, X_k) \in J^{2,\pm}_{\mathcal{O}} u(x_k) \text{ and } (x_k, u(x_k), p_k, X_k) \to (x, u(x), p, X)\}$$

and note that if u is a subsolution of $F = 0$ in $\mathcal{O}$, then $F(x, u(x), p, X) \leq 0$ for $x \in \mathcal{O}$ and $(p, X) \in J^{2,+}_{\mathcal{O}} u(x)$. If F is lower semicontinuous, this remains true if $(p, X) \in \overline{J}^{2,+}_{\mathcal{O}} u(x)$. Similar remarks are true for supersolutions and solutions.

2.3. Boundary conditions

Viscosity solutions also allow precise formulations of boundary conditions. Consider the boundary value problem

$$\begin{cases} F(x, u(x), \nabla u(x), D^2 u(x)) = 0, & x \in \Omega \\ B(x, u(x), \nabla u(x)) = 0, & x \in \partial\Omega \end{cases} \tag{2.9}$$

in an open set $\Omega \subset \mathbb{R}^n$, where F and B are both proper functions. The correct definition of a viscosity solution of (2.9) is then

Definition 2.2 *A function* $u \in \mathrm{USC}(\overline{\Omega})$ *is a* subsolution *of (2.9) if*

$$\begin{cases} F(x, u(x), p, X) \leq 0 & x \in \Omega, \ (p, X) \in \overline{J}^{2,+}_{\Omega} u(x), \\ B(x, u(x), p) \wedge F(x, u(x), p, X) \leq 0 & x \in \partial\Omega, \ (p, X) \in \overline{J}^{2,+}_{\Omega} u(x). \end{cases} \tag{2.10}$$

$u \in \mathrm{LSC}(\overline{\Omega})$ *is a* supersolution *of (2.9) if*

$$\begin{cases} F(x, u(x), p, X) \geq 0 & x \in \Omega, \ (p, X) \in \overline{J}^{2,-}_{\Omega} u(x), \\ B(x, u(x), p) \vee F(x, u(x), p, X) \geq 0 & x \in \partial\Omega, \ (p, X) \in \overline{J}^{2,-}_{\Omega} u(x). \end{cases} \tag{2.11}$$

Finally, u *is a* solution *if it is both a subsolution and a supersolution.*

Here, $a \vee b = \max\{a, b\}$ and $a \wedge b = \min\{a, b\}$, so what the definition basically means is that on the boundary, either the boundary condition or the equation should hold. That we can not expect the boundary conditions to hold in a stronger sense is demonstrated by an example in [7].

2.4. Singular equations

Since the equation we are interested in has a singularity for $\nabla u = 0$, we need to introduce a third definition of viscosity solutions.

To start with, given a function $u : \Omega \to \mathbb{R}$, with $\Omega \subset \mathbb{R}^n$, we introduce the upper and lower semicontinuous relaxations

$$u^*(x) = \lim_{\varepsilon \to 0} \sup\{u(y) \mid y \in \Omega, |x - y| < \varepsilon\}, \tag{2.12}$$

$$u_*(x) = \lim_{\varepsilon \to 0} \inf\{u(y) \mid y \in \Omega, |x - y| < \varepsilon\}, \tag{2.13}$$

which are defined on $\overline{\Omega}$ and take values in $\mathbb{R} \cup \{\infty\}$ and $\mathbb{R} \cup \{-\infty\}$ respectively.

Also, supposing our function F is only defined on a dense subset W of $L(\Omega) = \Omega \times \mathbb{R} \times \mathbb{R}^n \times \mathcal{S}(n)$, we may similarly define the relaxations F^* and F_* on $\overline{W} = L(\Omega)$ as

$$F^*(x, u, p, X) = \lim_{\varepsilon \to 0} \sup\{F(y, v, q, Y) \mid (y, v, q, Y) \in W, \ \|(x, u, p, X) - (y, v, q, Y)\| < \varepsilon\},$$

$$F_*(x, u, p, X) = \lim_{\varepsilon \to 0} \inf\{F(y, v, q, Y) \mid (y, v, q, Y) \in W, \ \|(x, u, p, X) - (y, v, q, Y)\| < \varepsilon\},$$

where $\|(x, u, p, X)\|$ is just the sum of the norms for each component. Then, we can make the following definition of viscosity solutions:

Definition 2.3 *A function* $u : \Omega \to \mathbb{R}$ *is a* subsolution *of (2.1) if* $u^* < \infty$ *in* Ω *and*

$$F_*(x, u^*(x), p, X) \leq 0 \text{ for all } x \in \Omega, (p, X) \in \overline{J}^{2,+}_{\Omega} u^*(x),$$

a supersolution *if* $u_* > -\infty$ *in* Ω *and*

$$F^*(x, u_*(x), p, X) \geq 0 \text{ for all } x \in \Omega, (p, X) \in \overline{J}^{2,-}_{\Omega} u_*(x),$$

and a solution *if it is both sub- and supersolution.*

This definition is adopted from [6], where it is used for proving existence of solutions of curvature flow equations and similar.

2.5. Comparison

We finally make some comments on the method generally used for proving existence of viscosity solutions to suitable equations. Such proofs in general consist of three steps. The first is to establish a comparison result, i.e that if u is a subsolution and v is a supersolution, then $u \leq v$. From this result it follows immediately that if there is a solution, it must be unique. It also follows that if u is a solution by the definition for singular equations given above, then u is continuous. Furthermore, the comparison result proves helpful to us in our proof of convergence in section 5.

The second step of a proof of existence is to construct a subsolution and a supersolution. The third step is to invoke Perron's method to show that in that case, there exists a solution. See Crandall, Ishii, Lions [7] for details. For applications of the method, see also the proofs of existence of solutions to level set equations for mean curvature motion, e.g [6], [11], [20].

3. The algorithm

Now, having given the necessary background, we turn to the problem of extending the algorithm for approximating generalized curvature flow to the case of right-angle boundary conditions and prove its convergence as the time step tends to zero. For that purpose, let Ω be an *open*, bounded domain in $\mathbb{R}^n$ with C^2-boundary $\partial\Omega$. Given $u_0 \in C(\overline{\Omega})$, we consider the level set equation

$$\begin{cases} \dfrac{\partial u}{\partial t}(x,t) - |\nabla u(x,t)|\, g(\operatorname{curv}(u(x,t))) = 0, & x \in \Omega,\ t \in (0,T) \\ \dfrac{\partial u}{\partial \vec{n}}(x,t) = 0, & x \in \partial\Omega,\ t \in (0,T) \\ u(x,0) = u_0(x), & x \in \overline{\Omega} \end{cases} \tag{3.1}$$

for $T > 0$, where

$$\operatorname{curv}(u(x)) = \operatorname{div}\left(\frac{\nabla u(x)}{|\nabla u(x)|}\right) = \frac{1}{|\nabla u(x)|}\sum_{i,j=1}^{n}\left(\delta_{ij} - \frac{u_{x_i}(x)u_{x_j}(x)}{|\nabla u(x)|^2}\right)u_{x_ix_j}(x) \tag{3.2}$$

is n times the mean curvature of the level set of u passing through the point x, $\vec{n}$ is the outward unit normal to Ω and $g : \mathbb{R} \to \mathbb{R}$ fulfills the conditions

$$\begin{aligned} &\text{(i)} \quad g \in C(\mathbb{R}),\ g(0) = 0 \\ &\text{(ii)} \quad g(x) = O(x) \text{ as } x \to \pm\infty \\ &\text{(iii)} \quad g \text{ is increasing.} \end{aligned} \tag{3.3}$$

The PDE describes a function whose level sets $\{x \in \mathbb{R}^n \mid u(x) = \lambda\}$ move with normal velocity $g(\operatorname{curv}(u(x)))$ and intersect $\partial\Omega$ at a right angle, at least formally. The equation is degenerate parabolic and has singularities for $\nabla u = 0$, but in spite of these difficulties, Sato [20] showed that if Ω is convex, the equation has a unique viscosity solution in $C(\overline{\Omega}\times[0,T))$ for any $T > 0$. Furthermore, if g is linear, Giga and Sato [11] proved that there is a unique viscosity solution even if Ω is not convex. Since we are interested in nonlinear functions g, we need the additional assumption that

$$\Omega \text{ is convex.}$$

It should be noted, however, that we only use this assumption through the use of the comparison principle from Sato [20]. So if a proof of existence of unique solutions is constructed for the case of non-convex Ω for nonlinear g, the proof of convergence of the algorithm will be valid for this case too.

Because of (1.4), the convergence of the algorithm also requires that

$$\text{(iv)} \quad g \in C^1(\mathbb{R}) \text{ and } \exists \zeta_1, \zeta_2 > 0 : \forall x \in \mathbb{R} : g'(x) \in (\zeta_1, \zeta_2), \tag{3.4}$$

which of course implies (ii) and (iii). But we will then also show how to get around this problem if we can find uniform approximations $g_\nu \to g$, where g fulfills (3.3), but has unbounded or zero derivative and g_ν fulfills (3.3) and (3.4) for all ν.

To formulate the approximation scheme, we choose non-negative, measurable, radially symmetric weight functions, ρ_1 and ρ_2 satisfying the conditions

$$\begin{aligned} &\text{(i)} \quad \int_{\mathbb{R}^n} \rho_i(x)\, dx < \infty \\ &\text{(ii)} \quad \int_{\mathbb{R}^{n-1}} \rho_i(\xi, 0)\, d\xi < \infty \\ &\text{(iii)} \quad \operatorname{supp} \rho_i \text{ is compact,} \end{aligned} \tag{3.5}$$

as well as the conditions (1.4) depending on g, which we state again for convenience:

$$\begin{aligned} &0 < \frac{b_1}{a_1} < g' < \frac{b_2}{a_2}, \\ &d = a_1b_2 - a_2b_1 > 0 \end{aligned}$$

where

$$\begin{aligned} a_i &= \int_{\mathbb{R}^{n-1}} \rho_i(y', 0)\, dy' \\ b_i &= \frac{1}{2}\int_{\mathbb{R}^{n-1}} y_1^2 \rho_i(y', 0)\, dy' \end{aligned}$$

Condition (3.5) (iii) is not really necessary and could be replaced by conditions for rapid decrease of ρ_i, but since the proofs are much simpler when we assume compact support, we use that assumption. See [14] for the proof when g is linear including the case when ρ has non-compact support.

We now define

$$N_i^C(x,h) = \int_\Omega \rho_i^{\sqrt{h}}(y-x)\chi_C(y)\, dy - \frac{1}{2}\int_\Omega \rho_i^{\sqrt{h}}(y-x)\, dy$$

for $i = 1, 2$ and a mapping $\mathcal{G}_h$, that maps subsets of $\mathbb{R}^n$ to subsets of $\mathbb{R}^n$, by

$$\mathcal{G}_h(C) = \{x \in \mathbb{R}^n \mid F(N_1^C(x,h), N_2^C(x,h)) \geq 0\} \tag{3.6}$$

for $C \subset \mathbb{R}^n$ with F defined by (1.3), that is

$$F(N_1, N_2) = \frac{1}{\sqrt{h}}\frac{b_2N_1 - b_1N_2}{d} - g\left(\frac{1}{\sqrt{h}}\frac{a_2N_1 - a_1N_2}{d}\right).$$

In order to prove that this mapping produces a generalized mean curvature flow as $h \to 0$, we need to connect it to the PDE (3.1). For that purpose, given a function $\varphi \in C(\overline{\Omega})$ and a real number λ, we consider the super-level set $\{\varphi \geq \lambda\} \equiv \{x \in \mathbb{R}^n \mid \varphi(x) \geq \lambda\}$ and set

$$\begin{aligned} N_i(\lambda) = N_i^{\{\varphi\geq\lambda\}}(x,h) &= \int_\Omega \rho_i^{\sqrt{h}}(y-x)\chi_{\{\varphi\geq\lambda\}}(y)\, dy - \frac{1}{2}\int_\Omega \rho_i^{\sqrt{h}}(y-x)\, dy, \\ \widetilde{N}_i(\lambda) = \widetilde{N}_i^{\{\varphi\geq\lambda\}}(x,h) &= \int_{\mathbb{R}^n} \rho_i^{\sqrt{h}}(y-x)\chi_{\{\varphi\geq\lambda\}}(y)\, dy - \frac{1}{2}\int_{\mathbb{R}^n} \rho_i^{\sqrt{h}}(y-x)\, dy, \end{aligned} \tag{3.7}$$

for i=1,2. This notation does not explicitly show the dependence on x, h and φ, but that will be clear from the context. Finally, we also define mappings $G_h, \widetilde{G}_h : C(\overline{\Omega}) \to C(\overline{\Omega})$, corresponding to $\mathcal{G}_h$, by

$$[G_h\varphi](x) = \sup\{\lambda \in \mathbb{R} \mid F(N_1(\lambda), N_2(\lambda)) \geq 0\} \tag{3.8}$$

$$[\widetilde{G}_h\varphi](x) = \sup\{\lambda \in \mathbb{R} \mid F(\widetilde{N}_1(\lambda), \widetilde{N}_2(\lambda)) \geq 0\} \tag{3.9}$$

for $h > 0$ and $\varphi \in C(\overline{\Omega})$. Note that all symbols with tilde ($\widetilde{G}$, $\widetilde{N}$ etc.) denote entities in the case of no boundary conditions, while the same symbols without the tilde denote the same entity in the domain Ω.

The main result, theorem 5.1, is now that the repeated application of the mappings G_h gives us an approximation of the solution to the level set equation (3.1). For clarity, we state the algorithm explicitly:

Algorithm 3.1
Given Ω and g, choose functions ρ_1 and ρ_2 according to the assumptions above.
Choose an initial set C_0.
For each iteration k,
  Choose a time-step t_k,
  For each point $x \in \Omega$,
    Calculate $N_i^{C_{k-1}}(x, t_k)$, $i = 1, 2$.
    Evaluate the function $F(N_1^{C_{k-1}}(x, t_k), N_2^{C_{k-1}}(x, t_k))$.
    If $F \geq 0$, let x belong to C_k.
  End loop
End loop

For efficient implementation of this and other BMO-type algorithms, see Ruuth [19] and Grzibovskis-Heintz [12].

4. Properties of G_h and $\mathcal{G}_h$

In this section, we prove some crucial properties of the operators $\mathcal{G}_h$ and G_h. We start with the inclusion principle for $\mathcal{G}_h$.

Proposition 4.1 *Let $\mathcal{G}_h$ be defined by (3.6). Then for all $h > 0$ and all closed sets $C_1, C_2 \subset \overline{\Omega}$, we have*

$$C_1 \subset C_2 \Rightarrow \mathcal{G}_h(C_1) \subset \mathcal{G}_h(C_2).$$

Proof. Since the weight functions ρ_i are positive, $C_1 \subset C_2$ implies that $N_i^{C_1} \leq N_i^{C_2}$, $i = 1, 2$ and since F is increasing in both arguments, we have $F(N_1^{C_1}, N_2^{C_1}) \leq F(N_1^{C_2}, N_2^{C_2})$ and therefore

$$\{F(N_1^{C_1}, N_2^{C_1}) \geq 0\} \subset \{F(N_1^{C_2}, N_2^{C_2}) \geq 0\}$$

and $\mathcal{G}_h(C_1) \subset \mathcal{G}_h(C_2)$. ∎

From this principle, some properties of G_h follow. Note that by the definitions of G_h and $\mathcal{G}_h$, the connection between the two is

$$[G_h\varphi](x) = \sup\{\lambda \in \mathbb{R} \mid x \in \mathcal{G}_h(\{\varphi \geq \lambda\})\}.$$

Proposition 4.2 *For all $h > 0$ and $u, u_1, u_2 \in C(\overline{\Omega})$,*

(i) $G_h(u + C) = G_h u + C$, *for all* $C \in \mathbb{R}$,

(ii) $G_h(\theta \circ u) = \theta \circ (G_h u)$, *for any increasing, continuous function* $\theta : \mathbb{R} \to \mathbb{R}$.

(iii) *if* $u_1(x) \leq u_2(x)$ *for all* $x \in \overline{\Omega}$, *then* $[G_h u_1](x) \leq [G_h u_2](x)$ *for all* $x \in \overline{\Omega}$,

(iv) $\|G_h u_1 - G_h u_2\| \leq \|u_1 - u_2\|$ *in sup-norm,*

(v) $G_h u(x) = \inf\{\lambda \in \mathbb{R} \mid F(N_1(\lambda), N_2(\lambda)) \leq 0\}$

Proof.

(i) This follows directly from the definition (3.8) of G_h.

(ii) This also follows from the definition (3.8) of G_h, since θ commutes with the taking of supremum.

(iii) Suppose $u_1 \leq u_2$ in $\overline{\Omega}$ and that there is an $x_0 \in \overline{\Omega}$ such that $[G_h u_1](x_0) > [G_h u_2](x_0)$. Set $\lambda_1 = [G_h u_1](x_0)$ and $\lambda_2 = [G_h u_2](x_0)$, i.e.

$$\lambda_1 = \sup\{\lambda \mid x_0 \in \mathcal{G}_h(\{u_1 \geq \lambda\})\}, \qquad \lambda_2 = \sup\{\lambda \mid x_0 \in \mathcal{G}_h(\{u_2 \geq \lambda\})\}, \tag{4.1}$$

so that $\lambda_1 > \lambda_2$. Then, for every $\varepsilon \geq 0$, since also $u_1 \leq u_2$,

$$\{u_1 \geq \lambda_1 - \varepsilon\} \subset \{u_2 \geq \lambda_1 - \varepsilon\} \subset \{u_2 \geq \lambda_2 - \varepsilon\},$$

which by proposition 4.1 implies

$$\mathcal{G}_h(\{u_1 \geq \lambda_1 - \varepsilon\}) \subset \mathcal{G}_h(\{u_2 \geq \lambda_2 - \varepsilon\}).$$

This in turn implies that $\lambda_1 \leq \lambda_2$ by (4.1), which is a contradiction. Therefore we must have $[G_h u_1](x) \leq [G_h u_2](x)$ for all $x \in \overline{\Omega}$ and we are done.

(iv) Let $u_1, u_2 \in C(\overline{\Omega})$. It is enough to prove that

$$\|(G_h u_1 - G_h u_2)^{\pm}\|_\infty \leq \|(u_1 - u_2)^{\pm}\|_\infty,$$

with $(\cdot)^+ = \max\{0, \cdot\}$ and $(\cdot)^- = -\min\{0, \cdot\}$. The result then follows immediately. It is also only necessary to prove the plus-case, since the minus-case follows by the same argument.
So, aiming for a contradiction, we assume that there is an $x_0 \in \overline{\Omega}$, such that

$$[G_h u_1](x_0) - [G_h u_2](x_0) > \|(u_1 - u_2)^+\|_\infty \equiv C.$$

Then, by (i), we get

$$[G_h u_1](x_0) - [G_h(u_2 + C)](x_0) = [G_h u_1](x_0) - [G_h u_2](x_0) - C > 0$$

But since $u_1 \leq (u_2 + C)$, this contradicts the result in (iii) with $u_1 = u_1$ and $u_2 = u_2 + C$.

(v) Fix $x \in \overline{\Omega}$, let r_i be the radius of the support of ρ_i for $i = 1, 2$ and assume $r_1 < r_2$. We are interested in values of λ for which $F(N_1(\lambda), N_2(\lambda)) = 0$. Since F is increasing in both variables and $F(0,0) = 0$, we must then have $N_1(\lambda) \geq 0$ and $N_2(\lambda) \leq 0$ or the other way around. Thus we must have

$$\{u \geq \lambda\} \cap B_n(x, r_i\sqrt{h}) \neq \varnothing \quad \text{and} \quad \{u \geq \lambda\} \cap B_n(x, r_i\sqrt{h}) \neq B_n(x, r_i\sqrt{h})$$

for $i = 1$ or 2 or both. The only other possibility would be $\{u \geq \lambda\} \cap B_n(x, r_1\sqrt{h}) = \varnothing$ and $\{u \geq \lambda\} \cap B_n(x, r_2\sqrt{h}) = B_n(x, r_2\sqrt{h})$, but this is impossible since $B_n(x, r_1\sqrt{h}) \subset B_n(x, r_2\sqrt{h})$.

But then, for any λ_1 and λ_2 near where $F = 0$, with $\lambda_1 < \lambda_2$, we have

$$\{u \geq \lambda_2\} \cap B_n(x, r_i\sqrt{h}) \subsetneq \{u \geq \lambda_1\} \cap B_n(x, r_i\sqrt{h})$$

and therefore $N_i(\lambda_2) < N_i(\lambda_1)$ with strict inequality for at least one of N_1 and N_2. We also have $N_i(\lambda_2) \leq N_i(\lambda_1)$ for the other one and therefore the function $\lambda \mapsto F(N_1(\lambda), N_2(\lambda))$ is strictly decreasing near the points where it is zero, and thus

$$[G_h u](x) \equiv \sup\{\lambda \in \mathbb{R} \mid F(N_1(\lambda), N_2(\lambda)) \geq 0\} = \inf\{\lambda \in \mathbb{R} \mid F(N_1(\lambda), N_2(\lambda)) \leq 0\}.$$

■

Finally, the following proposition, which is analogous to lemma 3.1 in [14], is a crucial part of the proof of our main theorem.

Proposition 4.3 *For all $\varphi \in C^2(\overline{\Omega})$, $z \in \overline{\Omega}$ and $\varepsilon > 0$, there is a $\delta > 0$ such that*

(i) *If $z \in \Omega$ and $\nabla\varphi(z) \neq 0$, then*

$$[G_h\varphi](x) \leq \varphi(x) + h|\nabla\varphi(z)|\, g(\operatorname{curv}(\varphi(z))) + \varepsilon h, \quad x \in B_n(z,\delta), h \in (0,\delta] \tag{4.2}$$

and

$$[G_h\varphi](x) \geq \varphi(x) + h|\nabla\varphi(z)|\, g(\operatorname{curv}(\varphi(z))) - \varepsilon h, \quad x \in B_n(z,\delta), h \in (0,\delta] \tag{4.3}$$

(ii) *If $z \in \partial\Omega$ and $\partial\varphi/\partial\hat{n}(z) > 0$, then (4.2) holds for all $x \in B_n(z,\delta) \cap \overline{\Omega}$ and $h \in (0,\delta]$.*

(iii) *If $z \in \partial\Omega$ and $\partial\varphi/\partial\hat{n}(z) < 0$, then (4.3) holds for all $x \in B_n(z,\delta) \cap \overline{\Omega}$ and $h \in (0,\delta]$.*

The proof uses the same idea as in [14], namely to compare this case to the problem without boundary conditions in the whole of $\mathbb{R}^n$. The analogue of proposition 4.3 for that case can be found in a different shape in Grzibovskis and Heintz [12, lemma 2] and is stated as lemma 4.4 below. The rest of the proof is mainly concerned with comparing $N_i(\lambda)$ and $\widetilde{N}_i(\lambda)$ (with and without boundary conditions). This requires the rather lengthy proofs of lemmas 4.5 and 4.6. Lemma 4.5 is proved exactly as in [14, lemma 3.1, case 1] and the proof is therefore not given here. The setup for that proof is identical to the one in the proof of lemma 4.6, which is given here and which we need because we have two convolution kernels instead of one.

Lemma 4.4 (see [12, lemma 2])
Let $\varphi \in C^2(\mathbb{R}^n)$ and $z \in \mathbb{R}^n$. If $\nabla\varphi(z) \neq 0$, then for each $\varepsilon > 0$, there is a $\delta > 0$ such that

$$[\widetilde{G}_h\varphi](x) \leq \varphi(x) + h|\nabla\varphi(z)|\, g(\operatorname{curv}(\varphi(z))) + \varepsilon h,$$
$$[\widetilde{G}_h\varphi](x) \geq \varphi(x) + h|\nabla\varphi(z)|\, g(\operatorname{curv}(\varphi(z))) - \varepsilon h$$

for all $x \in B_n(z,\delta)$ and $h \in (0,\delta)$.

Proof of proposition 4.3. As in [14], the main idea behind the proof is to compare $G_h\varphi(x)$ to $\widetilde{G}_h\varphi(x)$ and use lemma 4.4 to get the desired result.

Let $\varphi \in C^2(\overline{\Omega})$ and $\varepsilon > 0$. If $z \in \Omega$ and $D\varphi(z) \neq 0$, then since supp ρ_i is compact, there is a $\delta > 0$, such that for all $x \in B_n(z,\delta) \subset \Omega$ and $h \in (0,\delta]$, we have

$$[G_h\varphi](x) = \widetilde{G}_h\varphi(x).$$

So proposition 4.3, part (i), follows directly from lemma 4.4. To prove the rest of the proposition, we assume $z \in \partial\Omega$ and $\partial\varphi/\partial\hat{n}(z) > 0$ and prove part (ii), noting that part (iii) may be proved similarly.

Since $\varphi \in C^2(\overline{\Omega})$ and $\partial\Omega$ is C^2, we may extend φ so that $\varphi \in C^2(B_n(z, r_0))$ for some $r_0 > 0$. Using proposition 4.2 (v), we also set

$$\lambda \equiv [G_h\varphi](x) = \inf\{\mu \in \mathbb{R} \mid F(N_1(\mu), N_2(\mu)) \leq 0\}$$

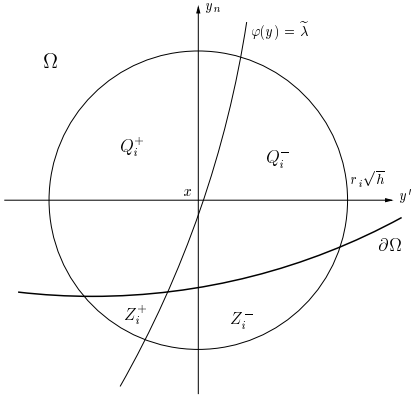

Figure 4.1: The sets $Q_i^\pm$ and $Z_i^\pm$ in case 1.

and

$$\widetilde{\lambda} \equiv [\widetilde{G}_h\varphi](x) = \inf\{\mu \in \mathbb{R} \mid F(\widetilde{N}_1(\mu), \widetilde{N}_2(\mu)) \le 0\}.$$

Now, we wish to show that there is an $h_0 > 0$ such that $\lambda \le \widetilde{\lambda}$ for each $x \in B_n(z, r_0) \cap \overline{\Omega}$ and each $h \in (0, h_0]$. By the definition of $\widetilde{\lambda}$, we know that $F(\widetilde{N}_1(\widetilde{\lambda}), \widetilde{N}_2(\widetilde{\lambda})) = 0$ and by the definition (1.3) of $F(N_1, N_2)$, we also know that $F(0, 0) = 0$. If we can show that $N_i(\widetilde{\lambda}) \le 0$, $i = 1, 2$, we would know that $F(N_1(\widetilde{\lambda}), N_2(\widetilde{\lambda})) \le 0$ (since F is increasing in both variables) and thus

$$\lambda = \inf\{\mu \in \mathbb{R} \mid F(N_1(\mu), N_2(\mu)) \le 0\} \le \widetilde{\lambda}. \tag{4.4}$$

The same result would be obtained if we could show that $N_i(\widetilde{\lambda}) \le \widetilde{N}_i(\widetilde{\lambda})$, $i = 1, 2$.

To prove that $N_i(\widetilde{\lambda}) \le 0$ or that $N_i(\widetilde{\lambda}) \le \widetilde{N}_i(\widetilde{\lambda})$, we first define

$$Q_i^+ \equiv \{y \in B_n(x, r_i\sqrt{h}) \mid \varphi(y) < \widetilde{\lambda}\} \cap \Omega,$$
$$Q_i^- \equiv \{y \in B_n(x, r_i\sqrt{h}) \mid \varphi(y) > \widetilde{\lambda}\} \cap \Omega,$$

for $i = 1, 2$, with $r_i = \inf\{r \in \mathbb{R} \mid \operatorname{supp} \rho_i \subset B_n(0, r)\}$. See figure 4.1. We then consider one ρ_i at a time (i.e. $i = 1$ or 2) and divide the treatment into three cases (cf. figures 4.1 and 4.2).

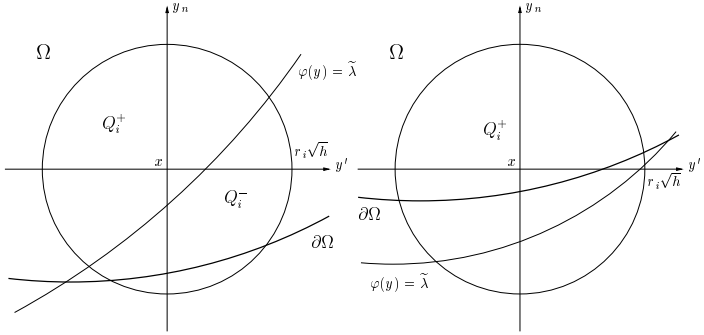

Figure 4.2: Case 2 on the left and case 3 on the right.

Case 1: $\{y \in B_n(x, r_i\sqrt{h}) \mid \varphi(y) = \widetilde{\lambda}, y \in \partial\Omega\} \neq \varnothing$
In general, we have

$$\begin{aligned} N_i(\widetilde{\lambda}) &= \int_\Omega \rho_i^{\sqrt{h}}(y - x)\chi_{\{\varphi \ge \widetilde{\lambda}\}}(y)\, dy - \frac{1}{2}\int_\Omega \rho_i^{\sqrt{h}}(y - x)\, dy \\ &= \int_{Q_i^-} \rho_i^{\sqrt{h}}(y - x)\, dy - \frac{1}{2}\int_{Q_i^+ \cup Q_i^-} \rho_i^{\sqrt{h}}(y - x)\, dy \\ &= \frac{1}{2}\left(\int_{Q_i^-} \rho_i^{\sqrt{h}}(y - x)\, dy - \int_{Q_i^+} \rho_i^{\sqrt{h}}(y - x)\, dy\right). \end{aligned}$$

Lemma 4.5 below now shows that $N_i(\widetilde{\lambda}) \le 0$ and lemma 4.6 gives $N_i(\widetilde{\lambda}) \le \widetilde{N}_i(\widetilde{\lambda})$.

Case 2: $\{y \in B_n(x, r_i\sqrt{h}) \mid \varphi(y) = \widetilde{\lambda}\} \subset \Omega$
In this case, the intersection of the hypersurfaces $\{\varphi = \widetilde{\lambda}\}$ and $\partial\Omega$ lies outside the ball $B_n(x, r_i\sqrt{h})$. We then see that $Q_i^+ \cap \Omega^c = \varnothing$ and therefore, starting as before, we get

$$\begin{aligned} N_i(\widetilde{\lambda}) &= \frac{1}{2}\left(\int_{Q_i^-} \rho_i^{\sqrt{h}}(y - x)\, dy - \int_{Q_i^+} \rho_i^{\sqrt{h}}(y - x)\, dy\right) \\ &\le \frac{1}{2}\left(\int_{\{\varphi \ge \widetilde{\lambda}\}} \rho_i^{\sqrt{h}}(y - x)\, dy - \int_{\{\varphi < \widetilde{\lambda}\}} \rho_i^{\sqrt{h}}(y - x)\, dy\right) = \widetilde{N}_i(\widetilde{\lambda}). \end{aligned}$$

Case 3: $\{y \in B_n(x, r_i\sqrt{h}) \mid \varphi(y) = \widetilde{\lambda}\} \subset \Omega^c$
In this case, $Q_i^- = \varnothing$ and since ρ_i is positive, we can conclude $N_i(\widetilde{\lambda}) \le 0$ from the expression in Case 1.

Now, we need to see what happens when we have two convolution kernels. Set $r_i = \inf\{r \in \mathbb{R} \mid \operatorname{supp} \rho_i \subset B_n(0,r)\}$, $i = 1, 2$ and assume $r_1 < r_2$. Then, if case 1 applies to ρ_1, then it obviously also applies to ρ_2, so that $N_i(\widetilde{\lambda}) \le 0$ (and $N_i(\widetilde{\lambda}) \le \widetilde{N}_i(\widetilde{\lambda})$) for $i = 1, 2$ and therefore $\lambda \le \widetilde{\lambda}$ by the argument leading to (4.4). If case 3 applies to ρ_1, then either case 1 or 3 applies to ρ_2, but in both cases we have again $N_i(\widetilde{\lambda}) \le 0$ for $i = 1, 2$ and $\lambda \le \widetilde{\lambda}$. Finally, if case 2 applies to ρ_1, then either case 1 or 2 applies to ρ_2. In both cases, we have $N_i(\widetilde{\lambda}) \le \widetilde{N}_i(\widetilde{\lambda})$ for $i = 1, 2$ and thus $\lambda \le \widetilde{\lambda}$. Therefore, once we prove lemmas 4.5 and 4.6, the proof is completed.

■

All that now remains is to prove the next two lemmas. As mentioned earlier, lemma 4.5 is proved by Ishii and Ishii [14] for regular mean curvature flow and there are only very minor differences in our case, so we omit the proof. Besides, the same setup is used in the proof of lemma 4.6, which we give in detail and which is needed in our case in view of the discussion following the three cases above, because we have two convolution kernels.

Lemma 4.5 (cf. [14, lemma 3.1, case 1])
Let r_i, Q_i^+ and Q_i^- be defined as above, and assume that $z \in \partial\Omega$ and $\partial\varphi/\partial\hat{n}(z) > 0$. Then there is an $r_0 > 0$ and an $h_0 > 0$ such that for all $x \in B_n(z, r_0)$ and $h \in (0, h_0)$, it holds that if $\{y \in B_n(x, r_i\sqrt{h}) \mid \varphi(y) = \widetilde{\lambda}, y \in \partial\Omega\} \neq \varnothing$ (i.e. case 1 applies) then

$$\int_{Q_i^+} \rho_i^{\sqrt{h}}(y - x)\, dy \ge \int_{Q_i^-} \rho_i^{\sqrt{h}}(y - x)\, dy, \tag{4.5}$$

and thus $N_i(\widetilde{\lambda}) \le 0$.

Lemma 4.6 *Let r_i be defined as above, and assume $z \in \partial\Omega$ and $\partial\varphi/\partial\hat{n}(z) > 0$. Then there is an $r_0 > 0$ and an $h_0 > 0$ such that for all $x \in B_n(z, r_0)$ and $h \in (0, h_0)$ it holds that if $\{y \in B_n(x, r_i\sqrt{h}) \mid \varphi(y) = \widetilde{\lambda}, y \in \partial\Omega\} \neq \varnothing$ (i.e. case 1 holds), then*

$$N_i(\widetilde{\lambda}) \le \widetilde{N}_i(\widetilde{\lambda}).$$

Proof. We wish to prove that $N_i(\widetilde{\lambda}) \le \widetilde{N}_i(\widetilde{\lambda})$, which is the same as

$$\int_{Q_i^-} \rho_i^{\sqrt{h}}(y - x)\, dy - \int_{Q_i^+} \rho_i^{\sqrt{h}}(y - x)\, dy \le \int_{\widetilde{Q}_i^-} \rho_i^{\sqrt{h}}(y - x)\, dy - \int_{\widetilde{Q}_i^+} \rho_i^{\sqrt{h}}(y - x)\, dy,$$

with $Q_i^\pm$ defined above and

$$\widetilde{Q}_i^+ = \{y \in B_n(x, r_i\sqrt{h}) \mid \varphi(y) < \widetilde{\lambda}\}$$
$$\widetilde{Q}_i^- = \{y \in B_n(x, r_i\sqrt{h}) \mid \varphi(y) > \widetilde{\lambda}\},$$

so that $Q_i^\pm = \widetilde{Q}_i^\pm \cap \Omega$. Rewriting this expression once more, we get

$$\int_{Z_i^-} \rho_i^{\sqrt{h}}(y - x)\, dy - \int_{Z_i^+} \rho_i^{\sqrt{h}}(y - x)\, dy \ge 0 \tag{4.6}$$

with

$$Z_i^+ = \{y \in B_n(x, r_i\sqrt{h}) \cap \Omega^c \mid \varphi(y) < \widetilde{\lambda}\},$$
$$Z_i^- = \{y \in B_n(x, r_i\sqrt{h}) \cap \Omega^c \mid \varphi(y) > \widetilde{\lambda}\}$$

as is illustrated in figure 4.1.

Given $z \in \partial\Omega$ with $\partial\varphi/\partial\hat{n}(z) > 0$, there is an $r_0 > 0$ such that φ can be extended to be in $C^2(B_n(z, r_0))$ (since $\partial\Omega$ is C^2 and $\varphi \in C^2(\overline{\Omega})$). The idea of the proof is now that for any $x \in B_n(z, r_0) \cap \overline{\Omega}$, we may approximate the hypersurfaces $\{\varphi = \widetilde{\lambda}(x)\}$ and $\partial\Omega$ by hyperplanes and show that for small h, since $\partial\varphi/\partial\hat{n} > 0$, the contribution from the set denoted W in figure 4.3 is greater than the contributions from the the sets enclosed by dashed lines, within which the two hypersurfaces must lie. But first we need to introduce some notation.

We take a parameterization $\psi \in C^2(\mathbb{R}^{n-1})$ of $\partial\Omega$ and choose coordinate system so that

$$y_n - z_n = \psi(y' - z') \text{ for all } y = (y', y_n) \in B_n(z, r_0) \cap \partial\Omega,$$
$$\nabla'\psi(0) = 0,$$
$$y_n - z_n > \psi(y' - z') \text{ for all } y \in B_n(z, r_0) \cap \Omega,$$

where $\nabla' = (\partial/\partial x_1, \dots, \partial/\partial x_{n-1})$. Taking r_0 smaller if necessary, we may assume

$$\nabla_n\varphi(y) \le -\gamma, \quad |\nabla'\varphi(y)| \le K, \quad |\nabla'\psi(y' - z')| \le \varepsilon \text{ for all } y \in B_n(z, r_0),$$

where $\nabla_n = \partial/\partial x_n$ and $\gamma, K > 0$ are independent of $\varepsilon > 0$ and r_0.

Now fix $x \in B_n(z, r_0) \cap \overline{\Omega}$ and a small $h_0 > 0$. Let $h \in (0, h_0)$ and choose a point $\xi \in \partial\Omega \cap \{\varphi = \widetilde{\lambda}\} \cap B_n(x, r_i\sqrt{h})$. We then set

$$a(x) = \frac{\nabla'\varphi(x)}{\nabla_n\varphi(x)}, \qquad b(\xi') = \nabla'\psi(\xi')$$

and define the hyperplanes A and B by

$$A = \{y \in \mathbb{R}^n \mid \langle (a, -1), y - x\rangle = 0\},$$
$$B = \{y \in \mathbb{R}^n \mid \langle (b, -1), y - \xi\rangle = 0\} = \{y \in \mathbb{R}^n \mid \langle (b, -1), y - x\rangle = c\},$$

with $c = \langle (b, -1), \xi - x\rangle$, see figure 4.3. We also note that $|a| \le K/\gamma$ and $|b| \le \varepsilon$, so that

$$|\langle (a, -1), (b, -1)\rangle| = |\langle a, b\rangle + 1| \ge 1 - \frac{\varepsilon K}{\gamma} \ge \frac{1}{2} \tag{4.7}$$

if $\varepsilon \in (0, \varepsilon_0)$ with $\varepsilon_0 = \gamma/(2K)$, which in particular means that A and B are not perpendicular.

We now need three elementary lemmas with counterparts in [14].

Lemma 4.7 (cf. Ishii and Ishii [14, lemma 3.4])
There exists a $\delta > 0$ and a $C_1 > 0$ independent of $x \in B_n(z, \delta)$ such that if $h \in (0, \delta)$ and $x \in B_n(z, \delta)$, then $|\widetilde{\lambda} - \varphi(x)| \le C_1 h$.

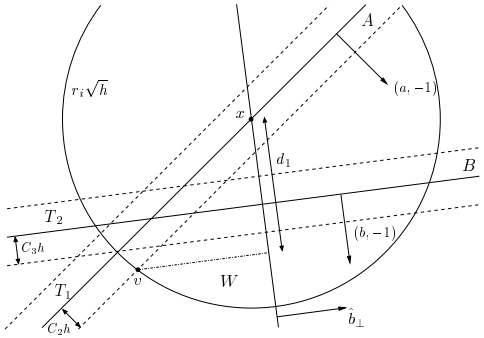

Figure 4.3: The plane spanned by $(a,-1)$ and $(b,-1)$, showing the hyperplanes A and B as well as the position of v and the distance d_1.

Proof. This follows directly from lemma 4.4, using the fact that $\nabla\varphi(z) \neq 0$. ∎

Lemma 4.8 (cf. [14, lemma 3.6])
There exists a $C_2 > 0$, which is independent of $x \in B_n(z, r_0)$, and an $h_1 > 0$, such that if $y \in B_n(x, r_i\sqrt{h})$ satisfies $\varphi(y) = \widetilde{\lambda}$, with $h \in (0, h_1)$, and if $a(x) = \nabla'\varphi(x)/\nabla_n\varphi(x)$, then $|y_n - x_n - \langle a(x), y' - x'\rangle| \leq C_2 h$.

Proof. This follows easily from lemma 4.7. See [14] for details. ∎

Lemma 4.9 (cf. [14, lemma 3.3 (i)])
For any $r > 0$, $a, b \in \mathbb{R}^{n-1}$ and $c \in \mathbb{R}^n$, with $|a| \leq K_1$, $|b| \leq \varepsilon_1$, and $\varepsilon_1 K_1 < 1$, there is a $\theta \in (0,1)$, depending only on ε_1 and K_1, such that if

$$\{y \in B_n(0,r) \mid y_n = \langle b, y'\rangle + c,\ \ y_n = \langle a, y'\rangle\} \neq \varnothing,$$

then

$$\frac{|c|}{\sqrt{1+|b|^2}} \leq \theta r.$$

(Note that the quantity on the left is the orthogonal distance from the origin to the hyperplane $\langle (b,-1), y\rangle = c$).

Proof. The proof is given in [14] and is a straightforward geometrical argument. ∎

By lemma 4.8, we have

$$\left|\langle (a,-1), y - x\rangle\right| \leq C_2 h, \qquad \text{if } \varphi(y) = \widetilde{\lambda},\ \ y \in B_n(x, r_i\sqrt{h}), \tag{4.8}$$

and since $\psi \in C^2(\mathbb{R}^{n-1})$, we also have

$$\left|\langle (b,-1), y - x\rangle - c\right| \leq C_3 h, \qquad \text{for all } y \in \partial\Omega \cap B_n(x, r_i\sqrt{h}), \tag{4.9}$$

for some $C_3 > 0$, independent of $x \in B_n(z, r_0)$ and $h \in (0, h_0)$.

We now restrict our view to the plane spanned by $(a,-1)$ and $(b,-1)$, which is obviously perpendicular to both A and B. This is the view shown in figure 4.3.

We set

$$v = x + \beta_1(a,-1) + \beta_2(b,-1),$$

with

$$\beta_1 = \frac{1}{(1+|a|^2)^{1/2}}\left(C_2 h - \frac{1+\langle a,b\rangle}{D^{1/2}}\sqrt{r_i^2 h - C_2^2 h^2}\right),$$
$$\beta_2 = \frac{(1+|a|^2)^{1/2}}{D^{1/2}}\sqrt{r_i^2 h - C_2^2 h^2}$$

defined so that v is located as in figure 4.3, that is $\langle (a,-1), v - x\rangle/(1+|a|^2)^{1/2} = C_2 h$ and $|v - x| = r_i\sqrt{h}$. Here, $D = (1+|a|^2)(1+|b|^2) - (1+\langle a,b\rangle)^2$. Expanding this expression, it is easily seen that $D \geq |a-b|^2$, so that it is zero only when $a = b$. Assume for the moment that $a \neq b$, so that $D \neq 0$ and v is well-defined.

Now,

$$\begin{aligned} d_1 &= \frac{\langle v - x, (b,-1)\rangle}{(1+|b|^2)^{1/2}} \\ &= \frac{1+\langle a,b\rangle}{(1+|a|^2)^{1/2}(1+|b|^2)^{1/2}}\left(C_2 h - \frac{1+\langle a,b\rangle}{D^{1/2}}\sqrt{r_i^2 h - C_2^2 h^2}\right) \\ &\quad + \frac{(1+|a|^2)^{1/2}(1+|b|^2)^{1/2}}{D^{1/2}}\sqrt{r_i^2 h - C_2^2 h^2} \\ &= \left(\frac{D}{(1+|a|^2)(1+|b|^2)}\right)^{1/2}\sqrt{r_i^2 h - C_2^2 h^2} + \frac{1+\langle a,b\rangle}{(1+|a|^2)^{1/2}(1+|b|^2)^{1/2}} C_2 h. \end{aligned}$$

Investigating the first term, we see first that $0 < \sqrt{r_i^2 h - C_2^2 h^2} \leq r_i\sqrt{h}$ if $h \in (0, h_0)$ with $h_0 < r_i^2/C_2^2$. Using (4.7), we also see that

$$\begin{aligned} \frac{D}{(1+|a|^2)(1+|b|^2)} &= \frac{(1+|a|^2)(1+|b|^2) - (1+\langle a,b\rangle)^2}{(1+|a|^2)(1+|b|^2)} = 1 - \frac{(1+\langle a,b\rangle)^2}{(1+|a|^2)(1+|b|^2)} \\ &\leq 1 - \frac{1}{4(1+(K/\gamma)^2)(1+\varepsilon_0^2)} \equiv \theta_1^2 < 1. \end{aligned}$$

For the second term, we may now use (4.7) to get the estimate

$$\frac{1+\langle a,b\rangle}{(1+|a|^2)^{1/2}(1+|b|^2)^{1/2}}C_2h \le \frac{3}{2}C_2h$$

Thus,

$$d_1 \le \theta_1 r_i\sqrt{h} + \frac{3}{2}C_2h \le \frac{\theta_1+1}{2}r_i\sqrt{h} < r_i\sqrt{h} \tag{4.10}$$

if $h \in (0,h_0)$ with $h_0 \le (r_i(1-\theta_1)/(3C_2))^2$.

Also, by lemma 4.9, there is a $\theta_2 \in (0,1)$ such that $|c|/\sqrt{1+|b|^2} \le \theta_2 r_i\sqrt{h}$. We note that

$$\theta_2 r_i\sqrt{h} + C_3h \le \frac{\theta_2+1}{2}r_i\sqrt{h} < r_i\sqrt{h} \tag{4.11}$$

if $h \in (0,h_0)$ with $h_0 \le (r_i(1-\theta_2)/(2C_3))^2$. Therefore, we set $\theta = \max\{\theta_1+1, \theta_2+1\}/2 < 1$ and $h_0 \le \min\{(r_i(1-\theta_1)/(3C_2))^2, (r_i(1-\theta_2)/(2C_3))^2\}$.

We are now ready to estimate the integrals over Z^+ and Z^- in (4.6). Letting $\hat{b}_\perp$ denote the unit vector in the $(a,-1),(b,-1)$-plane with $\langle(b,-1),\hat{b}_\perp\rangle = 0$ and $\langle(a,-1),\hat{b}_\perp\rangle > 0$, we define

$$\begin{aligned}
W &= \{y \in B_n(x, r_i\sqrt{h}) \mid \langle(b,-1), y-x\rangle/\sqrt{1+|b|^2} \ge \theta r_i\sqrt{h},\ \langle\hat{b}_\perp, y-x\rangle < 0\}\\
T_1 &= \{y \in B_n(x, r_i\sqrt{h}) \mid |\langle(a,-1), y-x\rangle| \le C_2h\}\\
T_2 &= \{y \in B_n(x, r_i\sqrt{h}) \mid |\langle(b,-1), y-x\rangle - c| \le C_3h\}
\end{aligned}$$

(see figure 4.3) and note that

$$\begin{aligned}
Z^- &\supset (\{y \in B_n(x, r_i\sqrt{h}) \mid \langle(b,-1), y-x\rangle > c + C_3h,\ \langle\hat{b}_\perp, y-x\rangle > 0\} \setminus T_1) \cup W,\\
Z^+ &\subset (\{y \in B_n(x, r_i\sqrt{h}) \mid \langle(b,-1), y-x\rangle > c - C_3h,\ \langle\hat{b}_\perp, y-x\rangle < 0\} \cup T_1) \setminus W.
\end{aligned}$$

The definition of θ ensures that $W \neq \varnothing$. From this, it is clear that

$$\int_{Z^-}\rho_i^{\sqrt{h}}(y-x)\,dy - \int_{Z^+}\rho_i^{\sqrt{h}}(y-x)\,dy \ge 2\int_W \rho_i^{\sqrt{h}}(y-x)\,dy - \left(\int_{T_1}+\int_{T_2}\right)\rho_i^{\sqrt{h}}(y-x)\,dy \tag{4.12}$$

Now, we see from (4.10) and (4.11) that

$$\mathcal{L}^n(W) = \alpha h^{n/2},$$

for some $\alpha > 0$ depending only on θ and r_i, where $\mathcal{L}^n$ is the n-dimensional Lebesgue measure. From this we conclude that

$$\int_W \rho_i^{\sqrt{h}}(y-x)\,dy \ge C_4,$$

for some $C_4 > 0$, independent of $x \in B_n(z,r_0)$, $\varepsilon \in (0,\varepsilon_0)$ and $h \in (0,h_0)$.

Furthermore, it is clear that $\mathcal{L}^n(T_1) \le C_5h^{(n+1)/2}$ for some $C_5 > 0$ depending only on C_2, r_i and n, and that $\mathcal{L}^n(T_2) \le C_6h^{(n+1)/2}$ for some C_6 depending only on C_3, r_i and n. Thus, by changing variables $(y-x)/\sqrt{h} \to \hat{y}$, we see

$$\begin{aligned}
&\left(\int_{T_1}+\int_{T_2}\right)\rho_i^{\sqrt{h}}(y-x)\,dy = \left(\int_{T_1(h,x)}+\int_{T_2(h,x)}\right)\rho_i(\hat{y})\,d\hat{y}\\
&\mathcal{L}^n(T_1(h,x)) \le C_5\sqrt{h}, \qquad \mathcal{L}^n(T_2(h,x)) \le C_6\sqrt{h}\\
&T_j(h,x) = \{(y-x)/\sqrt{h} \mid y \in T_j\},\ j=1,2.
\end{aligned}$$

Therefore, we conclude that there is an $h_0 > 0$ such that if $h \in (0,h_0)$, then

$$2\int_W \rho_i^{\sqrt{h}}(y-x)\,dy - \left(\int_{T_1}+\int_{T_2}\right)\rho_i^{\sqrt{h}}(y-x)\,dy \ge 0,$$

for all $x \in B_n(z,r_0)$, which proves $N_i(\widetilde{\lambda}) \le \widetilde{N}_i(\widetilde{\lambda})$ by (4.12) and the discussion leading to (4.6).

Finally, we need to cover the case $a = b$. In this case, we define

$$W = \{y \in B_n(x, r_i\sqrt{h}) \mid \langle(b,-1), y-x\rangle > \max\{c + C_3h, C_2h\}\},$$

with $W \neq \varnothing$ if h is small enough. Then $Z^- \supset W \setminus T_1$ and $Z^+ \subset T_1 \cup T_2$, and the same argument holds.

■

5. The convergence theorem

In this section, we prove the convergence of the output of algorithm 3.1 to the viscosity solution of the level set PDE (3.1) as the time step h tends to zero. The proof is based on the proofs by Ishii and Ishii [14] and Ishii [13].

We begin by defining the approximations u^m as follows. Given a function $f \in C(\overline{\Omega})$, let $u^m \in C(\overline{\Omega} \times [0,T))$ be defined for $m \in \mathbb{Z}_+$ by

$$u^m(x,t) = [G_{t-lh} \circ (G_h)^l f](x), \tag{5.1}$$

where $h = T/m$ and $l \in \mathbb{N}$ is chosen so that $lh \leq t < (l+1)h$. The main convergence theorem is then the following.

Theorem 5.1 *Choose $f \in C(\overline{\Omega})$ and let $\{u^m\}_{m=1}^{\infty}$ be defined by (5.1). Then $u^m \to u$ locally uniformly on $\overline{\Omega} \times [0,T)$ as $m \to \infty$, where u is the unique viscosity solution of the PDE*

$$\begin{cases} \dfrac{\partial u}{\partial t}(x,t) - |\nabla u(x,t)| g(\operatorname{curv}(u(x,t))) = 0 & x \in \Omega,\ t \in (0,T), \\ \dfrac{\partial u}{\partial \hat{n}}(x,t) = 0 & x \in \partial\Omega,\ t \in (0,T), \\ u(x,0) = f(x) & x \in \overline{\Omega}, \end{cases}$$

which exists by theorem 3.12 in Sato [20].

The idea of the proof is to define $\overline{u}(x,t)$ and $\underline{u}(x,t)$ by

$$\begin{aligned} \overline{u}(x,t) &= \lim_{\varepsilon \to 0} \sup\{u^m(y,s) \mid m > \varepsilon^{-1}, (y,s) \in \overline{\Omega} \times [0,T), |x-y| + |s-t| < \varepsilon\}, \\ \underline{u}(x,t) &= \lim_{\varepsilon \to 0} \inf\{u^m(y,s) \mid m > \varepsilon^{-1}, (y,s) \in \overline{\Omega} \times [0,T), |x-y| + |s-t| < \varepsilon\}, \end{aligned} \tag{5.2}$$

and prove that these are sub- and supersolution respectively of the level set PDE. It then follows from the comparison result by Sato [20] (theorem 5.2 below) that $\overline{u} \leq \underline{u}$ and thus (since $\overline{u} \geq \underline{u}$ by definition) that $u = \overline{u} = \underline{u}$ is a solution.

First, we state the definition of a viscosity solution to the PDE (3.1) in accordance with the definitions given in section 2. However, it is convenient to give the definition using test functions instead of semijets as follows.

Definition 5.1 *A function $u \in C(\overline{\Omega} \times [0,T))$ is a viscosity* subsolution *of (3.1) if for any $\varphi \in C^2(\overline{\Omega} \times [0,T))$ such that $u - \varphi$ has a maximum at $(x_0,t_0) \in \overline{\Omega} \times [0,T)$, then*

$$\varphi_t'(x_0,t_0) - |\nabla\varphi(x_0,t_0)| g\big(\operatorname{curv}(\varphi(x_0,t_0))\big) \leq 0$$

if $x_0 \in \Omega$ and $\nabla\varphi(x_0,t_0) \neq 0$ or $x_0 \in \partial\Omega$ and $\partial\varphi/\partial\hat{n}(x_0,t_0) > 0$, and

$$\varphi_t'(x_0,t_0) \leq 0$$

if $x_0 \in \Omega$, $\nabla\varphi(x_0,t_0) = 0$ and $D^2\varphi(x_0,t_0) = 0$.

$u \in C(\overline{\Omega} \times [0,T))$ is a viscosity supersolution *of (3.1) if for any $\varphi \in C^2(\overline{\Omega} \times [0,T))$ such that $u - \varphi$ has a minimum at $(x_0,t_0) \in \overline{\Omega} \times [0,T)$, then*

$$\varphi_t'(x_0,t_0) - |\nabla\varphi(x_0,t_0)| g\big(\operatorname{curv}(\varphi(x_0,t_0))\big) \geq 0$$

if $x_0 \in \Omega$ and $\nabla\varphi(x_0,t_0) \neq 0$ or $x_0 \in \partial\Omega$ and $\partial\varphi/\partial\hat{n}(x_0,t_0) < 0$, and

$$\varphi_t'(x_0,t_0) \geq 0$$

if $x_0 \in \Omega$, $\nabla\varphi(x_0,t_0) = 0$ and $D^2\varphi(x_0,t_0) = 0$.

u is a viscosity solution *if it is both a sub- and a supersolution.*

We remark that there is no condition in the case $\nabla\varphi(x_0,t_0) = 0$, $D^2\varphi(x_0,t_0) \neq 0$, since it follows from the other cases by the argument given by Barles and Georgelin [3, proposition 2.2], using the hypothesis (3.3) (ii) that g must not grow faster than linearly towards infinity.

For clarity, we also state the comparison result by Sato [20] in slightly reduced form.

Theorem 5.2 ([20], theorem 2.1)
Let $u(x,t)$ be a subsolution and $v(x,t)$ a supersolution of (3.1) according to definition 5.1. If $u(x,0) \leq v(x,0)$ in $\overline{\Omega}$, then $u(x,t) \leq v(x,t)$ in $\overline{\Omega} \times [0,T)$.

Before we begin with the actual proof of the convergence theorem, we also need a few lemmas.

Lemma 5.3 *For each $z \in \Omega$, there is a $C > 0$, a $\delta > 0$ and an $h_0 > 0$ such that*

$$\begin{aligned} [G_h(|\cdot - z|^2)](x) &\leq |x-z|^2 + Ch, \\ [G_h(-|\cdot - z|^2)](x) &\geq -|x-z|^2 - Ch \end{aligned}$$

for all $x \in B_n(z,\delta)$ and $h \in (0,h_0)$.

Proof. Fix $z \in \Omega$. Since the convolution kernels ρ_i have compact support, there is an $R > 0$, such that supp $\rho_i^{\sqrt{h}}(\cdot - x) \subset B_n(x, R\sqrt{h})$, $i = 1,2$, for any $x \in \overline{\Omega}$ and any $h > 0$. Furthermore, there is a $\delta > 0$ and an $h_0 > 0$ such that if $h \in (0,h_0)$, then $B_n(x, R\sqrt{h}) \subset \Omega$ for any $x \in B_n(z,\delta)$.

We fix such R, h_0, δ and an $x \in B_n(z,\delta)$. In order to estimate $[G_h(|\cdot - z|^2)](x)$, we need to see which level sets of the function $\xi \mapsto |\xi - z|^2$ that may reach x. Therefore we investigate the situation in figure 5.1, where y is on the level set $\{\xi \in \Omega \mid |\xi - z|^2 = \lambda\}$, which is chosen so that it intersects $\partial B_n(x, R\sqrt{h})$ on the hyperplane $\{\xi \in \mathbb{R}^n \mid \langle \xi - x, x - z\rangle = 0\}$.

Since $F(N_1,N_2)$ is increasing in both variables and $F(0,0) = 0$, we need to have either $N_1 \geq 0$ or $N_2 \geq 0$ in order to get $F(N_1,N_2) \geq 0$. And since in the situation in figure 5.1, no part of the super-level set $\{\xi \in \Omega \mid |\xi - z|^2 \geq \lambda\}$ is in the left half of $B_n(x, R\sqrt{h})$, obviously $N_i(x,h) < 0$, $i = 1,2$ and thus $F(N_1,N_2) < 0$ so that

$$x \notin \mathcal{G}_h(\{\xi \in \Omega \mid |\xi - z|^2 \geq \lambda = |y - z|^2\}).$$

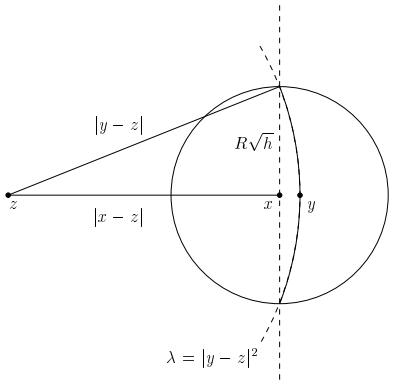

Figure 5.1: The setting in the proof of lemma 5.3.

Therefore,

$$[G_h(|\cdot - z|^2)](x) - |x-z|^2 \le |y-z|^2 - |x-z|^2 = (R\sqrt{h})^2 = R^2 h$$

by the Pythagorean theorem, which gives the desired result with $C = R^2$.

A similar argument gives the result for $-|x-z|^2$.

■

Lemma 5.4 *Let $f \in C^2(\overline{\Omega})$ in (5.1). If $\partial f/\partial \hat{n} > 0$ on $\partial\Omega$, then there is a constant $C > 0$ and an $M \in \mathbb{Z}_+$, such that*

$$\sup_{x\in\overline{\Omega}, m\ge M} (u^m(x,t) - f(x)) \le Ct$$

for all $t \in [0,T)$. If instead $\partial f/\partial \hat{n} < 0$ on $\partial\Omega$, then

$$\inf_{x\in\overline{\Omega}, m\ge M} (u^m(x,t) - f(x)) \ge -Ct.$$

Proof. We assume $\partial f/\partial \hat{n} > 0$ on $\partial\Omega$ and prove the first inequality. The other case may be proved similarly.

We prove that there is a $C > 0$ and an $h_0 > 0$ such that

$$G_h f(x) \le f(x) + Ch, \qquad \text{for all } x \in \overline{\Omega},\ h \in (0, h_0), \tag{5.3}$$

which, if M is chosen so that $T/M \le h_0$, may be iterated to give the desired result.

We fix $z \in \overline{\Omega}$. If $\nabla f(z) \neq 0$, it follows from proposition 4.3 with $\varepsilon = 1$ that there is a $\delta_1 > 0$ such that

$$G_h f(x) \le f(x) + (|\nabla f(z)| g(\mathrm{curv}(f(z))) + 1)h$$

holds for all $x \in B_n(z,\delta_1) \cap \overline{\Omega}$ and $h \in (0,\delta_1)$.

Now suppose that $\nabla f(z) = 0$. Since we assume $\partial f/\partial \hat{n} > 0$ on $\partial\Omega$, we then have $z \in \Omega$. Since $f \in C^2(\overline{\Omega})$, there is a $\delta_2 > 0$, such that

$$|f(x) - f(z)| \le \|D^2 f\|_\infty |x-z|^2 \tag{5.4}$$

for all $x \in B_n(z,\delta_2) \subset \Omega$. Taking $C_1 = 2\|f\|_\infty/\delta_2^2 + \|D^2 f\|_\infty$, we then get

$$f(x) \le f(z) + C_1 |x-z|^2$$

for all $x \in \overline{\Omega}$. Applying G_h to both sides of this inequality with z fixed, and using in turn proposition 4.2 (i)-(iii), lemma 5.3 and (5.4), we get

$$\begin{aligned} G_h f(x) &\le f(z) + C_1[G_h(|\cdot - z|^2)](x) \le f(z) + C_1(|x-z|^2 + C_2 h) \\ &\le f(x) + (C_1 + \|D^2 f\|_\infty)|x-z|^2 + C_1 C_2 h \end{aligned}$$

for all $x \in B_n(z,\delta_3)$ and $h \in (0,h_1)$ for some $h_1 > 0$ and $\delta_3 > 0$.

That is, for any $z \in \overline{\Omega}$ and any $\varepsilon > 0$, there are $C > 0$ and $h_0 > 0$ independent of ε and a $\delta = \delta(\varepsilon) > 0$ such that

$$G_h f(x) \le f(x) + Ch + \varepsilon \tag{5.5}$$

for all $x \in B_n(z,\delta)$ and $h \in (0,h_0)$. Since $\overline{\Omega}$ is compact, it may be covered by a finite number of such δ-neighborhoods and taking the largest of the C:s and the smallest of the h_0:s, (5.5) holds for all $x \in \overline{\Omega}$ and $h \in (0,h_0)$. Since ε is arbitrary, we get (5.3).

■

Lemma 5.5 *Let $f \in C(\overline{\Omega})$ in (5.1). Then $\overline{u}(x,0) = \underline{u}(x,0) = f(x)$ for all $x \in \overline{\Omega}$.*

Proof. Take a sequence $\{f_k\}$ in $C^2(\overline{\Omega})$ satisfying

$$\|f_k - f\|_\infty < \frac{1}{k}, \qquad \frac{\partial f_k}{\partial \hat{n}} > 0 \text{ on } \partial\Omega$$

and define u_k^m as in (5.1) with f_k instead of f.

Then, by lemma 5.4, there are constants C_k such that for each $k \in \mathbb{Z}_+$,

$$u_k^m(y,t) - f_k(y) \le C_k t$$

for all $y \in \overline{\Omega}$ and $t \in [0,T)$. Because of proposition 4.2 (iv) and since $\|f_k - f\|_\infty < 1/k$, it follows that

$$|u^m(y,t) - u_k^m(y,t)| \le \frac{1}{k}$$

and thus

$$u^m(y,t) - f(y) \le C_k t + \frac{2}{k} \tag{5.6}$$

for all $y \in \overline{\Omega}$, $t \in [0,T)$ and $k \in \mathbb{Z}_+$.

For any fixed $x \in \overline{\Omega}$, we may now let $m \to \infty$, $y \to x$, $t \to 0$ and finally $k \to \infty$ to conclude by the definition (5.2) of $\overline{u}$ that

$$\overline{u}(x,0) \le f(x).$$

Since $\underline{u}(x,0) \ge f(x)$ by a similar argument and $\underline{u}(x,0) \le \overline{u}(x,0)$ by definition, the result follows.

■

Proof of theorem 5.1. As already mentioned, the idea of the proof is to show that $\overline{u}(x,t)$ (defined by (5.2)) is a subsolution and $\underline{u}(x,t)$ a supersolution of the level set PDE (3.1). It then follows from lemma 5.5 and the comparison result by Sato [20] (theorem 5.2 in this text) that $\overline{u}(x,t) \le \underline{u}(x,t)$ for all $(x,t) \in \overline{\Omega} \times [0,T)$ and thus $u = \overline{u} = \underline{u}$ is a viscosity solution of (3.1). By an argument in chapter 6 of Crandall, Ishii, Lions [7] it also follows that $u^m \to u$ locally uniformly.

We prove that $\overline{u}(x,t)$ is a subsolution and note that the proof that $\underline{u}(x,t)$ is a supersolution is similar.

So, we fix a function $\varphi \in C^2(\overline{\Omega} \times [0,T))$ and assume that $\overline{u} - \varphi$ has a strict maximum at (x_0,t_0). Since only the local behavior of φ is important, we may assume that this maximum is global. Also, since we are only interested in $\partial\varphi/\partial t$, $\nabla\varphi$ and $D^2\varphi$, we may choose φ on the form

$$\varphi(x,t) = \varphi_1(x) + \varphi_2(t)$$

for some functions φ_1, φ_2.

According to definition 5.1, in order to show that $\overline{u}$ is a subsolution, there are two distinct cases.

Case 1: *$x_0 \in \Omega$ and $\nabla\varphi_1(x_0) \neq 0$, or $x_0 \in \partial\Omega$ and $\partial\varphi_1/\partial\hat{n}(x_0) > 0$.*

Fix $\varepsilon > 0$. Then, setting $M \equiv |\nabla\varphi_1(x_0)| g(\mathrm{curv}(\varphi_1(x_0)))$, by proposition 4.3, there is a $\delta_1 > 0$ such that

$$[G_h\varphi_1](x) \le \varphi_1(x) + Mh + \varepsilon h \tag{5.7}$$

for all $x \in B_n(x_0,\delta_1)$ and $h \in (0,\delta_1]$, with $M \equiv |\nabla\varphi_1(x_0)| g(\mathrm{curv}(\varphi_1(x_0)))$.

Since (x_0,t_0) is a strict global maximum point of $\overline{u} - \varphi$, by the definition of $\overline{u}$ there is an $m \in \mathbb{Z}_+$ such that

$$\sup_{B_{n+1}((x_0,t_0),\delta_1)} (u^m - \varphi)(x,t) > \sup_{(\overline{\Omega}\times(0,T))\setminus B_{n+1}((x_0,t_0),\delta_1)} (u^m - \varphi)(x,t)$$

and $h \equiv T/m < \delta_1$. Then, we can choose $(\xi,\tau) \in B_{n+1}((x_0,t_0),\delta_1)$ so that

$$(u^m - \varphi)(\xi,\tau) + \varepsilon h > (u^m - \varphi)(x,t) \tag{5.8}$$

for all $(x,t) \in \overline{\Omega} \times (0,T)$.

Now choose $l \in \mathbb{N}$ so that $lh \le \tau < (l+1)h$. By the definition of u^m, we then have

$$u^m(x,\tau) = [G_{\tau-lh} \circ G_h u^m(\cdot,(l-1)h)](x)$$

for all $x \in \overline{\Omega}$. Also, from (5.8) follows that

$$u^m(x,(l-1)h) \le \varphi_1(x) + \varphi_2((l-1)h) + \varepsilon h + (u^m - \varphi)(\xi,\tau),$$

which, using proposition 4.2 (i), (iii) and (5.7), gives us

$$\begin{aligned} [G_h u^m(\cdot,(l-1)h)](x) &\le [G_h\varphi_1](x) + \varphi_2((l-1)h) + \varepsilon h + (u^m - \varphi)(\xi,\tau) \\ &\le \varphi_1(x) + Mh + 2\varepsilon h + \varphi_2((l-1)h) + (u^m - \varphi)(\xi,\tau) \end{aligned} \tag{5.9}$$

for all $x \in B_n(x,\delta_1)$.

In order to get a similar inequality for each $x \in \overline{\Omega}$, we note that by proposition 4.2 (iv),

$$|u^m(x,t)| \le \|f\| \tag{5.10}$$

for all $(x,t) \in \overline{\Omega} \times [0,T]$ and we choose $C > 0$ such that

$$2\|f\| + \|\varphi_1\| + 2\sup_{[0,T]}|\varphi_2| + |M|T \le C, \tag{5.11}$$

$\delta_2 \in (0,\delta_1)$ and a function $\psi_1 \in C^2(\overline{\Omega})$ with $\psi_1 = \varphi_1$ in $B_n(x_0,\delta_2) \cap \overline{\Omega}$, $\psi_1 \ge \varphi_1$ in $\overline{\Omega}$ and $\psi_1 \ge C$ in $B_n(x_0,\delta_1)^c \cap \overline{\Omega}$. Then, by proposition 4.3, there is a $\delta_3 \in (0,\delta_2]$ such that

$$[G_h\psi_1](x) \le \psi_1(x) + Mh + \varepsilon h \tag{5.12}$$

for all $x \in B_n(x_0,\delta_3)$ and $h \in (0,\delta_3)$. By (5.10) and (5.11), we note that

$$[G_h u^m(\cdot,(l-1)h)](x) - Mh - 2\varepsilon h - \varphi_2((l-1)h) - (u^m - \varphi)(\xi,\tau) \le C,$$

and thus, using also (5.9) and the definition of ψ_1,

$$[G_h u^m(\cdot,(l-1)h)](x) - Mh - 2\varepsilon h - \varphi_2((l-1)h) - (u^m - \varphi)(\xi,\tau) \le \psi_1(x) \tag{5.13}$$

for all $x \in \overline{\Omega}$.

Now, with m chosen so large that $h \le \delta_3$, we apply $G_{\tau-lh}$ to both sides of (5.13) and use (5.12) to get

$$\begin{aligned} u^m(x,\tau) &= [G_{\tau-lh} \circ G_h u^m(\cdot,(l-1)h)](x) \\ &\le \psi_1(x) + M(\tau - lh) + \varepsilon(\tau - lh) + Mh + 2\varepsilon h \\ &\quad + \varphi_2((l-1)h) + (u^m - \varphi)(\xi,\tau) \\ &\le \varphi_1(x) + M(\tau - (l-1)h) + 2\varepsilon(\tau - (l-1)h) \\ &\quad + \varphi_2((l-1)h) + (u^m - \varphi)(\xi,\tau) \end{aligned}$$

for all $x \in B_n(x_0, \delta_3)$, since $h \leq \tau - (l-1)h$. Specifically for $x = \xi$, we get

$$u^m(\xi,\tau) \leq \varphi_1(\xi) + M(\tau-(l-1)h) + 2\varepsilon(\tau-(l-1)h) + \varphi_2((l-1)h) + (u^m-\varphi)(\xi,\tau),$$

that is, since $\varphi(\xi,\tau) = \varphi_1(\xi) + \varphi(\tau)$,

$$\varphi_2(\tau) - \varphi_2((l-1)h) \leq (M+2\varepsilon)(\tau-(l-1)h).$$

Letting $\varepsilon \to 0$, so that $m \to \infty$, $h \to 0$ and $(\xi,\tau) \to (x_0,t_0)$, we get

$$\varphi_t'(x_0,t_0) \leq M = |\nabla\varphi(x_0,t_0)|g(\operatorname{curv}(\varphi(x_0,t_0))),$$

which is what we want according to definition 5.1.

Case 2: $x_0 \in \Omega$, $\nabla\varphi_1(x_0) = 0$ *and* $D^2\varphi_1(x_0) = 0$.

We need to prove that $\varphi_t'(x_0,t_0) \leq 0$.

Fix $\varepsilon > 0$ and choose $C_\varepsilon > 0$ so that

$$\varphi_1(x) \leq \varphi_1(x_0) + \varepsilon|x-x_0|^2 + C_\varepsilon|x-x_0|^4$$

for all $x \in \overline{\Omega}$. Also choose a small $\gamma > 0$. Then there is a $\delta = \delta(\gamma) > 0$ such that for $\xi \in B_n(x_0,\delta) \subset \Omega$,

$$\varphi_1(x) \leq \varphi_1(\xi) + \gamma\varepsilon + \varepsilon|x-\xi|^2 + C_\varepsilon|x-\xi|^4 \tag{5.14}$$

for all $x \in \overline{\Omega}$. We also assume $\delta \to 0$ as $\gamma \to 0$.

As before, since (x_0,t_0) is a strict global maximum of $\overline{u} - \varphi$, there is an $m = m(\gamma) \in \mathbb{N}$ such that

$$\sup_{B_{n+1}((x_0,t_0),\delta)} (u^m - \varphi) > \sup_{(\overline{\Omega}\times(0,T))\setminus B_{n+1}((x_0,t_0),\delta)} (u^m - \varphi)(x,t)$$

and with $m(\gamma) \to \infty$ as $\gamma \to 0$.

Now, set $h = T/m$ and choose $(\xi,\tau) \in B_{n+1}((x_0,t_0),\delta)$ so that

$$(u^m-\varphi)(\xi,\tau) + \gamma\varepsilon > (u^m-\varphi)(x,t) \tag{5.15}$$

for all $(x,t) \in \overline{\Omega}\times(0,T)$. Also, choose $k,l \in \mathbb{N}$ so that

$$kh \leq \tau - \gamma < (k+1)h \quad \text{and} \quad lh \leq \tau < (l+1)h.$$

The definition of u^m then gives us that for all $x \in \overline{\Omega}$,

$$u^m(x,\tau) = [G_{\tau-lh} \circ (G_h)^{l-k+1} u^m(\cdot,(k-1)h)](x).$$

Choosing m larger if necessary, by lemma 5.3 there is a $\delta_2 > 0$ such that

$$[G_\eta(|\cdot-\xi|^2)](x) \leq |x-\xi|^2 + C\eta$$

for some $C > 0$ and all $\eta \in (0,h]$, $x \in B_n(\xi,\delta_2)$.

Thus, applying G_h to both sides of (5.14) and using proposition 4.2 (i), (ii) and (iii), we get

$$\begin{aligned}[G_h\varphi_1](x) &\leq \varphi_1(\xi) + \gamma\varepsilon + [G_h(\varepsilon|\cdot-\xi|^2 + C_\varepsilon|\cdot-\xi|^4)](x)\\ &= \varphi_1(\xi) + \gamma\varepsilon + \varepsilon[G_h(|\cdot-\xi|^2)](x) + C_\varepsilon([G_h(|\cdot-\xi|^2)](x))^2\\ &\leq \varphi_1(\xi) + \gamma\varepsilon + \varepsilon(|x-\xi|^2 + Ch) + C_\varepsilon(|x-\xi|^2+Ch)^2,\end{aligned}$$

for $x \in B_n(\xi,\delta_2)$. Since $\|G_h\varphi_1\|_\infty \leq \|\varphi_1\|_\infty$, setting $C_2 = 2\|\varphi_1\|/\delta_2^4 < \infty$, we then see that

$$[G_h\varphi_1](x) \leq \varphi_1(\xi) + \gamma\varepsilon + \varepsilon(|x-\xi|^2+Ch) + C_\varepsilon(|x-\xi|^2+Ch)^2 + C_2|x-\xi|^4,$$

for all $x \in \overline{\Omega}$. We may then apply G_h again to both sides, which gives us

$$\begin{aligned}[G_{h_1}\circ G_{h_2}\varphi_1](x) &\leq \varphi_1(\xi) + \gamma\varepsilon + \varepsilon([G_{h_1}(|\cdot-\xi|^2)](x) + Ch_2)\\ &\quad + C_\varepsilon([G_{h_1}(|\cdot-\xi|^2)](x) + Ch_2)^2 + C_2([G_{h_1}(|\cdot-\xi|^2)](x))^2\\ &\leq \varphi_1(\xi) + \gamma\varepsilon + \varepsilon(|x-\xi|^2 + C(h_1+h_2))\\ &\quad + C_\varepsilon(|x-\xi|^2 + C(h_1+h_2))^2 + C_2(|x-\xi|^2+Ch_1)^2,\end{aligned}$$

for any $h_1, h_2 \in (0,h]$ and $x \in B_n(\xi,\delta_2)$.

Therefore, with $x = \xi$, we may conclude

$$\begin{aligned}[G_{\tau-lh}\circ(G_h)^{l-k+1}\varphi_1](\xi) &\leq \varphi_1(\xi) + \gamma\varepsilon + \varepsilon C(\tau-(k-1)h) + C_\varepsilon C^2(\tau-(k-1)h)^2\\ &\quad + C_2C^2(\tau-(k-1)h)^2,\end{aligned}$$

since $\tau - kh < \tau - (k-1)h$. And since from (5.15),

$$u^m(x,(k-1)h) \leq \varphi(x,(k-1)h) + (u^m-\varphi)(\xi,\tau) + \gamma\varepsilon,$$

application of $G_{\tau-lh}\circ(G_h)^{l-k+1}$ on both sides gives

$$\begin{aligned}u^m(\xi,\tau) &\leq \varphi_1(\xi) + \varphi_2((k-1)h) + 2\gamma\varepsilon + \varepsilon C(\tau-(k-1)h)\\ &\quad + C^2(C_\varepsilon + C_2)(\tau-(k-1)h)^2 + (u^m-\varphi)(\xi,\tau).\end{aligned}$$

Moving the terms around, noting that $\gamma < \tau - (k-1)h$ and finally dividing by $\tau-(k-1)h$, we get

$$\frac{\varphi_2(\tau)-\varphi_2((k-1)h)}{\tau-(k-1)h} \leq 2\varepsilon + \varepsilon C + C^2(C_\varepsilon+C_2)(\tau-(k-1)h).$$

Finally, letting $\gamma \to 0$, so that $\tau \to t_0$ and $\tau-(k-1)h \to 0$, we get $\varphi_2'(t_0) \leq 2\varepsilon + \varepsilon C$, i.e $\varphi_t'(x_0,t_0) \leq 0$, since ε is arbitrary.

■

Finally, we show how to get around the assumption (3.4) that g has bounded derivative from above and below. The next theorem shows that if g does not have bounded derivative it is enough to approximate g with functions g_ν, such that $g_\nu \to g$ uniformly as $\nu \to 0$. We remark that if g has unbounded derivative, the derivative of g_ν must obviously tend to infinity when $\nu \to 0$, which means that in order to fulfill the inequalities (1.4) restricting the choice of a_i and b_i, the radius r_2 of the support of the convolution kernel ρ_2 must also tend to infinity. In order to get a converging scheme, the choice of ν and the time-step h must be such that $r_2\sqrt{h}$, which is the radius of the support of the scaled kernel $\rho_2^{\sqrt{h}}$, is small. The exact choice of ν depends on the exact definition of the approximations g_ν and the choice of the kernel ρ_2.

Theorem 5.6 *Let $g \in C(\mathbb{R})$ have the properties (3.3) and let $\{g_\nu\}_{\nu>0}$ be a family of functions in $C(\mathbb{R})$ that fulfill the assumptions (3.3) and (3.4), such that $g_\nu(\kappa) \to g(\kappa)$ as $\nu \to 0$, uniformly for $\kappa \in \mathbb{R}$. Define*

$$F_\nu(N_1, N_2) = \frac{1}{\sqrt{h}}\frac{b_2(\nu)N_1 - b_1(\nu)N_2}{d(\nu)} - g_\nu\left(\frac{1}{\sqrt{h}}\frac{a_2(\nu)N_1 - a_1(\nu)N_2}{d(\nu)}\right),$$
$$[G_h^\nu\varphi](x) = \sup\{\lambda \in \mathbb{R} \mid F_\nu(N_1(\lambda), N_2(\lambda)) \geq 0\},$$

with N_i, a_i, b_i and d as in (1.3). Finally, for $f \in C(\overline{\Omega})$ define

$$u_\nu^m(x,t) = [G_{t-lh}^\nu \circ (G_h^\nu)^l f](x),$$

with $h \equiv T/m$ and $lh \leq t < (l+1)h$.

Then there is a sequence $\{\nu_m\}_{m=1}^\infty$ with $\nu_m \to 0$ as $m \to \infty$, such that

$$u_{\nu_m}^m(x,t) \to u(x,t) \text{ as } m \to \infty,$$

locally uniformly for $(x,t) \in \overline{\Omega} \times (0,T)$, where $u(x,t)$ is the unique viscosity solution of the level set PDE (3.1).

Proof. It follows from remarks 6.3 and 6.4 in [7], that if v_n is a subsolution of a proper equation $F_n(x, v, \nabla v, D^2 v) = 0$ in a set $\mathcal{O}$ for $n = 1, 2, \ldots$, and we define

$$\overline{v}(x) = \limsup_{n\to\infty}{}^* v_n(x) = \lim_{j\to\infty} \sup\{v_n(y) \mid n \geq j,\ y \in \mathcal{O},\ |y - x| < j^{-1}\}, \tag{5.16}$$

then $\overline{v}$ is a subsolution of the equation $G = 0$, where

$$G(x, v, p, X) = \liminf_{n\to\infty}{}_* F_n(x, v, p, X)$$
$$= \lim_{j\to\infty} \inf\{F_n(y, w, q, Y) \mid n \geq j,\ (y, w, q, Y) \in W,\ \|(x, v, p, X) - (y, w, q, Y)\| \leq j^{-1}\},$$

with W dense in $\mathcal{O} \times \mathbb{R} \times \mathbb{R}^n \times \mathcal{S}(n)$ (cf. section 2.4). Furthermore, $v_n \to \overline{v}$ locally uniformly. The equivalent conclusion holds for supersolutions (with $\liminf_*$ and $\limsup^*$ interchanged). There may also be boundary conditions included in F_n and G (see also section 7.A in [7]).

In our case, this means that solutions u_{ν_m} to the equations $u_t - |\nabla u| g_{\nu_m}(\text{curv}(u)) = 0$, with Neumann boundary conditions, tend locally uniformly to the solution u of the equation $u_t - |\nabla u| g(\text{curv}(u)) = 0$ with Neumann boundary conditions. This is because

$$\liminf_{m\to\infty}{}_* |p| g_{\nu_m}(\text{curv}(p, X)) = \limsup_{m\to\infty}{}^* |p| g_{\nu_m}(\text{curv}(p, X)) = |p| g(\text{curv}(p, X)),$$

since $g_{\nu_m} \to g$ uniformly. Here, $\text{curv}(p, X)$ is defined as $\text{curv}(u)$ (see (3.2)) with $p = \nabla u$ and $X = D^2 u$, but this notation is chosen to emphasize that the *-limits are with respect to p and X, and not u.

Furthermore, theorem 5.1 shows that for each fixed ν, the approximations u_ν^m tend to u_ν locally uniformly. Therefore, we can find a sequence $\{\nu_m\}_{m=1}^\infty$ such that for each compact $K \subset \overline{\Omega} \times [0,T)$, it holds that for each $\varepsilon > 0$, there is an $M \in \mathbb{Z}_+$ such that

$$|u_{\nu_m}^m(x,t) - u(x,t)| \leq |u_{\nu_m}^m(x,t) - u_{\nu_m}(x,t)| + |u_{\nu_m}(x,t) - u(x,t)| < \varepsilon/2 + \varepsilon/2 = \varepsilon$$

for all $m \geq M$ and $(x,t) \in K$. That is, $u_{\nu_m}^m \to u$ locally uniformly as $m \to \infty$.

■

6. Examples

We conclude with two examples showing the output of the algorithm. The images have been computed using the code developed by Ricards Grzibovskis for the case without boundary conditions. The code has then been adapted to include boundary conditions by him and Alexei Heintz, and used by them to generate time-series of two different evolutions shown below. The images are generated using the VRweb software [1]. Because of lack of time to adapt the code, the examples show only regular mean curvature evolutions, with $g(\kappa) = \kappa$.

The first example is shown in figure 6.1 and shows the mean curvature flow of a cylinder, which has been placed slightly off center inside a sphere. The intersections with the sphere are to the left and right. The time points are chosen to produce interesting images, meaning that the time intervals between subsequent images are not equal. However, time flows from top to bottom. We see that, due to the boundary conditions, the cylinder becomes slightly bent and thinner in the middle than at the edges. The thin bottleneck then rapidly becomes thinner and eventually pinches off to produce two distinct surfaces, which both quickly diminish and vanish at the boundary. This evolution is quite different from the evolution of a closed cylinder without boundary conditions, since in that case the cylinder would remain a single surface until it vanishes to a point.

The second example (see figure 6.2) shows an ellipsoid inside a sphere. This time the ellipsoid has been shifted from the center of the sphere both in x and y directions to produce an asymmetric evolution. Once again, the intersections with the sphere are seen to the left and right and the evolution proceeds downwards from the top image. We see that the surface adapts to the right-angle boundary condition at both edges and that the thin neck on the left, where the surface intersects the boundary, becomes gradually thinner until, finally, it releases from the boundary. The evolution then continues with the surface quickly shrinking from the left to the right until it disappears at the right boundary (not shown).

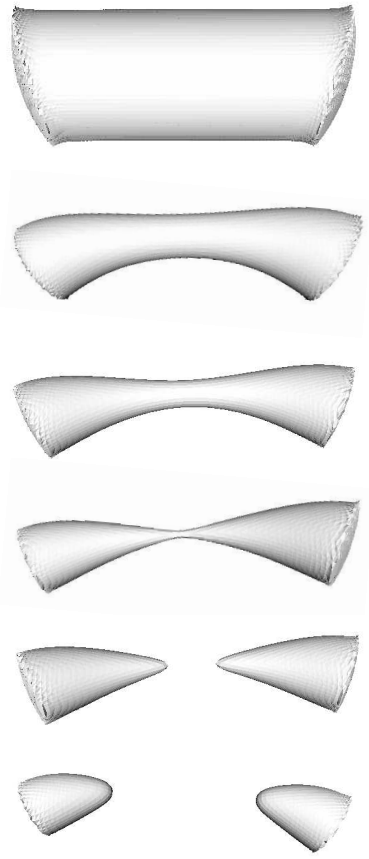

Figure 6.1: First example (cylinder inside sphere).

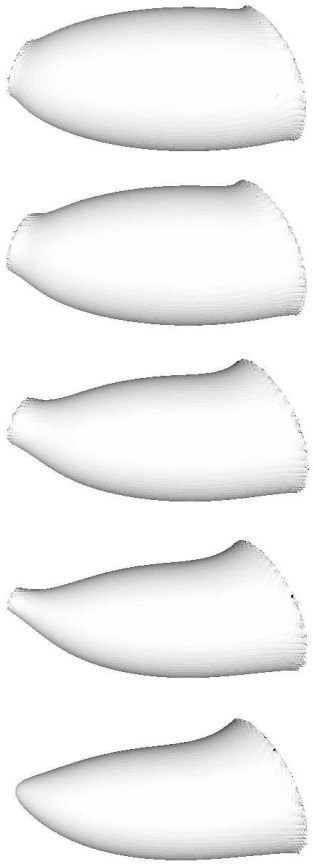

Figure 6.2: Second example (ellipsoid inside sphere).

References

[1] VRweb VRML browser, `http://www2.iicm.edu/vrweb`.

[2] L. Alvarez, F. Guichard, P.-L. Lions, and J.-M. Morel. Axioms and fundamental equations of image processing. *Arch. Rational Mech. Anal.*, 123(3):199–257, 1993.

[3] G. Barles and C. Georgelin. A simple proof of convergence for an approximation scheme for computing motions by mean curvature. *SIAM J. Numer. Anal.*, 32(2):484–500, 1995.

[4] J. Bence, B. Merriman, and S. Osher. Diffusion generated motion by mean curvature. In *Computational Crystal Growers Workshop*, pages 73–83. American Mathematical Society, 1992.

[5] K. A. Brakke. *The Motion of a Surface by its Mean Curvature*. Princeton University Press, Princeton, NJ, 1978.

[6] Y.-G. Chen, Y. Giga, and S. Goto. Uniqueness and existence of viscosity solutions of generalized mean curvature flow equations. *J. Differential Geom.*, 33:749–786, 1991.

[7] M. G. Crandall, H. Ishii, and P.-L. Lions. User's guide to viscosity solutions of second order partial differential equations. *Bull. Amer. Math. Soc.*, 27(1):1–67, 1992.

[8] K. Ecker. *Regularity Theory for Mean Curvature Flow*. Birkhäuser Verlag, 2004.

[9] L. C. Evans. Convergence of an algorithm for mean curvature motion. *Indiana Univ. Math. J.*, 42:533–556, 1993.

[10] L. C. Evans and J. Spruck. Motion of level sets by mean curvature, I. *J. Differential Geom.*, 33:635–681, 1991.

[11] Y. Giga and M.-H. Sato. Neumann problem for singular degenerate parabolic equations. *Differential and Integral Equations*, 6(6):1217–1230, 1993.

[12] R. Grzibovskis and A. Heintz. A convolution-thresholding approximation of generalized curvature flows. to appear in SIAM J. Numer. Anal., 2004.

[13] H. Ishii. A generalization of the Bence, Merriman and Osher algorithm for motion by mean curvature. In *Curvature flows and related topics (Levico, 1994)*, volume 5 of *GAKUTO Internat. Ser. Math. Sci. Appl.*, pages 111–127. Gakkōtosho, Tokyo, 1995.

[14] H. Ishii and K. Ishii. An approximation scheme for motion by mean curvature with right-angle boundary condition. *SIAM J. Math. Anal.*, 33(2):369–389, 2001.

[15] H. Ishii, G. E. Pires, and P. E. Souganidis. Threshold dynamics type approximation schemes for propagating fronts. *J. Math. Soc. Japan*, 51(2):267–308, 1999.

[16] H. Ishii and P. Souganidis. Generalized motion of noncompact hypersurfaces with velocity having arbitrary growth on the curvature tensor. *Tohoku Math. J.*, 47:227–250, 1995.

[17] S. Osher and J. A. Sethian. Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobiformulations. *J. Comput. Phys.*, 79:12–49, 1988.

[18] J. Rubinstein, P. Sternberg, and J. Keller. Fast reaction, slow diffusion and curve shortening. *SIAM J. Appl. Math.*, 49:116–133, 1989.

[19] S. J. Ruuth. Efficient algorithms for diffusion-generated motion by mean curvature. *J. Comput. Phys.*, 144(2):603–625, 1998.

[20] M.-H. Sato. Interface evolution with Neumann boundary condition. *Adv. Math. Sci. Appl.*, 4(1):249–264, 1994.