M-Matrices in Meshless Finite Difference Methods

Benjamin Seibold
Department of Mathematics
University of Kaiserslautern, Germany
Email: seibold@mathematik.uni-kl.de

Vom Fachbereich Mathematik
der Technischen Universität Kaiserslautern
zur Verleihung des akademischen Grades
Doktor der Naturwissenschaften
(Doctor rerum naturalium, Dr. rer. nat.)
genehmigte Dissertation.

1. Gutachter	Prof. Dr. Axel Klar
2. Gutachter	Prof. Dr. Michael Junk
Vollzug der Promotion	28. September 2006

Acknowledgments

I would like to express my gratitude to Prof. Helmut Neunzert (TU Kaiserslautern) for finding this interesting subject and for giving me the opportunity to study and research in the graduate college *Mathematik und Praxis*. I gratefully acknowledge his advice, his help and his criticism all throughout my academic studies in Kaiserslautern.

I am grateful to Prof. Axel Klar (TU Kaiserslautern) for being advisor for my work in the last two years. I would like to thank him for his trust in my working, his support, and especially for the friendly and helpful atmosphere.

I would like to thank Dr. Jörg Kuhnert (Fraunhofer ITWM) for his help and comments regarding the application of my ideas. He gave me the opportunity to test my methods on real data and his interest in the results was a strong motivation for investigating the approach deeper. I also would like to thank Dr. Sudarshan Tiwari (Fraunhofer ITWM) for his help and support with various technical aspects of the FPM code.

Great acknowledgments go to Prof. Sven Krumke (TU Kaiserslautern) for his help on various aspects in optimization, to Dr. Klaus Stüben and his colleagues (Fraunhofer SCAI) for giving me the opportunity to work with the SAMG code, and to Christian Mense (TU Berlin) for his help in testing multigrid solvers for systems obtained by approaches developed in this thesis.

Many thanks go my colleagues from the Technomathematics group, firstly for helpful comments and fruitful discussions on my research, and secondly for the friendly atmosphere in many non-scientific activities. The same applies for my co-PhD-students in the graduate college.

The financial support of the DFG Graduiertenkolleg *Mathematik und Praxis* is gratefully acknowledged.

Finally, I would like to thank my parents and my friends, especially my Ultimate Frisbee team, for their support and help.

Contents

1 Introduction **1**

2 Meshless Particle Methods for Flow Problems **5**

2.1 Equations of Fluid Mechanics 5

2.1.1 Conservation of Mass 6

2.1.2 Balance of Momentum 7

2.1.3 Conservation of Energy 8

2.2 Numerical Methods for Flow Problems 9

2.2.1 Numerical Methods for Incompressible Viscous Flows 10

2.2.2 Meshless Methods for Elliptic Problems 12

2.3 Smoothed Particle Hydrodynamics 14

2.3.1 Smoothing and Particle Approximation 14

2.3.2 Numerical Method 16

2.3.3 Application to Hydrodynamic Equations 16

2.3.4 Features of SPH 18

2.3.5 From SPH to FPM 19

2.4 FPM for Incompressible Navier-Stokes Equations 20

2.4.1 Boundary Conditions 21

2.4.2 Examples of Application 23

2.5 Particle Management 25

2.5.1 Adding Particles 25

2.5.2 Removing Particles 26

2.6 Particle Management for Burgers' Equation 26

2.6.1 Solutions of Burgers' Equation 27

2.6.2 FPM for Inviscid Burgers' Equation 28

2.6.3 FPM for Viscous Burgers' Equation 33

3 Point Clouds **37**

3.1 Point Cloud Geometry 38

3.2 Data Management 42

3.2.1 Generating Lists 42

3.2.2 Finding Neighboring Points 43

3.2.3 Removing Close Points . . . 43
3.2.4 Filling Holes . . . 43
3.2.5 Varying Point Density . . . 44
3.3 Point Cloud Generation . . . 44
3.4 Point Cloud Manipulation . . . 45
3.5 Neighborhood Relations . . . 45
3.5.1 Geometric Neighborhood Concepts . . . 46

4 Meshless Finite Differences for the Poisson Equation 51
4.1 Matrix Generation . . . 51
4.1.1 Matrix Graph and Connectivity . . . 52
4.2 Consistent Derivative Approximation . . . 54
4.2.1 Positive Stencils . . . 56
4.2.2 Higher Order Approximations . . . 56
4.2.3 Special Configurations . . . 58
4.3 General Second Order Elliptic Problems . . . 60

5 Least Squares Methods 63
5.1 Local Least Squares Method . . . 63
5.1.1 Approximating Local Least Squares . . . 64
5.1.2 Interpolating Local Least Squares . . . 66
5.1.3 Minimization Formulation . . . 67
5.1.4 Approximating and Interpolating Approach in 1d . . . 69
5.2 Moving Least Squares Method . . . 71
5.2.1 Distance Weight Function . . . 71
5.2.2 Approximation in Polynomial Basis . . . 73
5.2.3 Shepard's Method . . . 74
5.2.4 Derivatives by the Moving Least Squares Method . . . 75
5.3 Derivatives by Approximating Moving Least Squares . . . 77
5.3.1 Consistency of First and Second Derivative . . . 77
5.3.2 Special Cases in One Space Dimension . . . 79
5.3.3 Special Case: First Derivatives on Symmetric Grid . . . 81
5.3.4 Special Case: Gaussian Weight Function on Arbitrary Grid . . . 83
5.3.5 Special Case: Gaussian Weight Function on Symmetric Grid . . . 85
5.3.6 Inconsistency for Insufficient Polynomial Degree . . . 91
5.3.7 Special Case: One Sided Derivative . . . 94
5.4 Derivatives by Interpolating Moving Least Squares . . . 96
5.4.1 Consistency for Varying Polynomial Degree . . . 97
5.4.2 Special Case: Symmetric Grid . . . 100
5.4.3 Special Case: One Sided Derivative . . . 103
5.4.4 IMLS in Matrix Notation . . . 104
5.5 Comparison of the Approaches . . . 105
5.5.1 Comparison for Regular Symmetric Grid . . . 106
5.5.2 Comparison for Poisson Test Problem . . . 107
5.5.3 Numerical Effort of Stencil Computations . . . 112
5.5.4 Conclusions . . . 114

6 The Minimal Positive Stencil Method 117
6.1 M-Matrices . . . 118
6.1.1 Sufficient Conditions for M-Matrix Structure . . . 119
6.1.2 Consequences of M-Matrix Structure . . . 121
6.2 Approaches to Obtain Positive Stencils . . . 122
6.3 Linear Minimization Approach . . . 123
6.3.1 Comparison of Quadratic and Linear Minimization . . . 126
6.3.2 Geometric Interpretation of Minimal Positive Stencils . . . 128
6.3.3 Minimal Positive Stencils for Specific Geometries . . . 129
6.3.4 Discontinuous Dependence on Geometry . . . 131
6.4 Conditions for the Existence of Positive Stencils . . . 132
6.4.1 A Necessary Criterion for Positive Stencils . . . 134
6.4.2 A Sufficient Criterion for Positive Stencils . . . 135
6.4.3 Condition on Point Cloud Geometry . . . 140
6.5 Neumann Boundary Points . . . 141
6.5.1 Conditions for Positive Neumann Stencils . . . 143
6.5.2 Linear Minimization for Neumann Points . . . 144
6.5.3 Geometric Interpretation of Neumann Stencils . . . 145
6.6 Minimal Stencils and Matrix Connectivity . . . 146
6.7 Optimization Routines . . . 148
6.7.1 Effort of Simplex in Comparison to Least Squares . . . 150
6.7.2 Improvement of the Simplex Method . . . 151
6.7.3 Forcing a Specific Point into the Stencil . . . 152
6.8 Implementation of the MPS Method . . . 153
6.9 Neighborhood Concepts Revisited . . . 154
6.10 Generalizations of the MPS Approach . . . 156
6.10.1 Hybrid Approximation Methods . . . 156
6.10.2 MPS Cells . . . 156
6.11 General Second Order Elliptic Problems . . . 158

7 Linear Solvers 159
7.1 Iterative Solvers . . . 160
7.1.1 Examples and Convergence for M-Matrices . . . 161
7.2 Krylov Subspace Methods . . . 162
7.3 Preconditioning . . . 163
7.4 Multigrid . . . 164
7.4.1 Basic Multigrid . . . 164
7.4.2 Multigrid Versions . . . 166
7.4.3 Algebraic Multigrid . . . 167
7.4.4 Convergence Results for AMG . . . 168
7.4.5 FPM and Multigrid for Small Time Steps . . . 170

8 Numerical Experiments **173**
8.1 Test Problem in 2d – Compare MPS and LSQ 173
8.1.1 Matrix Structure 175
8.1.2 Inverse Matrix Structure 177
8.1.3 Approximation Error 179
8.2 Test Problem in 3d – Compare Solvers 181
8.3 Investigate AMLI Solver 184
8.4 MPS Method in FPM Simulation 185

9 Conclusions and Outlook **191**

Bibliography **201**

Chapter 1

Introduction

In numerical simulations with complex time-dependent geometries a significant amount of computation time can be devoted to the generation and management of meshes. Meshless methods provide an alternative, which can be advantageous in particularly the cases in which meshing and remeshing would be computationally elaborate. Meshless methods typically only have points as computational instances. No topological information like edges or surfaces is given or considered.

In this thesis we consider problems of fluid flow. Numerical methods of interest are meshless Lagrangian particle methods. These are methods in which the data points move with the fluid flow. This approach is in some sense natural for fluid flows, since the governing equations themselves can be derived by considering small fluid particles which move with the flow. Flow equations typically posses an advection term, which is present if the flow is viewed at by a stationary observer. Lagrangian particle methods take the point of view of a moving particle, consequently the nonlinear term is not present.

The price to pay is the loss of control over the particles. Their movement is given by the flow. In order to prevent particles from moving into configurations problematic for a numerical approximation, a continuous particle management has to be done.

A classical and famous Lagrangian particle method for flow equations is Smoothed Particle Hydrodynamics (SPH). Originally developed for astrophysical simulations, it has been generalized to many other types of flows and applications. One of these generalizations is the Finite Pointset Method (FPM). The governing equations are approximated in their differential (strong) form using meshless finite difference approximations.

A main focus of this thesis lies on the application of the Finite Pointset Method to the incompressible Navier-Stokes equations. The incompressibility is taken care of by a projection method. This requires the solution of Poisson problems in each time step. Since the particle positions are given by the flow, the data is given on a scattered point cloud. The

Finite Pointset Method originally uses the classical approach of a least squares method to discretize the arising Poisson equations.

In this thesis, we consider various aspects of this approach. Chapter 2 provides an overview over the considered flow equations and the commonly used meshless methods to numerically approximate these equations. The method of SPH is outlined, and how it generalizes to the FPM. The aspect of particle management is considered for Burgers' equation, which is the simplest model equation for flow problems with nonlinear advection.

Since in Lagrangian methods the particles move with the flow, in each time step one is confronted with a point cloud. The cloud describes the computational domain with its boundaries. Particle management has to be performed on the point cloud, and Poisson problems have to be solved. Chapter 3 provides the relevant ingredients for these aspects. These are geometric properties, organization of the data structures and concepts of neighborhood.

In applications of the FPM to viscous, incompressible flows, multiple Poisson-type problems have to be solved in each time step. This consists of two parts. Firstly, the equation has to be discretized, i.e. the Poisson equation is approximated by a finite dimensional linear system. Secondly, the derived linear system has to be solved. Solving these Poisson problems can easily take up two thirds of the total computational effort. Hence, fast methods for solving the Poisson problems are of interest.

Chapter 4 deals with the discretization of the Poisson equation by meshless finite differences. These are based on Taylor expansion. We consider aspects, which will be of importance in the following chapters, such as the connection between the finite difference approximation and the graph of the system matrices, the aspect of positive stencils, as well as higher order approximations.

The classical, commonly used approach to approximate derivatives in a meshless context are least squares methods. Chapter 5 is devoted to various least squares approaches and their similarities and differences. Least squares approaches for derivative approximation can be divided into local and moving approaches. For each such case, one must distinguish between approximating and interpolation approaches. All approaches are firstly considered in specific geometric configurations, and secondly compared for a Poisson model problem. One major focus is the aspect of positive stencils.

A stencil is the collection of coefficients which, used in a linear combination of function values of neighboring points, approximate a derivative at a considered central point. A stencil is positive if the neighboring coefficients are all of the same sign. Under appropriate additional assumptions, the resulting finite difference matrix is an M-matrix, if all stencils are positive. In Chapter 6 we present a new approach, which enforces positivity of stencils. Compared to least squares approaches, the situation becomes very different. The obtained stencils turn out to be minimal, i.e. no more points than required for the finite difference

approximation are actually used. Consequently, we call this new approach *Minimal Positive Stencil* (MPS) method. A major drawback will turn out to be a discontinuous dependence on the point positions. We show that under acceptable conditions on the point cloud, the MPS method always succeeds in finding positive stencils (also for Neumann boundary points). The obtained system matrices are M-matrices.

The M-matrix structure is advantageous for various linear solvers, as outlined in Chapter 7. We consider multigrid solvers for the meshless Poisson problem. In particular, we focus on Algebraic Multigrid (AMG) approaches.

The considered finite difference approaches are based on Taylor expansion. This gives a large flexibility, but it is not a very powerful tool for obtaining results about stability and convergence. In Chapter 8 we perform various numerical experiments. We compare the MPS approximation with different types of least squares approximations with respect to accuracy and properties of the system matrices. For test problems in two and three space dimensions, different discretization approaches and different linear solvers are compared. In particular, we consider SAMG, a robust black box AMG solver, and AMLI in the form of a simple version of an algebraic twogrid solver. We compare the SAMG methods to a Krylov subspace solver. Also, the computational effort for solving with matrices arising from different discretization approaches will be considered.

Remark 1.1. In this thesis, we use the term *meshless*. It stands identical to the term *meshfree* in other publications.

Chapter 2

Meshless Particle Methods for Flow Problems

In this chapter, we introduce the equations of fluid dynamics which will be of interest in this thesis. We provide an overview over meshless methods for these equations. Of particular interest are the incompressible Navier-Stokes equations. In this context, meshless methods for elliptic problems are important, which we give an overview of methods for. The main focus of this thesis is the Finite Pointset Method (FPM). It is based on Smoothed Particle Hydrodynamics (SPH), a method originating in 1977 for the simulation of astrophysical systems. We outline the SPH method, and how it leads to the FPM. We present the FPM for incompressible viscous flows. Particle management is an important aspect in particle methods. We present the fundamental ideas of particle management for FPM and investigate particle management for the one dimensional Burgers' equation.

2.1 Equations of Fluid Mechanics

The equations of fluid mechanics are derived from physical balance and conservation principles (which are given in integral form) by assuming the fluid to be a continuum and the quantities being sufficiently smooth.

Consider a domain $\Omega \subset \mathbb{R}^d$ filled with a fluid. Let $\mathbf{u}(\mathbf{x}, t)$ denote the velocity of a fluid particle which is moving through $\mathbf{x}$ at time t. Further assume the existence of a mass density $\rho(\mathbf{x}, t)$, such that the total mass in a volume $V \subset \Omega$ is given by

$$m(V, t) = \int_V \rho(\mathbf{x}, t) d\mathbf{x} \ . \tag{2.1}$$

We derive the equations of fluid motion in a Lagrangian point of view. Basic ingredient is the

Theorem 2.1 (Reynolds Transport Theorem)**.** *Let $f(\mathbf{x},t)$ be a smooth quantity on the domain Ω. Let further $V(t)$ be a* control volume*, i.e. a volume which moves with the flow. Then*

$$\frac{d}{dt}\int_{V(t)} f\,\mathrm{d}\mathbf{x} = \int_{V(t)} \frac{\partial f}{\partial t} + \nabla\cdot(f\mathbf{u})\,\mathrm{d}\mathbf{x}\ . \tag{2.2}$$

Proof. The proof is given in [CM00][pp. 9]. □

For the Lagrangian form of the equations we consider the following

Definition 2.1. Let $\mathbf{u}(\mathbf{x},t)$ be a velocity field and $f(\mathbf{x},t)$ a smooth quantity on Ω. Then the *material derivative* or *Lagrangian derivative* is defined by

$$\frac{Df}{Dt} = \frac{\partial f}{\partial t} + \mathbf{u}\cdot\nabla f\ . \tag{2.3}$$

Remark 2.1. Note that the material derivative can also be applied to vector quantities, in particular to the velocity field $\mathbf{u}$ itself, in which case the nonlinear term $\mathbf{u}\cdot\nabla\mathbf{u}$ can be interpreted as the Jacobian matrix of $\mathbf{u}$ multiplied from the left by the row vector $\mathbf{u}$.

2.1.1 Conservation of Mass

Applying Theorem (2.1) to the mass density ρ yields

$$\frac{d}{dt}\int_{V(t)} \rho\,\mathrm{d}\mathbf{x} = \int_{V(t)} \frac{\partial \rho}{\partial t} + \nabla\cdot(\rho\mathbf{u})\,\mathrm{d}\mathbf{x}\ . \tag{2.4}$$

On the other hand, due to the principle of *conservation of mass*, the mass in a control volume must be conserved, thus the left hand side is zero. Since equation (2.4) holds for any control volume, one obtains the *continuity equation* in differential form

$$\frac{\partial \rho}{\partial t} + \nabla\cdot(\rho\mathbf{u}) = 0\ . \tag{2.5}$$

This leads to the

Theorem 2.2 (Transport Theorem)**.** *Let $f(\mathbf{x},t)$ be a smooth quantity on the domain Ω, and let ρ be a density satisfying the continuity equation. Then for every control volume $V(t)$ we have*

$$\frac{d}{dt}\int_{V(t)} \rho f\,\mathrm{d}\mathbf{x} = \int_{V(t)} \rho\frac{Df}{Dt}\,\mathrm{d}\mathbf{x}\ . \tag{2.6}$$

Proof. Due to (2.1) and the chain rule the left hand side equals

$$\frac{d}{dt}\int_{V(t)} \rho f\,\mathrm{d}\mathbf{x} = \int_{V(t)} (\rho_t + \nabla\cdot(\rho\mathbf{u}))f + \rho(f_t + \mathbf{u}\cdot\nabla f)\,\mathrm{d}\mathbf{x}\ . \tag{2.7}$$

Due to (2.5), the first summand vanishes, yielding the above right hand side. □

Equation (2.5) can also be written in Lagrangian form

$$\frac{D\rho}{Dt} = -\rho \nabla \cdot \mathbf{u} \ . \tag{2.8}$$

2.1.2 Balance of Momentum

According to Newton's second law of motion, the change of momentum of a lump of fluid in a control volume $V(t)$ equals the forces acting on it. The physical forces can be separated into *body forces*, such as gravity, and *surface forces*, which act through the surface of the control volume, such as pressure and stress. Let $\mathbf{g}$ combine external forces, and let τ be the stress tensor. Then the change of momentum density $\rho\mathbf{u}$ in $V(t)$ is given by

$$\frac{d}{dt}\int_{V(t)} \rho\mathbf{u}\,\mathrm{d}\mathbf{x} = \int_{V(t)} \rho\mathbf{g}\,\mathrm{d}\mathbf{x} + \int_{\partial V(t)} \tau \cdot \mathbf{n}\,\mathrm{d}S\mathbf{x} \ . \tag{2.9}$$

The left hand side equals due to Theorem 2.2

$$\int_{V(t)} \rho\frac{D\mathbf{u}}{Dt}\,\mathrm{d}\mathbf{x} \ , \tag{2.10}$$

whereas the *divergence theorem* transforms the right hand side into

$$\int_{V(t)} \rho\mathbf{g} + \nabla \cdot \tau\,\mathrm{d}\mathbf{x} \ . \tag{2.11}$$

Since the momentum balance equation must hold for any control volume, one obtains the *momentum balance equation* in differential form

$$\rho\frac{D\mathbf{u}}{Dt} = \rho\mathbf{g} + \nabla \cdot \tau \ . \tag{2.12}$$

The stress tensor τ is typically decomposed into a pressure part and a *reduced stress tensor*

$$\tau = -pI + \sigma \ . \tag{2.13}$$

Two major special cases result from specific assumptions on the reduced stress tensor

- **Ideal fluid**
  No tangential forces are acting between two layers of fluid. One has

  $$\sigma = 0 \ . \tag{2.14}$$

  This leads to the incomplete Euler equations

  $$\frac{D\rho}{Dt} = -\rho \nabla \cdot \mathbf{u} \tag{2.15}$$

  $$\rho\frac{D\mathbf{u}}{Dt} = -\nabla p + \rho\mathbf{g} \tag{2.16}$$

The system is incomplete, since there are $d+2$ unknowns (ρ, $\mathbf{u}$, p), but only $d+1$ equations. The Euler equations are typically closed by adding an additional equation for the conservation of energy and prescribing an equation of state, which relates pressure to density and energy, as outlined in Section 2.1.3.

- **Stokes hypothesis**
  Neighboring fluid layers influence each other. However, the reduced stress tensor depends linearly on the *velocity deformation tensor* $D = \frac{1}{2}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right)$, which is a valid assumption if the fluid is Newtonian and $\nabla\mathbf{u}$ is small. As derived in [CM00] it follows

$$\sigma = \lambda(\nabla\cdot\mathbf{u})I + 2\mu D \ , \tag{2.17}$$

with appropriate viscosity parameters λ and μ, which we assume to be constant. Inserting into (2.12) yields

$$\rho\frac{D\mathbf{u}}{Dt} = -\nabla p + (\lambda+\mu)\nabla(\nabla\cdot\mathbf{u}) + \mu\Delta\mathbf{u} + \rho\mathbf{g} \ . \tag{2.18}$$

If we additionally assume the flow to be *incompressible*, i.e. $\nabla\cdot\mathbf{u} = 0$, equation (2.18) simplifies to

$$\rho\frac{D\mathbf{u}}{Dt} = -\nabla p + \mu\Delta\mathbf{u} + \rho\mathbf{g} \ . \tag{2.19}$$

If the fluid is *incompressible*, i.e. $\rho = \text{const.}$, then we can write

$$\frac{D\mathbf{u}}{Dt} = -\nabla p' + \nu\Delta\mathbf{u} + \mathbf{g} \ , \tag{2.20}$$

where $p' = \frac{p}{\rho}$ is the normalized pressure and $\nu = \frac{\mu}{\rho}$ is the kinematic viscosity. For simplicity, we omit the prime in the pressure. The incompressibility condition has closed the system, which now are the *incompressible Navier Stokes equations*

$$\begin{aligned} \frac{D\mathbf{u}}{Dt} &= -\nabla p + \nu\Delta\mathbf{u} + \mathbf{g} \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned} \tag{2.21}$$

They are a differential algebraic system. The pressure p is the Lagrange multiplier to satisfy the incompressibility constraint.

2.1.3 Conservation of Energy

Assume one can decompose the energy into kinetic and internal energy

$$E = \frac{1}{2}\rho\mathbf{u}^2 + \rho e \ . \tag{2.22}$$

Here e is the *specific internal energy*. The equations of energy conservation can be derived in various ways from physical principles. A simple conservation argument given by Zel'dovich and Raizer in [ZR66, p. 3] is that in the absence of external energy sources or sinks, the change of specific energy in a control volume equals the work due to compression

$$\frac{De}{Dt} = -p\frac{D}{Dt}\left(\frac{1}{\rho}\right) \ . \tag{2.23}$$

Using the chain rule and the mass balance equations (2.8) one obtains

$$\frac{D}{Dt}\left(\frac{1}{\rho}\right) = -\frac{1}{\rho^2}\frac{D\rho}{Dt} = \frac{1}{\rho}\nabla\cdot\mathbf{u} \ . \tag{2.24}$$

Inserting (2.24) into (2.23) yields the balance law for the internal energy

$$\rho\frac{De}{Dt} = -p\nabla\cdot\mathbf{u} \ . \tag{2.25}$$

A conservation equation for the total energy E can be derived from (2.25) using the mass and momentum balance equations (2.5) and (2.12) as

$$\frac{\partial E}{\partial t} + \nabla\cdot(\mathbf{u}(E+p)) = 0 \ . \tag{2.26}$$

In order to close the system, one has to provide an equation of state, which relates pressure to density and energy

$$p = p(\rho, e) \ . \tag{2.27}$$

Since our main focus is on incompressible flows, we do not specify any relations here. More information can be found in the books of Chorin and Marsden [CM00] and of LeVeque [LV92].

2.2 Numerical Methods for Flow Problems

The numerical approximation of fluid dynamics equations is a wide field. An overview is given e.g. by Anderson [Aj95] or Ferzinger and Peric [FP99]. Most approaches construct a mesh, which is then the basis for constructing finite elements, finite volumes or finite difference stencils.

Meshless approaches generally use a point cloud as a basis and construct finite element basis functions [BJ96], finite volumes [HSS00], or finite difference stencils from it. The latter approach includes SPH and the FPM and will be presented in more detail in the following sections.

For fluid dynamics computations, meshless approaches can be divided into *Eulerian* methods and *Lagrangian* methods (of course, the same applies for approaches with a mesh). In the

Eulerian point of view, the point cloud is stationary, while in the Lagrangian point of view, each point moves with the fluid velocity, i.e. the point cloud moves with the flow. Of course, also in Eulerian methods the point cloud can change in time due to manipulations in order to adapt to changes in geometry, but it does not move with the flow.

Points which move with the fluid flow are called *particles*. One of the first particle methods is Smoothed Particle Hydrodynamics (SPH), which we will briefly present in Section 2.3 for the case of the compressible Euler equations. The method in its original form has proved very successful for computations of compressible, inviscid flows. Viscosity and external forces can be introduced, but not without complications. Also a rigorous treatment of boundary conditions has been a problem in the original SPH. Various improvements, fixes and generalizations have been proposed to overcome these drawbacks of SPH. One of these approaches is the Finite Pointset Method (FPM), which is the main focus of this thesis. In Section 2.4, we present the FPM for the case of the incompressible Navier-Stokes equations.

Various other meshless Lagrangian particle methods for flow problems have been presented, for instance the *finite mass method* by Yserentant et.al. [Yse97, GLY00]. While the finite pointset method focuses on approximating the flow equations themselves, the finite mass method is an example of the opposite direction: Fundamental properties of flows are incorporated into the method. One less approximates the governing equations and more mimics the basic principles of flows.

2.2.1 Numerical Methods for Incompressible Viscous Flows

The incompressible Navier-Stokes equations (2.21) describe flows which on the one hand are fast enough and happen on a large enough length scale (i.e. the Reynolds number [CM00, p. 36] is not too small), such that inertia effects must not be neglected, and on the other hand are slow enough (i.e. the Mach number [FP99, p. 296] is not too large) and the fluid is of such nature, such that firstly viscosity effects play a role, and secondly a significant compression of the fluid does not take place. In the case of large Reynolds numbers the incompressible Navier-Stokes equations exhibit the phenomenon of turbulence [Fri95], which we do not consider in this thesis. We always assume that the solution of the Navier-Stokes equations is of such nature, that a direct numerical simulation (DNS) with an affordable resolution (i.e. number of particles not too large) yields the desired results.

An introduction to numerical methods for the incompressible Navier-Stokes equations is given by Griebel, Dornseifer and Neunhoeffer [GDN98] for the case of the presence of a grid and solely in a Eulerian point of view. However, many of the ideas transfer to meshless particle methods. In particular, the approach of the *Chorin projection method* [Cho68], which is a two step method. In each time step, one does

1. Evolve the momentum equation (2.21) one time step, neglecting the pressure

$$\frac{\tilde{\mathbf{u}}^{(n+1)} - \mathbf{u}^{(n)}}{dt} = \nu \Delta \mathbf{u}^{(n)} + \mathbf{g}^{(n)} \ , \tag{2.28}$$

   where the right hand side is appropriately discretized in space.

2. Correct the intermediate velocity field by a pressure

$$\frac{\mathbf{u}^{(n+1)} - \tilde{\mathbf{u}}^{(n+1)}}{dt} = -\nabla p^{(n+1)} \ , \tag{2.29}$$

   such that $\nabla \cdot \mathbf{u}^{(n+1)}$. This yields a Poisson equation for the pressure $p^{(n+1)}$ to be solved

$$\Delta p^{(n+1)} = \frac{1}{dt} \nabla \cdot \tilde{\mathbf{u}}^{(n+1)} \ . \tag{2.30}$$

Alternatively, one can implement the SMAC method [AH70a, AH70b], which is formally equivalent, but can be numerically preferable. In the SMAC method one keeps a hydrodynamic pressure $p^{(n)}$ in the momentum equation (2.28). The correction step then computes a small correction $\delta p^{(n)}$ to guarantee the incompressibility constraint. In both approaches, the flow is in the first step treated as if it was was compressible, which violates the incompressibility constraint. This error is then corrected in the second step.

The projection approaches for solving incompressible flows have been investigated for meshless particle methods. Cummins and Rudman [CR98, CR99] have presented how to apply a projection approach in SPH (see Section 2.3). Tiwari and Manservisi [MT02] applied the Chorin projection method to the FPM (see Section 2.4).

In a meshless context, the Poisson equation for the pressure requires the Laplace operator to be approximated on the cloud of particles. Additionally, the viscosity term includes the vector Laplace operator. In the case of ν being a constant, this operator is essentially d times the scalar Laplacian for the interior points. Hence it is straightforward to use one and the same discretization submatrix for all these cases. The question how to discretize the Laplace operator on the point cloud is one major aspect of this thesis. Chapter 4 and the following chapters are devoted to this aspect.

Remark 2.2. The momentum equation (2.28) is given in time explicit form. For the case of large viscosities ν or for complex boundary conditions, it can also be solved semi or fully implicit. In this case, a vector Poisson equation has to be solved. If the viscosity ν is constant, this is essentially d scalar Poisson equations, which, however, may be coupled via the boundary conditions.

In computations often times a major part of the computation time is devoted to the treatment of the arising Poisson equations. While in regular geometries with a mesh present the solution

of the linear systems is the main task, in meshless methods already the setup of the linear systems (i.e. the discretization of the Laplace operator) can require a large computational effort. Of course, solving the linear systems becomes particularly important if not only one, but several systems have to be solved in each time step. In this case, fast and robust solvers for the arising systems are required.

Remark 2.3. The presented projection approaches are so-called SIMPLE ("semi-implicit method for pressure-linked equations") methods [PS72]. They admit a separate treatment of the flow part and the pressure part. For particle methods, one can deal with the particle related aspects (moving, management, etc.) in the first step, and consider the second step as an isolated Poisson problem to be solved on the cloud of particles. This isolated point of view allows a separate analysis of each step, and gives a clear view on where and how to accelerate the methods. In this thesis, we deal with the first step in Chapter 3, and with the second step in Chapter 4 and the following chapters. In particular, in Chapter 6 we present a new approach on how to construct the linear systems, and in Chapter 7 we deal with the aspect of solving them. On the other hand, there are connections between the two steps, which must not be neglected. Additionally, the SIMPLE approach limits accelerations to each of the steps separately. For the aspect of multigrid approaches, for instance, the SIMPLE approach admits the use of multigrid methods for the solution of the Poisson problems. However, the outer iteration will remain untouched, which leaves a great potential of improvement (consider the remark in [TOS01, p. 315] on this aspect).

2.2.2 Meshless Methods for Elliptic Problems

It has to be pointed out that the two steps of firstly setting up the linear systems and secondly solving them have to be considered in connection. For a regular grid it is easy and straightforward to discretize the Laplace operator, hence much of the literature in this field mainly focuses on the aspect of solving the systems. A canonical finite difference approximation on a regular grid yields symmetric positive system matrices, which is an extremely favorable situation. Even more, for regular grids, often times finite difference and finite element approaches yield the same systems.

In a meshless context, however, there is no canonical way of discretizing the Laplacian. There is a wide range of approaches, and each may lead to different matrices with different properties.

Meshless finite element approaches use the point cloud as a basis for constructing finite dimensional subspaces of the solution space of the elliptic problem ($H^1(\Omega)$, $H_0^1(\Omega)$, etc.). Typically each point contributes to one basis function. The Diffuse Element Method (DEM), as introduced by Nayroles et.al. [NTVR92], uses the shape functions obtained by the moving

least squares method, as outlined in Section 5.2. Based on the DEM, Belytschko et.al. introduced the Element-Free Galerkin Method (EFGM) [BLG94], which uses a regular background grid for the numerical quadrature. Hence, the EFGM is an example of a hybrid method, which on the one hand uses a meshless approximation approach, on the other hand uses a grid for some computational instances. A comparison of the methods is presented by Duarte [Dua95]. Another difference between the DEM and the EFGM is that the latter uses additional terms in the derivative of the MLS function, a difference analogous to the local and global finite difference differentiation, as described in Section 5.2.4.

A major class of meshless methods are the Partition of Unity (PUM) methods. Babuska and Melenk introduced the Partition of Unity Finite Element Method (PUFEM) [BJ96, BJ97], which uses smooth partition of unity functions as basis functions for a finite element Galerkin approach. More general PUM approaches and computational issues have been presented by Griebel and Schweitzer [GS00, GS02a]. The authors present a PUM multilevel solver [GS02b]. As opposed to the multigrid approaches presented in Section 7.4 the PUM matrices are symmetric, which can be used in the multilevel approach. Two main drawbacks of many classical meshless approaches, the low order of approximation (e.g. SPH), and a lack of local adaptivity, have been remedied by Duarte et.al. with the introduction of *h*-*p*-clouds [DLT96, DO96].

The meshless finite element approaches construct finite dimensional subspaces of the solution spaces of the elliptic problem, which provides powerful tools of analysis. The approximation of the weak form of the underlying equation yields generally symmetric linear systems to be solved, given the differential operators are symmetric. Both is generally not the case with meshless finite difference methods. These approximate the strong form of the differential equation by exploiting local Taylor expansions. On the other hand, finite difference methods provide more flexibility in dealing with boundary conditions and various other local features.

The first non-regular finite difference approaches were presented for unstructured meshes, i.e. point clouds in which there is additional neighborhood information available. Perrone and Kao [PK75] presented in 1975 a general approach, which was later extended and generalized by Liszka, Orkisz, et.al. [LO80, DKL84] to the Generalized Finite Difference Method (GFDM). The general approach for arbitrary meshes can be easily extended to a meshless finite difference approach by defining the neighborhood of a point in the cloud (see Section 3.5 on various approaches for a neighborhood of a point). An overview over the class of GFDM is provided by Orkisz in [Kle98, Chap. III]. An application of the method to elliptic interface problems is presented by Iliev and Tiwari [IT02].

In meshless finite difference methods the resulting linear systems are in general non-symmetric, since there is no symmetry in the influence of a point $\mathbf{x}_i$ on the local approximation of another point $\mathbf{x}_j$ and the converse local approximation. Even though the system matrices are non-symmetric, one can ask for various other properties of the finite difference approx-

imation to yield. For instance, in the case of the Poisson equation to be approximated, a desirable property would be the system matrices to have an M-matrix structure, which would constitute in a discrete maximum principle, as outlined in Section 6.1.2. This approach is the main focus of this thesis and has been presented by the author in [Sei06b, Sei06a].

2.3 Smoothed Particle Hydrodynamics

Smoothed Particle Hydrodynamics (SPH) is one of the first methods for the numerical solution of hydrodynamic equations, and most Lagrangian particle methods originate from it. The first works on the field of SPH were presented by Lucy [Luc77] and Gingold and Monaghan [GM77]. Both articles present an approach based on the Monte-Carlo method to numerically solve the multidimensional hydrodynamics equations, which arise in astrophysics. Rasio [Ras00] provides an overview over particle methods in astrophysical fluid dynamics. Hernquist and Katz [HK89] showed how to introduce a variable smoothing length as well as presented how hierarchical tree data structures can reduce the complexity of the method to $O(n \log n)$. The method of SPH was generalized by Monaghan to a wider class of hydrodynamic equations [Mon88, Mon92], also the treatment of free boundaries has been investigated [Mon94].

The classical SPH is based on smoothing kernels and their derivatives. Dilts [Dil99] and Kuhnert [Kuh99] included the moving least squares method into the SPH approach, allowing an easy approximation of higher order derivatives to a higher order of approximation. Dilts denotes his approach *moving least squares particles hydrodynamics*, Kuhnert names his approach *generalized smoothed particles hydrodynamics*, later *Finite Pointset Method* (FPM).

The fundamental differences between SPH and FPM are described in Section 2.3.5. The FPM was applied successfully to problems with complex, time-dependent geometries [KT02b] and free surface flows [KT03]. The application to incompressible flows is described Section 2.4

2.3.1 Smoothing and Particle Approximation

Let f be a field quantity. Then

$$f(\mathbf{x}) = \int_{\mathbb{R}^d} f(\tilde{\mathbf{x}}) \delta(\mathbf{x} - \tilde{\mathbf{x}}) \, \mathrm{d}\tilde{\mathbf{x}} \; . \tag{2.31}$$

Assume to have a class of kernels $W(\cdot, h)$ which approximate the delta distribution, i.e.

$$\lim_{h\to 0} W(\mathbf{x}, h) = \delta(\mathbf{x}) \tag{2.32}$$

$$\int_{\mathbb{R}^d} W(\mathbf{x}, h)\,\mathrm{d}x = 1 \tag{2.33}$$

Additionally one requires the kernels to be radially symmetric and compactly supported

$$W(\mathbf{x}, h) = w(\|\mathbf{x}\|, h) \tag{2.34}$$

$$w(d, h) = 0 \quad \forall d > h \tag{2.35}$$

Replacing in (2.31) the delta distribution by the kernel yields the smoothed quantity

$$f^h(\mathbf{x}) = \int_{\mathbb{R}^d} f(\tilde{\mathbf{x}}) W(\mathbf{x} - \tilde{\mathbf{x}}, h)\,\mathrm{d}\tilde{\mathbf{x}}\ , \tag{2.36}$$

which approximates f

$$\lim_{h\to 0} f^h(\mathbf{x}) = f(\mathbf{x})\ . \tag{2.37}$$

Thus the parameter h is referred to as *smoothing length*.

Let ρ be a density. Introducing ρ into (2.36) one obtains in terms of the density measure $\mu_\rho(A) = \int_A \rho\,\mathrm{d}\mathbf{x}$

$$f^h(\mathbf{x}) = \int_{\mathbb{R}^d} \frac{f(\tilde{\mathbf{x}})}{\rho(\tilde{\mathbf{x}})} W(\mathbf{x} - \tilde{\mathbf{x}}, h)\rho(\tilde{\mathbf{x}})\,\mathrm{d}\tilde{\mathbf{x}} = \int_{\mathbb{R}^d} \frac{f(\tilde{\mathbf{x}})}{\rho(\tilde{\mathbf{x}})} W(\mathbf{x} - \tilde{\mathbf{x}}, h)\,\mathrm{d}\mu_\rho(\tilde{\mathbf{x}})\ . \tag{2.38}$$

The integral can be approximated by a Monte-Carlo like quadrature. Let $\{X^{(n)}\}_{n\in\mathbb{N}}$ be sequence of point clouds, where each point cloud $X^{(n)} = \left(\mathbf{x}_1^{(n)}, \ldots, \mathbf{x}_n^{(n)}\right)$ has associated weights $\left(m_1^{(n)}, \ldots, m_n^{(n)}\right)$, such that the point measure $\delta X^{(n)} = \sum_{i=1}^n m_i^{(n)} \delta_{\mathbf{x}_i^{(n)}}$ converges to the density measure μ_ρ

$$\lim_{n\to\infty} \delta X^{(n)} = \mu_\rho\ . \tag{2.39}$$

Replacing the integral in (2.38) by the discrete measure $\delta X^{(n)}$ yields the approximation

$$f^h(\mathbf{x}) \approx \int_{\mathbb{R}^d} \frac{f(\tilde{\mathbf{x}})}{\rho(\tilde{\mathbf{x}})} W(\mathbf{x} - \tilde{\mathbf{x}}, h)\,\mathrm{d}\delta X^{(n)}(\tilde{\mathbf{x}}) \tag{2.40}$$

$$= \sum_{i=1}^n m_i^{(n)} \frac{f\left(\mathbf{x}_i^{(n)}\right)}{\rho\left(\mathbf{x}_i^{(n)}\right)} W\left(\mathbf{x} - \mathbf{x}_i^{(n)}, h\right) =: f^{h,n}(\mathbf{x}) \tag{2.41}$$

Working with the quantity $u^{h,n}$ has the advantage that its only space dependence is in the kernel. Hence, for constance h, the derivatives apply solely to the kernel

$$\nabla f^{h,n}(\mathbf{x}) = \sum_{i=1}^n m_i^{(n)} \frac{f\left(\mathbf{x}_i^{(n)}\right)}{\rho\left(\mathbf{x}_i^{(n)}\right)} \nabla W\left(\mathbf{x} - \mathbf{x}_i^{(n)}, h\right)\ . \tag{2.42}$$

In practice, the derivative of the kernel function can be implemented analytically or computed in advance.

2.3.2 Numerical Method

In order to obtain a numerical method, one shifts the focus from the limits $h \to 0$ and $n \to \infty$ towards considering a fixed number of particles and a fixed smoothing length (which can in applications vary in time and space). The field quantities are considered only at the point positions $\mathbf{x}_i$, i.e. every particle is equipped with a vector of data $\mathbf{f}_i$. Omitting the number of particles in the expressions, we obtain for the approximate quantity, given by (2.41), for the k^{th} point

$$f_k^h = \sum_{i=1}^{n} \frac{m_i}{\rho_i} f_i W_{ki}^h \ , \tag{2.43}$$

where $W_{ki}^h = W\left(\mathbf{x}_k - \mathbf{x}_i, h\right)$. Thus a quantity at a point $\mathbf{x}_k$ is approximated by a linear combination of quantities at neighboring points. The same holds due to (2.42) for derivatives of the quantity at $\mathbf{x}_k$, of course with different weights

$$\nabla f_k^h = \sum_{i=1}^{n} \frac{m_i}{\rho_i} f_i \nabla W_{ki}^h \ . \tag{2.44}$$

2.3.3 Application to Hydrodynamic Equations

As a simple model for hydrodynamic equations we consider the adiabatic compressible Euler equations without external forces, as derived in Section 2.1 (see [CM00] for more details). It is a nonlinear system of conservation laws. The unknowns are density ρ, velocity $\mathbf{u}$ and internal energy e. The pressure $p = p(\rho, e)$ is a function of density and internal energy. In d space dimensions, on has $d+2$ equations for $d+2$ unknowns

$$\begin{aligned} \frac{\partial \rho}{\partial t} &= -\nabla \cdot (\rho \mathbf{u}) \\ \frac{D\mathbf{u}}{Dt} &= -\frac{\nabla p}{\rho} \\ \frac{De}{Dt} &= -\frac{p}{\rho} \nabla \cdot \mathbf{u} \end{aligned} \tag{2.45}$$

Using the chain rule and the notation of the material derivative $\frac{D}{Dt}$ the system can be rewritten in Lagrangian form

$$\begin{aligned} \frac{D\rho}{Dt} &= \mathbf{u} \cdot \nabla \rho - \nabla \cdot (\rho \mathbf{u}) \\ \frac{D\mathbf{u}}{Dt} &= -\nabla \left(\frac{p}{\rho} \right) - \frac{p}{\rho^2} \nabla \rho \\ \frac{De}{Dt} &= \mathbf{u} \cdot \nabla \left(\frac{p}{\rho} \right) - \nabla \cdot \left(\frac{\rho \mathbf{u}}{\rho} \right) \end{aligned} \tag{2.46}$$

The fundamental idea of Lagrangian particle methods is to move the points with the velocity of the fluid $\mathbf{u}$. A moving point is called *particle*. With the approximations derived in Section 2.3.1, one can approximate the Euler system on the point cloud. Each particle carries besides its position vector $\mathbf{x}_i$ the approximate velocity $\mathbf{u}_i^h$ and the approximate energy e_i^h. Additionally one has the masses m_i and the densities ρ_i^h. For a finite number of particles one obtains a system of ordinary differential equations. Since the particles move with the flow, the material derivatives in the equations' left hand side become ordinary time derivatives. The right hand sides are approximated according to equation (2.44), which yields

$$\begin{aligned}
\dot{\rho}_k^h &= \mathbf{u}_k^h \sum_{i=1}^n m_i \nabla W_{ki}^h - \sum_{i=1}^n m_i \mathbf{u}_i^h \nabla W_{ki}^h \\
\dot{\mathbf{u}}_k^h &= -\sum_{i=1}^n \frac{m_i}{\rho_i^h} \frac{p_i^h}{\rho_i^h} \nabla W_{ki}^h - \frac{p_k^h}{\left(\rho_k^h\right)^2} \sum_{i=1}^n m_i \nabla W_{ki}^h \\
\dot{e}_k^h &= \mathbf{u}_k^h \sum_{i=1}^n \frac{m_i}{\rho_i^h} \frac{p_i^h}{\rho_i^h} \nabla W_{ki}^h - \sum_{i=1}^n \frac{m_i}{\rho_i^h} \frac{p_i^h \mathbf{u}_i^h}{\rho_i^h} \nabla W_{ki}^h
\end{aligned} \tag{2.47}$$

The system can be rewritten as

$$\dot{\rho}_k^h = \sum_{i=1}^n m_i \left(\mathbf{u}_k^h - \mathbf{u}_i^h\right) \nabla W_{ki}^h \tag{2.48}$$

$$\dot{\mathbf{u}}_k^h = -\sum_{i=1}^n m_i \left(\frac{p_k^h}{\left(\rho_k^h\right)^2} + \frac{p_i^h}{\left(\rho_i^h\right)^2} \right) \nabla W_{ki}^h \tag{2.49}$$

$$\dot{e}_k^h = \sum_{i=1}^n m_i \frac{p_i^h}{\left(\rho_i^h\right)^2} \left(\mathbf{u}_k^h - \mathbf{u}_i^h\right) \nabla W_{ki}^h \tag{2.50}$$

Remark 2.4. One can show (see for instance [Kuh99]) that if the masses m_i remain unchanged in time, then the discrete total mass, momentum and energy are conserved. Note that the discrete conservation is obtained for the SPH discretization of system (2.46). On the other hand, one does not obtain these conservation properties when discretizing system (2.45) directly.

In practice, one will not implement the density evolution equation (2.48), but instead in every time step compute the density for each particle using relation (2.43) as

$$\rho_k^h = \sum_{i=1}^n m_i W_{ki}^h \ . \tag{2.51}$$

The pressure at a particle is computed as $p_k^h = p\left(\rho_k^h, e_k^h\right)$. Omitting the superscript ^{h} in the

approximate quantities, the final SPH particle system becomes

$$\dot{m}_k = 0 \tag{2.52}$$

$$\dot{\mathbf{x}}_k = \mathbf{v}_k \tag{2.53}$$

$$\dot{\mathbf{u}}_k = -\sum_{i=1}^{n} m_i \left(\frac{p_k}{(\rho_k)^2} + \frac{p_i}{(\rho_i)^2} \right) \nabla W_{ki} \tag{2.54}$$

$$\dot{e}_k = \sum_{i=1}^{n} m_i \frac{p_i}{(\rho_i)^2} (\mathbf{u}_k - \mathbf{u}_i) \nabla W_{ki} \tag{2.55}$$

$$\rho_k = \sum_{i=1}^{n} m_i W_{ki} \tag{2.56}$$

$$p_k = p(\rho_k, e_k) \tag{2.57}$$

2.3.4 Features of SPH

SPH is famous and infamous for the following features:

- SPH is a fully meshless method. The only objects to be treated in data structures are the particles and the field information they carry.
- SPH is a Lagrangian method. It satisfies the continuity equation exactly.
- SPH has proven to be very robust, instabilities seldom occur. This is positive for treating physically correct problems. However, often times also for unphysical models, SPH yields stable, but of course unphysical results.
- The method yields satisfying results already for small numbers of particles. However, due to the low order of accuracy, highly accurate results can in general not be achieved with a reasonable amount of particles.
- An accurate treatment of discontinuities or steep gradients requires a smart adaptation of the smoothing length and the particle density.
- In order to have reasonable particle densities, a particle management is required. Care has to be taken when merging and adding new particles, that the total mass in conserved.
- In its basic form, SPH has problems in treating boundary conditions. The radially symmetric smoothing kernels yield a good approximation in the interior of the domain. At the boundary, however, the good approximation properties break down (see [Kuh99, p. 19]). Monaghan [Mon92, Mon94] has introduced the idea of including boundary

particles, which exert forces on the particles in the interior. For fixed and also for flow-dependent boundary potentials (as investigated by Hartig [Har96]), this approach can model solid wall boundary conditions. Vila [Vil99] instead modifies the smoothing kernels near the boundaries and investigates the use of ghost particles. In a purely meshless context these are difficult to treat in complex geometries. The FPM by Kuhnert [Kuh99] equips every boundary with boundary particles. Since derivatives are approximated by the MLS method instead of smoothing kernels, one can easily incorporate all kinds of boundary conditions, as outlined in Section 2.4.1.

- Not all physical effects can easily be included into the SPH in its basic form. External forces, for instance, and also viscosity, require great care to be included.

2.3.5 From SPH to FPM

The Finite Pointset Method (FPM) is based on fundamental ideas of SPH. A cloud of particles is constructed in the fluid domain. Each particle carries a vector of field information. The particles are moved with the fluid velocity. The flow equations are formulated in Lagrangian form and evaluated at the particles. The equations' left hand side becomes a normal time derivative. The equations' right hand side is approximated using particles in a circular neighborhood. In some sense the FPM is a generalization of SPH, and there are fundamental differences to the classical version of SPH:

- The approximation of differential operators, such as the Laplace operator, is done by a moving least squares approach, which is a more general approach.

- Boundary particles are placed at all domain boundaries. Boundary conditions are implemented via the boundary particles.

- While in the classical SPH particles have a mass associated, this aspect does not appear in the FPM anymore. The particles are merely interpolation points for the field information, including density. This makes the method more flexible, and admits an easier particle management (see Section 2.5) and an easier treatment of boundaries. The price to pay is the method not being strictly conservative anymore, as opposed to SPH.

The FPM has been introduced by Kuhnert [Kuh99]. It has been successfully applied to the compressible Euler and Navier-Stokes equations [Kuh02].

2.4 FPM for Incompressible Navier-Stokes Equations

We consider the incompressible Navier-Stokes equations (2.21) on the fluid domain $\Omega(t)$. At time $t = 0$ the initial domain $\Omega(0) = \Omega_0$ is given, as well as the initial velocity $\mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0(\mathbf{x})$. The computational domain can vary with time, and if free boundaries are present, it is part of the unknowns. On the boundary $\partial\Omega$ various types of boundary conditions can be prescribed, as described in Section 2.4.1.

The Navier-Stokes equations shall be solved by the FPM, using a projection approach, as presented in Section 2.2.1. The Chorin projection method has been applied to the FPM by Tiwari and Manservisi [MT02]. In each time step one performs the following two steps:

1. Move the particles and solve equation (2.21) as if the flow was compressible

$$\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} + dt\ \mathbf{u}^{(n)} \tag{2.58}$$

$$\tilde{\mathbf{u}}^{(n+1)} = \mathbf{u}^{(n)} + dt\ \nu\ \Delta\mathbf{u}^{(n)} + dt\ \mathbf{g}^{(n)} \tag{2.59}$$

2. Project $\tilde{\mathbf{u}}^{(n+1)}$ onto the subspace of divergence free velocity fields

$$\mathbf{u}^{(n+1)} = \tilde{\mathbf{u}}^{(n+1)} - dt\ \nabla p^{(n+1)}, \tag{2.60}$$

   where $p^{(n+1)}$ solves the Poisson equation

$$\Delta p^{(n+1)} = \frac{1}{dt}\nabla \cdot \tilde{\mathbf{u}}^{(n+1)}. \tag{2.61}$$

The update relation (2.58) applies to interior and boundary points, where the latter can have velocity zero (e.g. no-slip solid wall boundary points). The momentum update (2.59) is d times as large as the number of points present. Conditions at boundary points, as described in Section 2.4.1, have to be imposed in this relation. If derivatives of $\mathbf{u}$ are included in the boundary conditions, one has to solve implicitly for the boundary points. The Laplace operator for the velocity can be incorporated implicitly into the system. This is a must for large viscosities in order to not have a limiting CFL-condition, but also for moderate viscosities this approach has proven advantageous. In case of small viscosities, efficient linear solvers converge very fast, anyhow. Altogether, the update step (2.59) requires a dn dimensional system to be solved, where d is the space dimension, and n is the total number of points. In case of a constant viscosity, the d velocity components have d equal blocks for the interior points, which may be coupled only via the boundary conditions. If the boundary conditions also decouple, one obtains d independent linear systems for each velocity component.

The Poisson equation (2.61) is discretized as described in Chapter 4. The approximation for the interior points is identical to the d Laplace-blocks in the matrix for the velocity for a constant viscosity.

In general, the movement of the particles is inexpensive. Computationally expensive are the discretization of the Laplace operator and the solution of the arising linear systems, which is one Poisson system (2.61) for the pressure, and a large system for the d velocity components. The solution of the linear systems is investigated in Chapter 7.

2.4.1 Boundary Conditions

Imposing proper boundary conditions partly depends on the underlying physics, such as stresses at boundaries, surface tension, etc. We briefly provide different types of boundaries we are confronted with in the considered viscous flow problems, and outline how the conditions are implemented in the FPM. Generally, in the FPM every boundary is covered by boundary particles. Sometimes it is known a-priori which particles are boundary particles, such as at inflow boundaries. In other cases it might be necessary to identify the boundary points during the computation, such as for free boundaries. Every boundary particle is marked which kind of the boundaries given below it represents. Accordingly, the resulting systems are set up.

Boundary conditions have to be provided for the velocity in equation (2.59), and for the pressure in equation (2.61). For the examples we consider in this thesis, the boundary conditions are the following.

- **Inflow boundary**
  At the inflow boundary, constantly inflow particles are placed. These quickly become internal particles, but at start they are inflow boundary particles. The following boundary conditions are prescribed

$$\mathbf{u} = \mathbf{u}_{\text{in}} \tag{2.62}$$

$$\frac{\partial p}{\partial \mathbf{n}} = \rho \mathbf{g} \cdot \mathbf{n} \tag{2.63}$$

  The inflow velocity is directly prescribed. The condition for the pressure is derived from multiplying the momentum equation (2.21) by a normal vector, and assuming that the inflow velocity is constant and no normal stresses are present at the inlet.

- **Solid wall boundary**
  For large viscosities one will prescribe a no-slip condition, in which case the particles stay in position, and the boundary condition for the velocity is

$$\mathbf{u} = 0 \ . \tag{2.64}$$

  For small viscosities one will impose slip conditions, in which case the boundary particles move along the solid wall. The boundary condition for the velocity is

$$\mathbf{u} \cdot \mathbf{n} = 0 \quad \text{and} \quad \frac{\partial \mathbf{u}}{\partial \mathbf{n}} \cdot \mathbf{t} = 0 \quad \forall \mathbf{t} \cdot \mathbf{n} = 0 \ , \tag{2.65}$$

i.e. there is no velocity component into or away from the wall, and the tangential velocity components do not change when approaching the wall. In both cases the pressure condition is derived as for inflow boundaries as

$$\frac{\partial p}{\partial \mathbf{n}} = \rho \mathbf{g} \cdot \mathbf{n} \ . \tag{2.66}$$

- **Free boundary**
  The free surface is described by the free boundary particles themselves. In each time step it needs to be checked which particles are free boundary particles. At a free boundary, surface tension acts. As described by Landau and Lifshitz [LL91], one has a force balance between stress tensor and surface tension forces

$$\tau \cdot \mathbf{n} = k\kappa \mathbf{n} \ . \tag{2.67}$$

  Here τ is the stress tensor, k is the surface tension, and κ is the curvature of the surface. According to (2.13) the stress tensor can be decomposed into a pressure part and a reduced stress tensor. The latter is expressed (2.17) in terms of the symmetrized velocity deformation tensor. Inserting these into (2.67) yields the condition

$$-p\mathbf{n} + \mu \left(\nabla \mathbf{u} + (\nabla \mathbf{u})^T \right) \cdot \mathbf{n} = k\kappa \mathbf{n} \ . \tag{2.68}$$

  Multiplying (2.68) by $\mathbf{n}^T$ yields a condition coupling pressure and velocity

$$p = 2\mu \frac{\partial \mathbf{u}}{\partial \mathbf{n}} \cdot \mathbf{n} - k\kappa \ . \tag{2.69}$$

  Multiplication with $\mathbf{t}^T$ yields $d-1$ conditions for the velocity

$$\frac{\partial \mathbf{u}}{\partial \mathbf{n}} \cdot \mathbf{t} + \frac{\partial \mathbf{u}}{\partial \mathbf{t}} \cdot \mathbf{n} = 0 \ . \tag{2.70}$$

  These conditions can be supplemented by the incompressibility condition $\nabla \cdot \mathbf{u} = 0$. The computation of flows with surface tension with the FPM is described by Kuhnert and Tiwari [KT02b].

- **Outflow boundary**
  At an outflow boundary, conditions have to be prescribed. Hence, stationary points are placed at the outflow boundary. Interior particles are removed once they come too close to the outflow boundary. At an outflow boundary, one typically prescribes homogeneous Neumann conditions for the velocity, while the pressure is prescribed at the outlet.

$$\frac{\partial \mathbf{u}}{\partial \mathbf{n}} = 0 \tag{2.71}$$

$$p = p_{\text{out}} \tag{2.72}$$

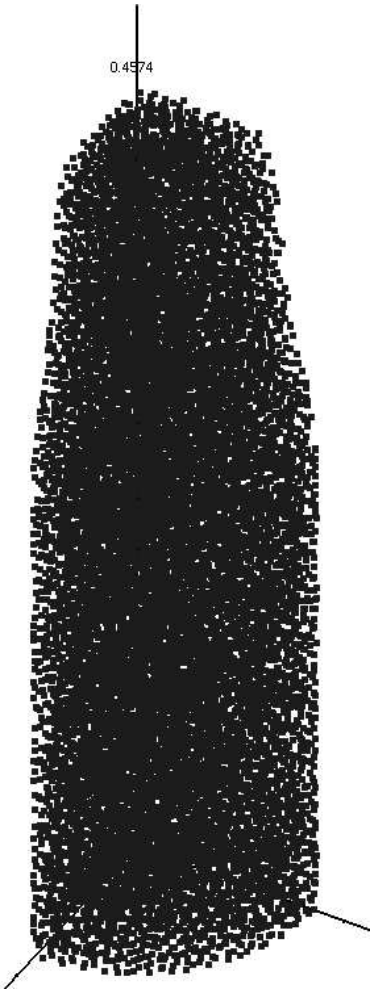

Figure 2.1: Glass Melting

Figure 2.2: Tank filling

2.4.2 Examples of Application

At the *Fraunhofer Institut für Techno- und Wirtschaftsmathematik*, the FPM has been applied to simulate flow problems in various fields. An overview over industrial applications is given at [FPM]. We outline two examples of applications to real three dimensional geometries. In Chapter 8 we will test our new method in these simulations. The considered problems are firstly the melting of a glass cylinder and secondly the filling of a car tank.

Glass Melting

We consider a technical process called "Senken" in the German glass industry. A long glass cylinder with small diameter is placed in a shell with a larger diameter. The glass cylinder is heated in an oven. It becomes liquid and sinks down, until it hits the wall of the outer shell. Once the glass has sunken down, it is cooled again. This way, the long and thin cylinder has

been transformed into a short and thick cylinder. The density of the glass can be assumed to be constant, the fluid is assumed to be incompressible. Challenging in this process is the fact that the viscosity depends strongly on the temperature. Hence, temperature has to be incorporated into the governing equations. It couples to the momentum equation via the glass viscosity $\mu(T)$. The system of equations is

$$\frac{D\mathbf{u}}{Dt} = -\nabla p + \frac{1}{\rho}\nabla \cdot \sigma + \mathbf{g} \tag{2.73}$$

$$\frac{DT}{Dt} = \nabla \cdot (\nu \nabla T) \tag{2.74}$$

$$\nabla \cdot \mathbf{u} = 0 \tag{2.75}$$

where the stress tensor is $\sigma = \mu\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right)$. Since the viscosity is not constant, the momentum equations do not decouple, as they do for constant viscosities. Using vector identities, one can express $\nabla \cdot \sigma$ using Laplace operator type derivatives only. This way, the results derived for the Poisson equation can be used for an implicit solve of the viscosity in the momentum equation. Additionally, the pressure correction equation has to be solved in each time step, as familiar from the case of constant viscosities. The parameter ν in the temperature equation is assumed to be constant. The glass forms the computational domain. Wherever it touches the outer shell, no-slip boundary conditions are imposed. Everywhere else, there are free boundaries. Figure 2.1 shows a snapshot of a time dependent simulation. Due to radial symmetry, only a quarter of a cylinder is simulated. At the bottom half, the glass already touches the outer shell, while at the upper half, the molten glass still sinks down. A movie of the evolution is shown in Figure 8.19.

Tank Filling

The geometry of the pipe connecting the nozzle inlet with the tank is given. The inlet is the nozzle of the spigot. A long and curved pipe leads to the tank. To be computed is the flow of the petrol into the pipe. The computational domain is the petrol itself. The computational domain changes with time and it is part of the unknowns. The governing equations are the incompressible Navier-Stokes equations (2.21). The geometry has various types of boundaries. The boundaries are implemented by boundary particles, as described in Section 2.4.1. They are

- An *inflow boundary* at the nozzle
- A *solid wall boundary (with slip)* at the pipe casing
- A *free boundary* between petrol and air
- An *outflow boundary* into the tank

The flow is described by the instationary incompressible Navier-Stokes equations, as given in Section 2.4. The computational domain $\Omega(t)$ depends on time. At time $t = 0$ the domain is a short fluid cylinder at the tip of the nozzle, in which all particles are released with the inflow velocity $\mathbf{u}_{\text{in}}$. The particles move into the pipe and fall under the influence of gravity. When hitting a surface of the pipe, solid wall boundary particles are included at the "wet" parts of the pipe casing. In Figure 2.2 a snapshot of a time dependent simulation is shown. Visualized are only the points. Since they are many, they form the boundaries of the computational domain. The dark parts are wall boundaries, the bright parts are free boundaries. Not visible in this plot are the interior points (since they are – of course – in the interior) and the inflow and outflow boundaries, which are covered by other points. Figure 8.21 shows a movie of the evolution.

2.5 Particle Management

In the FPM particles move with the fluid velocity. Since the velocity field is the solution itself, one cannot predict in advance where the particles will move. Hence, even if the particles are well distributed initially, they will in general cluster in some places and become scarce in others. The former can in best case result in an unnecessarily high local resolution and thus computational effort, in the worst case yield numerical instabilities. The latter can spoil the accuracy and cause problems in the point neighborhood search.

In order to remedy these problems, a constant particle management has to be done. Particles too close to each other have to be removed, or better merged, while new particles have to be inserted into large holes and gaps. This requires on the one hand efficient detection procedures for nearby particles and holes (see Section 3.2), on the other hand correct interpolation of the field data, when particles are merged and inserted. In Section 2.6 we analyze the particle management for Burgers' equation.

Both approaches presented below, adding and removing particles, might be done multiple times, until a satisfying point cloud is achieved.

2.5.1 Adding Particles

One prescribes a maximum hole size in the point cloud. All centers of locally maximal void balls with diameter larger than the prescribed maximal hole size are found, as described in Section 3.2.4. Into each center of a hole a new particle in inserted. If the points are in general position, the center of the hole has equal minimal distance from $d + 2$ neighboring points. The field data at this new particle is interpolated from the neighboring particles by scattered data interpolation. The simplest approach is to assign the arithmetic mean value

of all field quantities at the $d+2$ closest neighboring points. This approach works well in 1d, as shown in Section 2.6 for Burgers' equation. In higher space dimensions, it is more stable to incorporate more than the closest neighbors into the interpolation. A Shepard interpolation, as described in Section 5.2.3, typically yields satisfying results. One can also use higher order MLS approximations, as described in Section 5.2. Kuhnert [Kuh99] advises to additionally smooth the data.

2.5.2 Removing Particles

Particles which are too clustered are thinned by merging pairs of closeby points into a single one. By an iterative application, also large clusters can be thinned out. One prescribes a minimum distance between two points. Assume two points closer than this distance are detected (as described in Section 3.2.3). Then both are removed and replaced by a new point, i.e. the two particles are merged into a new single one. Let the two particles have the positions $\mathbf{x}_k$ and $\mathbf{x}_l$, and let the data $\mathbf{u}_k$ and $\mathbf{u}_l$ be associated data vectors. The new particle is inserted in the center of mass of the two particles, and the data is linearly interpolated.

$$\mathbf{x} = \frac{\mathbf{x}_k + \mathbf{x}_l}{2} \tag{2.76}$$

$$\mathbf{u} = \frac{\mathbf{u}_k + \mathbf{u}_l}{2} \tag{2.77}$$

Kuhnert [Kuh99] describes that for some problems a further smoothing of the data can be required.

Note that the position of the new particle is actually not too important, since the two particles are close by anyhow (i.e. one could place the new particle at a position of one of the two removed ones). The data, however, must be interpolated equally. In Section 2.6 we investigate for the 1d Burgers' equation the case of a different interpolation. In particular, one must not simply remove one of the two particles, since this would result in an incorrect approximation of discontinuous solutions. For smooth solutions with moderate gradients, the merging and averaging is generally more accurate.

2.6 Particle Management for Burgers' Equation

Burgers' equation is the simplest model for flow problems in one space dimension. It is a scalar equation, consisting of a Lagrangian derivative with a homogeneous right hand side, or a second derivative in the case of the viscous Burgers' equations. In both cases, the FPM can be applied. The application to the inviscid equation is particularly simple, thus particle management can be analyzed. The application to the viscous equation requires to

approximate the second derivative of the solution, which the FPM does by a meshless least squares approximation.

We analyze the aspect of particle management, as described in Section 2.5. This is in the 1d case the following:

- If two particles x_k and x_l are too close, both are removed and replaced by a new particle with position $\frac{x_k+x_l}{2}$. The new function value is $\frac{u_k+u_l}{2}$.
- If two neighboring particles x_k and x_l are too far away from each other, a new particle is added with position $\frac{x_k+x_l}{2}$. The new function value is $\frac{u_k+u_l}{2}$.

In other words, a new particle's position and function value is obtained in exactly the same manner, no matter if particles are removed or inserted. We shall motivate that this fact relates to the Rankine-Hugoniot condition for conservation laws.

2.6.1 Solutions of Burgers' Equation

Burgers' equation in conservation form it is given by

$$u_t + \left(\frac{1}{2}u^2\right)_x = 0 \tag{2.78}$$

and models compressible inviscid flows. Consider the Cauchy problem, i.e. the spacial variable extends to the whole $\mathbb{R}$, and the time variable is defined on $[0, \infty)$.

$$u_t + uu_x = 0 \tag{2.79}$$

$$u(x, 0) = g(x) \tag{2.80}$$

The solution is given by the method of characteristics, as described in [Eva98]. The characteristic curves are lines in the x–t-plane, starting in the initial line $t = 0$ with slope $g'(x_0)$. If the initial condition g is not everywhere monotonically increasing, smooth solutions exist only for finite times. At the first time the characteristic lines meet, the concept of weak solutions yields a shock solution, i.e. a moving discontinuity, whose speed s is thereafter given by the *Rankine-Hugoniot condition*

$$s = \frac{u^+ + u^-}{2} \ , \tag{2.81}$$

as derived in [LV92]. Here u^+ and u^- are the function values left and right of the shock. Upward discontinuities in the initial conditions yield a rarefaction wave, which is obtained as a unique weak solution if an additional entropy condition is imposed (see [LV92, Eva98]).

We also consider the viscous Burgers' equation

$$u_t + uu_x = \nu u_{xx} \tag{2.82}$$

$$u(x,0) = g(x) \tag{2.83}$$

For every $\nu > 0$ smooth solutions exist for all times. The solution can be derived using the *Hopf-Cole transformation* [Eva98] as

$$u(x,t) = \frac{\int_{-\infty}^{\infty} \frac{x-y}{t} e^{-\frac{|x-y|^2}{4\nu t} - \frac{h(y)}{2\nu}} \,\mathrm{d}y}{\int_{-\infty}^{\infty} e^{-\frac{|x-y|^2}{4\nu t} - \frac{h(y)}{2\nu}} \,\mathrm{d}y} \tag{2.84}$$

where

$$h(x) = \int_{-\infty}^{\infty} g(y) \,\mathrm{d}y \tag{2.85}$$

In the limit $\nu \to 0$, the solutions of (2.82) converge pointwise to the entropy solution of (2.79).

In case the space domain is a bounded interval, boundary conditions have to be imposed. For simplicity, we consider the special case $u(x,t) \geq 0 \; \forall x, t$, i.e. the flow always moves from left to right. For the inviscid Burgers' equation, the characteristic curves enter at the left boundary and leave at the right one. Consequently, a boundary condition has to be imposed at the left, the inflow boundary. If the inflow velocity is controlled, one assigns Dirichlet boundary conditions.

The viscous Burgers' equation is parabolic, hence two boundary conditions have to be imposed. A reasonable choice is to impose a Dirichlet boundary condition at the inflow and a homogeneous Neumann boundary condition at the outflow boundary.

2.6.2 FPM for Inviscid Burgers' Equation

Burgers' equation in Lagrangian coordinates becomes

$$\frac{Du}{Dt} = 0 \;, \tag{2.86}$$

i.e. each particles moves with a velocity equal to its function value, and the function value at each particle remains constant. The FPM for Burgers' equation is

1. Generate points $\mathring{x}_1, \ldots, \mathring{x}_n$.
2. Assign function values $(u_1, \ldots, u_n) = (g(\mathring{x}_1), \ldots, g(\mathring{x}_n))$.
3. Move each particle x_i with velocity u_i.

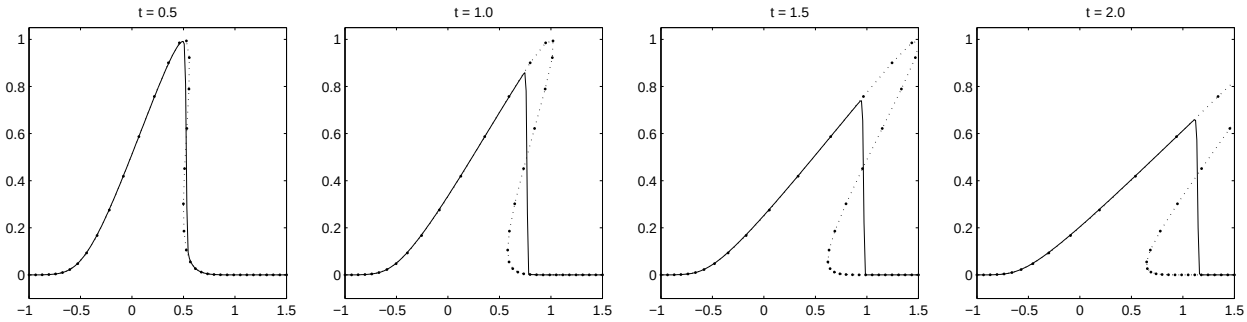

Figure 2.3: FPM without particle management for inviscid Burgers' equation with Gaussian initial data

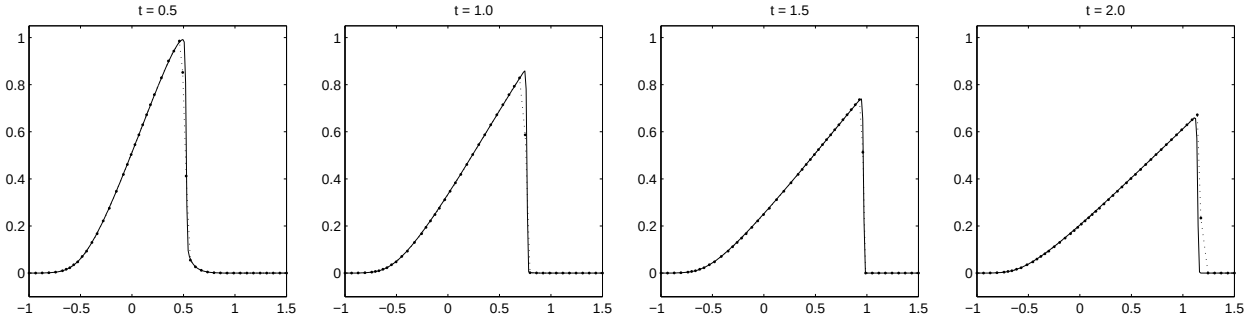

Figure 2.4: FPM with particle management for inviscid Burgers' equation with Gaussian initial data

The FPM system of ordinary differential equations is

$$\dot{x}_i = u_1 \qquad x_i(0) = \mathring{x}_i \tag{2.87}$$

$$\dot{u}_i = 0 \qquad u_i(0) = g(\mathring{x}_i) \tag{2.88}$$

The FPM is a characteristic method and is exact, since the value at each point is exactly transported along the characteristic curve. The derived ordinary differential equation can in principle be run for arbitrarily long times. However, if the true solution to a given setup becomes discontinuous, the FPM will not yield a function solution anymore.

Consider Burgers' equation in a bounded domain with Gaussian initial data

$$u_t(x,t) + u(x,t)u_x(x,t) = 0 \quad \text{for } (x,t) \in (-1,1.5) \times (0,2) \tag{2.89}$$

$$u(x,0) = \exp(-10x^2) \quad \text{for } x \in [-1,1.5] \tag{2.90}$$

$$u(-1,t) = 0 \quad \text{for } t \in [0,2] \tag{2.91}$$

For times $t > \sqrt{\frac{e}{20}}$, the true solution has a shock, i.e. a discontinuity which the characteristic curves run into. Figure 2.3 shows the solution obtained by the FPM without particle management (points with dotted connection) in comparison with the correct solution (solid line), plotted as a movie at times $t \in \{0.5, 1.0, 1.5, 2.0\}$. The FPM solution can be interpreted as a

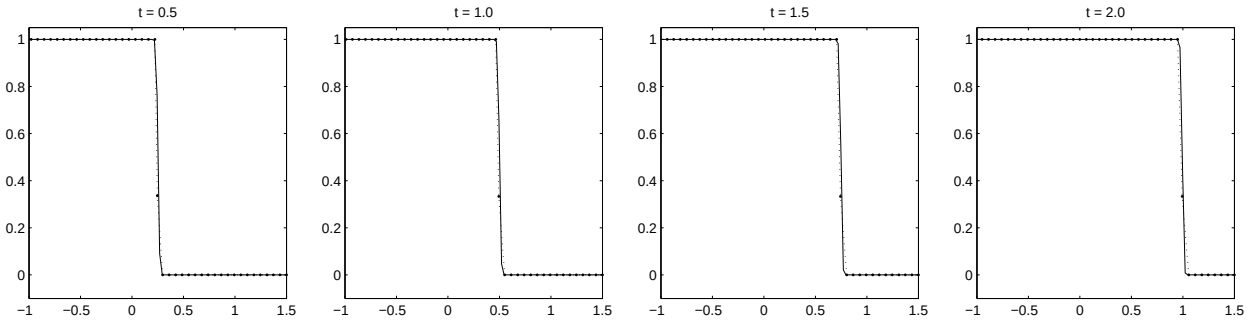

Figure 2.5: FPM with particle management for inviscid Burgers' equation with Riemann problem initial data

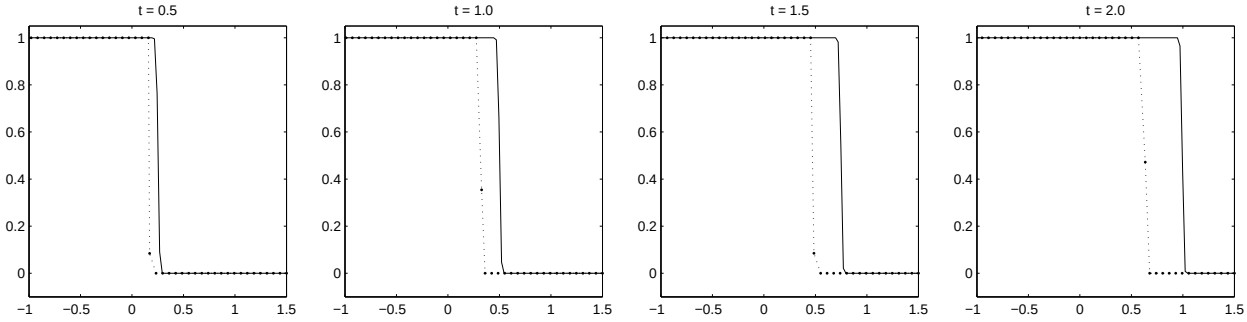

Figure 2.6: FPM with incorrect particle management for inviscid Burgers' equation with Riemann problem initial data

breaking wave solution, i.e. a solution which is not a function anymore. If one wants to solve Burgers' equation in the sense of breaking waves, the FPM yields the exact solution for all times. With respect to applications in fluid dynamics, however, there must always be one unique function value. A mechanism has to be included into the FPM which prevents the particles from overtaking each other. The particle management, as presented in Section 2.5, is such a mechanism. Before the particles overtake each other, they come very close to each other. Particles which are too close to each other are merged into a single particle in the middle (with the new function value the average of the old function values). If the time steps are small enough such that particles cannot overtake each other without being close one step before, one has a guarantee that the solution is always a function.

Figure 2.4 shows the solution obtained by the FPM with particle management, again in comparison with the correct solution. Observe that firstly the FPM solution is a function and secondly the resulting function coincides fairly well with the correct solution. Indeed, particle management with appropriate interpolation can already be sufficient for yielding weak solutions with correct shock speeds. We formulate the statement in the following

Remark 2.5. If the particle density left of the shock equals the particle density right of the shock, then the FPM with merging particles with averaging position and function value

yields correct shock speeds.

This statement is comparably vague and it cannot be easily extended to higher space dimensions or to systems like the compressible Euler equations. However, one can verify the statement for the one dimensional Burgers' equation with Riemann shock initial conditions and initially equidistantly spaced particles.

We consider the following problem

$$u_t(x,t) + u(x,t)u_x(x,t) = 0 \quad \text{for } (x,t) \in (-1, 1.5) \times (0,2) \tag{2.92}$$

$$u(x,0) = g(x) \quad \text{for } x \in [-1, 1.5) \tag{2.93}$$

$$u(-1,t) = 1 \quad \text{for } t \in [0,2] \tag{2.94}$$

where

$$g(x) = \begin{cases} 1 & x < 0 \\ 0 & x \geq 0 \end{cases} . \tag{2.95}$$

The correct solution is the similarity solution

$$u(x,t) = g\left(x - \frac{1}{2}t\right) , \tag{2.96}$$

i.e. the shock moves with speed $s = \frac{1}{2}$ to the right. Figure 2.5 shows the FPM solution in comparison with the correct solution. The two solutions coincide well. For the sake of a simpler analysis, we initially distribute particles equidistantly over the interval $[-3.5, 1.5]$, i.e. the particles initially stretch out over the left boundary, such that for the times $t < 2$ particles flow into the domain. In practice, one will of course insert new particles at the left boundary one after another.

In order to see that the averaging in the particle management is relevant, we run the same simulation, however new points are inserted at position and with function value

$$x = \lambda x_l + (1-\lambda)x_r \tag{2.97}$$

$$u = \lambda u_l + (1-\lambda)u_r \tag{2.98}$$

The correct interpolation $\lambda = 0.5$ is shown in Figure 2.5. In comparison, Figure 2.6 shows the case of an incorrect particle management ($\lambda = 0.2$). Here, two particles coming too close are replaced by a new particle which is too slow.

Figure 2.7 shows the paths of the particles in the x–t-domain, in the limit minimal particle distance $d_{\min} \to 0$. Strictly speaking, there is no unique shock speed. Instead, there is always a single particle inside the shock, which is constantly merged with other particles and thus constantly changes its velocity. Alternatingly, particles with velocity $u = 0$ and ones with velocity $u = 1$ hit the shock particle and slow it down respectively speed it up. In the beginning, there is some offset, but after some time, the central particle's speed approaches a behavior toggling between two velocities.

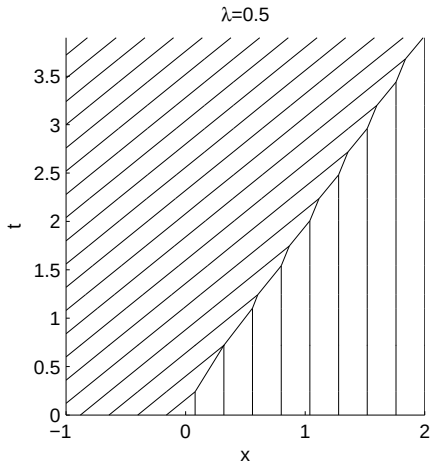

Figure 2.7: Particle movement with correct interpolation

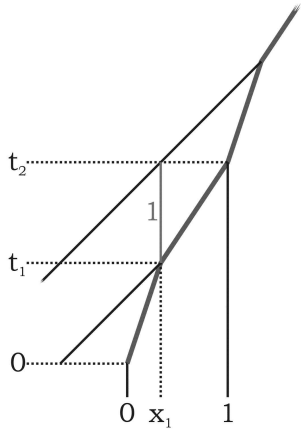

Figure 2.8: Effective shock speed

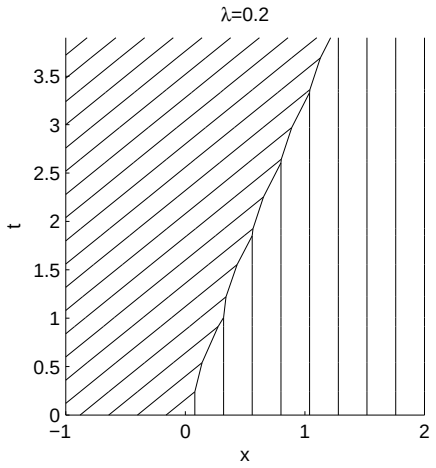

Figure 2.9: Particle movement with incorrect interpolation

Theorem 2.3. *For initially equidistant points, the effective shock speed with correct particle management is in the limit* $s = \frac{1}{2}$.

Proof. Consider the case of a toggling between two velocities s_1 and s_2. Let the interpolation be weighted with λ vs. $(1-\lambda)$. Then the two velocities satisfy

$$s_2 = \lambda + (1-\lambda)s_1 \tag{2.99}$$

$$s_1 = \lambda s_2 \tag{2.100}$$

Consequently

$$s_2 = \frac{\lambda}{1-\lambda+\lambda^2} \ . \tag{2.101}$$

Assume w.l.o.g. the particle distance to be 1. Then particles move as sketched in Figure 2.8. Due to the velocity $u = 1$ of particles coming from the left, one has

$$t_2 - t_1 = 1 \ . \tag{2.102}$$

The limiting velocities are s_1 and s_2, thus the distance on the x-axis satisfies

$$1 = s_1 t_1 + s_2(t_2 - t_1) = s_1 t_1 + s_2 = \frac{\lambda^2 t_1 + \lambda}{1-\lambda+\lambda^2} \tag{2.103}$$

This yields $t_2 = \frac{1-\lambda+\lambda^2}{(1-\lambda)^2}$, and thus the shock speed, averaged over the time interval $[0, t_2]$, equals

$$s = \frac{(1-\lambda)^2}{1-\lambda+\lambda^2} \ . \tag{2.104}$$

The correct shock speed $s = \frac{1}{2}$ is only obtained for the averaging $\lambda = \frac{1}{2}$, in which case the velocity toggles between $u = \frac{1}{3}$ and $u = \frac{2}{3}$. □

For comparison, Figure 2.9 shows the paths of the particles when the particle velocities upon impact are weighted 0.2 to 0.8. The merged particle is too slow, and it takes two particles of velocity $u = 1$ to make up for one particle with velocity $u = 0$. The resulting effective shock speed is significantly lower than $\frac{1}{2}$.

Remark 2.6. The presented results only hold if the initial particle density is uniform over the whole space. If, for instance, the particles left of the shock are only half as dense, fewer particles will "push" from the left than from the right, yielding a too small shock speed. If the initial particle densities are known, one could equip the point with "masses", and incorporate the masses as weights into the interpolation (i.e. $\lambda \neq \frac{1}{2}$ is chosen appropriately depending on the ration of masses). This would mean in some sense a return to SPH, in which particles *do* have masses, while in the FPM they do not need to have (although they can, see [Kuh99]). Note, however, that when combining two particles to a single one, the mass of the new particle must not be the sum of the two particle masses, since then the shock particle would become heavier and heavier and would not be influenced by incoming particles anymore. While the approach of fixing variable particle densities with particle masses could be followed, in general it is better to put the responsibility for uniformly distributed particles on the particle management.

We have seen that for the Riemann problem the correct shock speed, which is given by the Rankine-Hugoniot condition $s = \frac{u^+ + u^-}{2}$, is obtained only if particles are inserted precisely averaging the velocities $u = \frac{u^+ + u^-}{2}$. Since solutions to other initial conditions can be approximated by multiple solutions of Riemann problems [LV92], one can generalize this fact to Burgers' equation with general initial condition.

2.6.3 FPM for Viscous Burgers' Equation

Another approach to obtain a shock solution is to add a small viscosity into Burgers' equation. As the weak entropy solution of inviscid Burgers' equation is the vanishing viscosity limit of the solutions of viscous Burgers' equation, a small choice of ν should prevent the particles from overtaking each other, since close to the smoothed shock large second derivatives slow the particles down. Since we are in a particle concept, small values of ν require tiny time steps for the viscosity to slow down the particles, i.e. the FPM system for Burgers' equation with small viscosities is very stiff.

We use the approximating local least squares method, as described in Section 5.1.1, for the approximation of the second derivative. The radius of neighboring points is for each experiment and in each time step chosen uniformly 2.5 times the maximum distance between neighboring points.

Figure 2.10 and Figure 2.11 show the numerical results for $\nu = 4 \cdot 10^{-2}$, the first without, the second with particle management. Figure 2.12 and Figure 2.13 show the numerical results

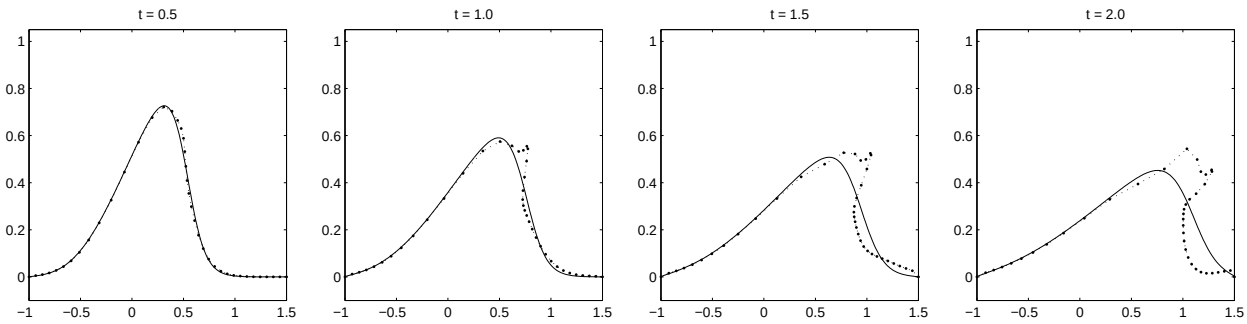

Figure 2.10: FPM without particle management for viscous Burgers' equation ($\nu = 4 \cdot 10^{-2}$) with Gaussian initial data

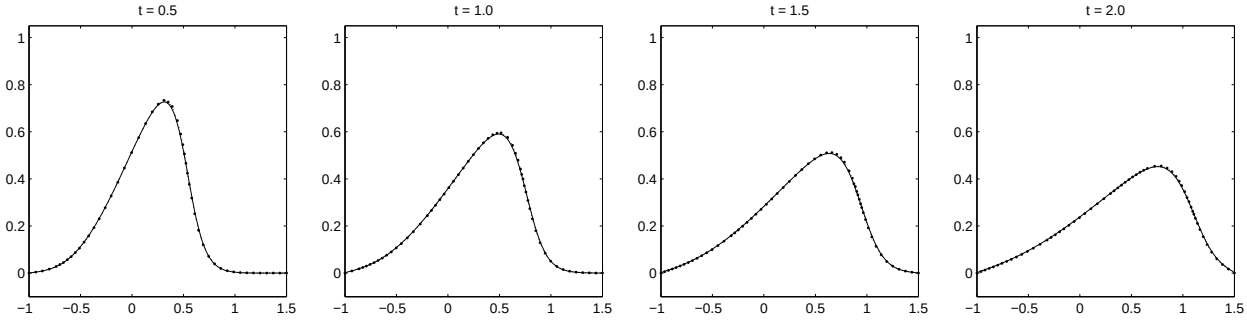

Figure 2.11: FPM with particle management for viscous Burgers' equation ($\nu = 4 \cdot 10^{-2}$) with Gaussian initial data

for $\nu = 8 \cdot 10^{-2}$, again the first without, the second with particle management. In all cases a Gaussian function is chosen as initial data.

Obviously, the method without particle management fails for the small viscosity. As can be seen in Figure 2.10, the viscosity is not large enough to prevent the particles from overtaking each other. Once this has taken place, the function is interpreted as being oscillatory, which can drive the points away, yielding oscillations in time. After some time, the numerical solution will typically blow up.

On the other hand, the numerical solution without particle management shows satisfying results for the larger viscosity. Here the second derivatives are large enough to prevent the particles from overtaking each other. Figure 2.12 shows the solution close to the truth. However, there is no hope for this numerical solution to remain stable, since no particles are removed. Instead, more and more particles cluster in the hump's lee side. In the last picture ($t = 2.0$), one can already see an onset of instability.

As opposed to the cases without management, the simulations with particle management, as shown in Figure 2.11 and Figure 2.13, show very satisfying results. The numerical solution is very close to the correct solution. Note that these results are obtained by considering only 40 points. A numerical method with a fixed grid, which discretizes the nonlinear term by

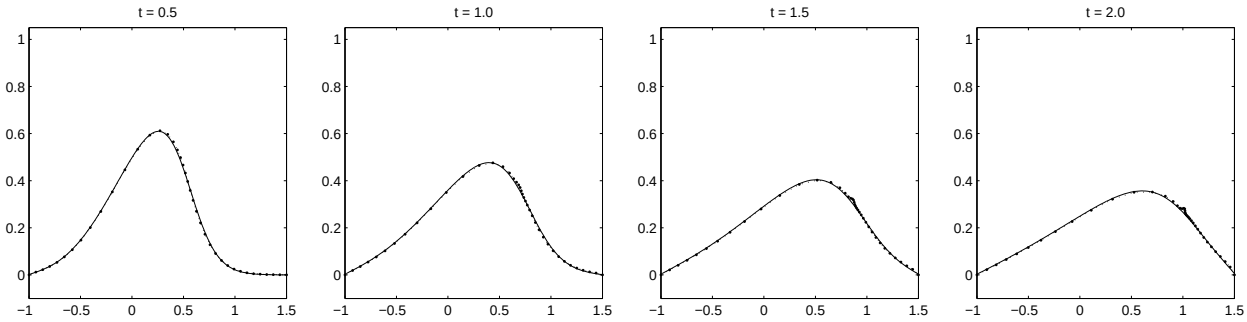

Figure 2.12: FPM without particle management for viscous Burgers' equation ($\nu = 8 \cdot 10^{-2}$) with Gaussian initial data

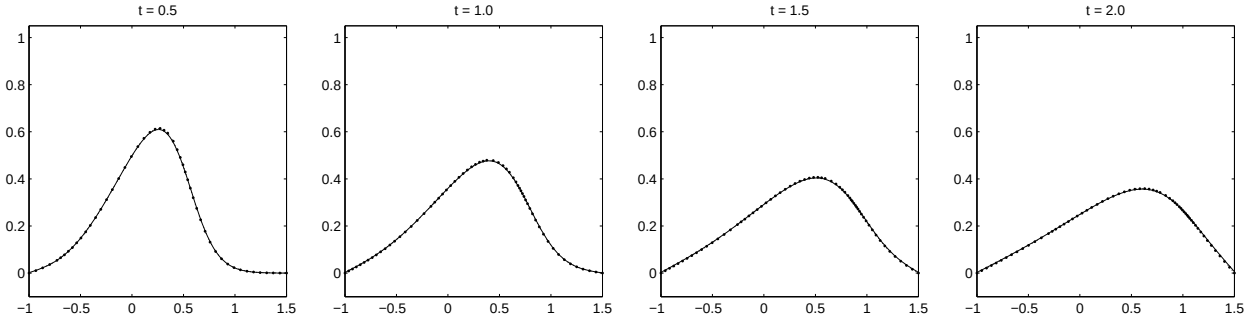

Figure 2.13: FPM with particle management for viscous Burgers' equation ($\nu = 8 \cdot 10^{-2}$) with Gaussian initial data

finite differences, would require significantly more points to achieve a comparable accuracy.

We can conclude that particle management is a fundamental feature of the FPM. The method does not run stable without it. On the other hand, if one interpolates correctly, one obtains correct discontinuous solutions solely by particle management. With view on the Navier-Stokes equations, the application to viscous Burgers' is of interest. Indeed, the FPM with particle management and a least squares approximation to the second derivatives yields an accurate method for such nonlinear advection-diffusion equations.

Chapter 3

Point Clouds

Let $\Omega \subset \mathbb{R}^d$ be a domain. For the sake of simplicity, we assume Ω to be connected. For domains with multiple components of connectivity the presented results simply apply to each of the components. In the finite pointset method, Ω is the computational domain at some time t. Assume the domain Ω and its boundary $\partial\Omega$ have a geometry, such that a well posed solution of the governing equations exists.

Let the boundary of the domain be denoted $\Gamma = \partial\Omega$. Depending on the problem being solved, the boundary consists of different components Γ_i, where $\Gamma_1 \cup \cdots \cup \Gamma_k = \Gamma$. Each component may have a different physical interpretation (such as free boundary, no-slip boundary, etc.). In the Poisson problem considered in Chapter 4, the two types are Dirichlet and Neumann boundary conditions.

We now consider a finite set of points $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \subset \overline{\Omega}$ ($n \geq 2$), which consists of interior points $X_i = \{\mathbf{x}_1, \ldots, \mathbf{x}_{n_i}\} \subset \Omega$ and boundary points $X_b = \{\mathbf{x}_{n_i+1}, \ldots, \mathbf{x}_n\} \subset \Gamma$. If different types of boundaries are present, the corresponding sets of points are denoted X_{b_i}. The only information are the point positions, no neighborhood or connectivity information are provided. Hence we speak of a *point cloud*.

Although boundary points are specifically marked, we additionally assume that the point cloud "describes" the geometry of Ω. This means in particular that distances between points must be significantly smaller than holes and gaps in the geometry of Ω. This rather vague formulation can be best described by the configurations shown in Figures 3.1 to 3.5. The cloud of 70 points, shown in Figure 3.2 does not describe the geometry given in Figure 3.1, since without marking the boundary points, the hole in the domain cannot be distinguished from holes in the point cloud due to distances of points. The cloud shown in Figure 3.3 with 260 is acceptable, and the cloud with 1140 points, shown in Figure 3.4 describes the geometry very accurately, as the plot in Figure 3.5 shows.

The task is to solve the equations of interest on the point cloud. This can mean many

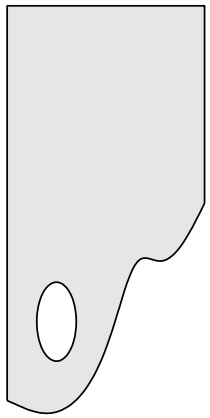

Figure 3.1: Domain Ω

Figure 3.2: 40 interior and 27 boundary points

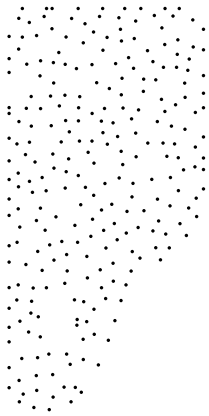

Figure 3.3: 200 interior and 60 boundary points

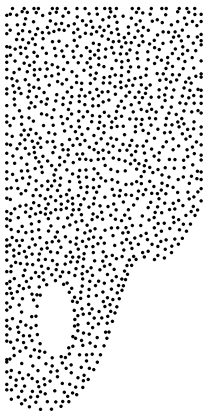

Figure 3.4: 1000 interior and 140 boundary points

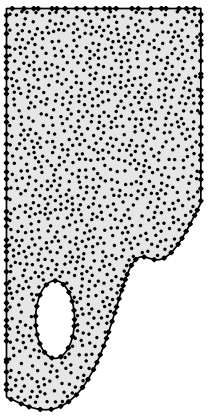

Figure 3.5: 1000 interior and 140 boundary points with domain Ω

things. In terms of the finite element method (FEM), for instance, this means that the solution, which lives in one function space, shall be approximated by another function from an appropriate finite dimensional subspace. The point cloud is used to characterize the subspace of the large function space. This can be done by forming a Delaunay triangulation of the point cloud, and using piecewise linear basis functions on the triangles, respectively tetrahedra, to solve a weak formulation of the equations to be solved. Obviously, this approach is not meshless. Meshless versions of FEM are partition of unity methods, such as PUM [GS00, GS02a] or PUFEM [BJ96, BJ97]. Also element-free Galerkin methods [BLG94] fall into this context.

In this thesis we are interested in finite difference (FD) approaches. Unlike FEM, the governing equations are discretized in their original strong form. The solution of the equations is approximated at the points x_i in terms of the function values, while FEM approximates in an L^2 sense in the considered function spaces.

Every point $\mathbf{x}_i$ carries a set of information $u_1(\mathbf{x}_i), \ldots, u_\nu(\mathbf{x}_i)$, which are the field quantities $u_1, \ldots, u_\nu$ evaluated at the points. A value of these quantities at a position $\mathbf{x} \in \Omega$, which is no data point, has to be approximated by interpolation using data values of neighboring points. The same holds for derivatives of the field quantities at the data points themselves.

3.1 Point Cloud Geometry

As defined by Levin [Lev98], we introduce

Definition 3.1. Let $\Omega \subset \mathbb{R}^d$ be a domain and X a point cloud, both as described above. Then we denote

- **Mesh size** h
  Define h as the minimal real number, such that
  $$\bar{\Omega} \subset \bigcup_{i=1}^{n} \bar{B}\left(\mathbf{x}_i, \frac{h}{2}\right) \tag{3.1}$$
  where $\bar{B}(\mathbf{x}, r)$ is the closed ball of radius r centered in $\mathbf{x}$ and $\bar{\Omega}$ is the closure of Ω.

- **Density** ρ
  For any point $\mathbf{x} \in \bar{\Omega}$, define $\rho(\mathbf{x})$ as the maximal real number, such that
  $$\#\{X \cap B(\mathbf{x}, qh)\} \leq \rho(\mathbf{x}) \cdot q^d \quad \forall q \geq 1 \tag{3.2}$$
  where $\sharp\{Y\}$ denotes the number of elements of a set Y.

- **Separation** δ
  Define δ as the maximum real number, such that
  $$\|\mathbf{x}_i - \mathbf{x}_j\| \geq h\delta \quad \forall i \neq j \tag{3.3}$$

Remark 3.1. The parameters introduced in Definition 3.1 have the following interpretations

- The mesh size h is the diameter of the largest hole in the point cloud. For $\varepsilon > 0$ there is at least one point $\mathbf{x} \in \Omega$, such that the ball $B(\mathbf{x}, h - \varepsilon)$ contains no point. Note that not necessarily $B(\mathbf{x}, h - \varepsilon) \subset \Omega$, i.e. near the boundary holes are smaller. In the FPM, enough boundary points will be present, in order to have no big holes at the boundary. Too large holes in the point cloud can spoil the solution's accuracy. Additionally, in Chapter 6 we use a bound on the mesh size to prove the existence of positive Laplace stencils.

- For every point $\mathbf{x} \in \Omega$ (and actually also for points outside of Ω, the density $\rho(\mathbf{x})$ measures the amount of points in the neighborhood. In complex geometries, point clouds with varying local density are desired. The circular neighborhood distance, as considered in Chapter 5 and Chapter 6 should be chosen dependent on the density ρ in order to keep the number of considered points in the same order allover the domain Ω.

- The quantity $h\delta$ is the smallest distance between points in the point cloud. As we consider finite point sets, this number is positive. Unless properties of the geometry or the solution of the equations require a significant local refinement of the point cloud, too small values of δ should in general be avoided, as these imply a larger number of points and may lead to numerical instabilities.

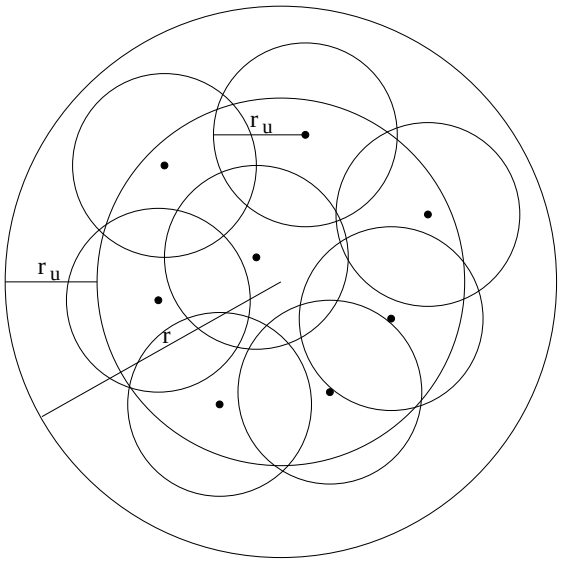

Figure 3.6: Minimum number of points inside a ball

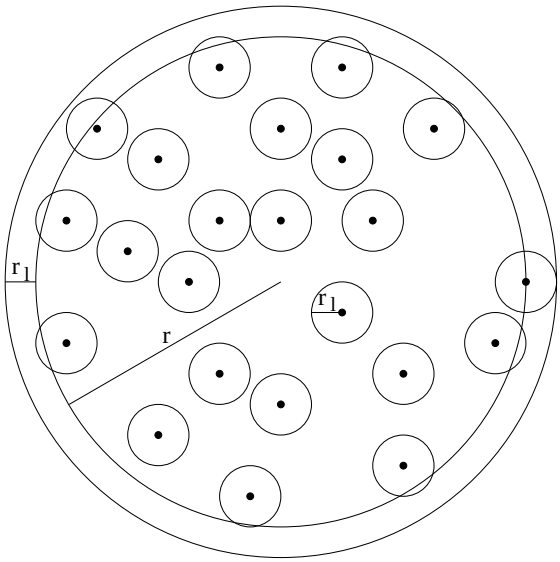

Figure 3.7: Maximum number of points inside a ball

Theorem 3.1. *If Ω is connected, then $\delta \leq 1$.*

Proof. Let $\mathbf{x}_k$ and $\mathbf{x}_l$ be two points of minimal distance. Let $d_{\min}$ denote this distance. Let $\varepsilon > 0$ and $d = d_{\min} - \varepsilon$. Obviously, the two balls $B\left(\mathbf{x}_k, \frac{d}{2}\right)$ and $B\left(\mathbf{x}_l, \frac{d}{2}\right)$ have no common points. Since any other two points have even larger distances, the set $C = \bigcup_{i=1}^{n} \bar{B}\left(\mathbf{x}_i, \frac{d}{2}\right)$ consists of n disjoint balls. However, if $\delta > 1$, there would be an $\varepsilon > 0$, such that $\bar{\Omega} \subset C$. This contradicts Ω being connected. □

Theorem 3.2. *If sufficiently many boundary points are present, one has even $\delta \leq \frac{\sqrt{3}}{2}$ in 2d, and $\delta \leq \sqrt{\frac{2}{3}}$ in 3d.*

Proof. Due to Remark 3.1, if sufficiently many boundary points are present, the mesh size h is the diameter of the largest void ball inside the domain $\Omega \subset \mathbb{R}^d$. This ball must touch at least $d+1$ points, otherwise there would be freedom to slightly move and increase the ball. The setup, such that the minimal distance between $d+1$ points on a sphere becomes maximal, is an equilateral triangle in 2d and an equilateral tetrahedron in 3d. The ratio between circumcircle diameter and triangle/tetrahedron edge length are the above values. □

The density ρ provides an estimate for the average number of points inside a ball with radius r. The mesh size h provides a lower bound for the number of points inside a ball.

Theorem 3.3. *Let $X \subset \Omega \subset \mathbb{R}^d$ be a point cloud with mesh size h. Any ball $B(\mathbf{x}, r) \subset \Omega$ contains at least*

$$\left(2\frac{r}{h} - 1\right)^d \tag{3.4}$$

points.

Proof. The setup is shown in Figure 3.6 with $r_u = \frac{h}{2}$. By definition of h, the union $\bigcup_{i=1}^{n} \bar{B}\left(\mathbf{x}_i, \frac{h}{2}\right)$ covers the whole domain. The ball $B(\mathbf{x}, r)$ can be divided into two parts. On the one hand, the ring $B(\mathbf{x}, r) \setminus B\left(\mathbf{x}, r - \frac{h}{2}\right)$. This set can be covered by balls $B\left(\mathbf{x}_i, \frac{h}{2}\right)$ corresponding to points positioned slightly outside the ball $B(\mathbf{x}, r)$, i.e. no points need to lie inside. On the other hand, there is the ball $B\left(\mathbf{x}, r - \frac{h}{2}\right)$. It must contain points $\mathbf{x}_{i_1}, \ldots, \mathbf{x}_{i_m}$, such that it is covered by the union $\bigcup_{\nu=1}^{m} \bar{B}\left(\mathbf{x}_{i_\nu}, \frac{h}{2}\right)$. A necessary condition for this is that the volumes satisfy

$$\lambda(B\left(\mathbf{x}, r - \frac{h}{2}\right)) \leq \lambda(\bigcup_{\nu=1}^{m} \bar{B}\left(\mathbf{x}_{i_\nu}, \frac{h}{2}\right)) \ . \tag{3.5}$$

This implies

$$\alpha(d)\left(r - \frac{h}{2}\right)^d \leq m\ \alpha(d)\left(\frac{h}{2}\right)^d \ , \tag{3.6}$$

where $\alpha(d)$ is the volume of the unit ball in $\mathbb{R}^d$. Solving for m yields the number of points to equal at least

$$m \geq \left(2\frac{r}{h} - 1\right)^d \ . \tag{3.7}$$

□

In a similar manner, the separation δ provides an upper bound for the number of points inside a ball with radius r.

Theorem 3.4. *Let $X \subset \Omega \subset \mathbb{R}^d$ be a point cloud with the minimum distance between points not smaller than $h\delta$. Any ball $B(\mathbf{x}, r)$ contains at most*

$$\left(2\frac{r}{h\delta} + 1\right)^d \tag{3.8}$$

points.

Proof. The setup is shown in Figure 3.7 with $r_l = \frac{h\delta}{2}$. Let $\mathbf{x}_1, \ldots, \mathbf{x}_m$ denote the points inside the ball $B(\mathbf{x}, r)$. Consider the union $S = \bigcup_{i=1}^{m} B\left(\mathbf{x}_i, \frac{h\delta}{2}\right)$. Since the points have distances of at least $h\delta$, the balls of the union do not intersect. Hence, the volume satisfies $\lambda(S) = m\ \alpha(d)\left(\frac{h\delta}{2}\right)^d$, where $\alpha(d)$ is the volume of the unit ball in $\mathbb{R}^d$. On the other hand, the union satisfies $S \subset B\left(\mathbf{x}, r + \frac{h\delta}{2}\right)$. This implies for the volumes

$$m\ \alpha(d)\left(\frac{h\delta}{2}\right)^d \leq \alpha(d)\left(r + \frac{h\delta}{2}\right)^d \ . \tag{3.9}$$

Solving for m yields an upper bound for the number of points

$$m \leq \left(2\frac{r}{h\delta} + 1\right)^d \ . \tag{3.10}$$

□

Example 3.1. Assume a very strict particle management guarantees $\delta = \frac{1}{3}$. Then a ball with radius $r = 2h$ contains at least 3^d and at most 13^d points. Obviously, these bounds are in general not sharp.

3.2 Data Management

Meshless methods, in particular particle methods, for which the point cloud changes from time step to time step, require a fair amount of data management in order to be efficient. Specific data structures and data management are not the main focus of this thesis. Still, in this Section we briefly outline the fundamental ideas required to obtain efficient methods.

The points are numbered from 1 to n and for each index one has direct access to the position $\mathbf{x}_i$ and the vector of field data $\mathbf{u}_i$, which can involve velocity, energy, properties (interior or boundary point), etc. The fundamental operation to be done on the point cloud is to find for a point its neighbors inside a ball with given radius r. Let us assume that the radius r is constant all over the domain Ω.

If merely the list of points is used, for each point all other points have to be checked for being a neighbor. Doing so for all points is an effort of $O(n^2)$, which is unacceptable for larger number of points. A common approach to remedy this problem is to construct a voxel data structure which contains the domain Ω. The voxels form a regular grid of squares respectively cubes with side length r. In general, for the approximation of derivatives, as presented in Chapter 5 and Chapter 6, the radius $r(n)$ is chosen such that the number of points inside one ball of radius r is roughly independent of the total number of points n. Consequently, the number of points in a voxel is independent of n, and thus the number of voxels is proportional to n.

One now constructs lists (or pointers) containing the following information:

1. For each voxel, the indices of points contained in it.
2. For each point, which voxel it is contained in.
3. For each point, which are the points closer by than r.

3.2.1 Generating Lists

The second list requires to loop over all points and computing its voxel. This is of complexity $O(n)$. The first list can be obtained from the second list by sorting with respect to the voxel indices. This is of complexity $O(n \log n)$. Of course, in time dependent problems, one will

generate the lists just once from scratch, and in the following update them from the previous time step.

3.2.2 Finding Neighboring Points

For a point $\mathbf{x}_i$, all points inside the ball $B(\mathbf{x}_i, r)$ need to be found. The first list gives direct access to the voxel this point is contained in. Now all points in the voxel and its 8 (in 2d) respectively 26 (in 3d) neighboring voxels are tested for being inside the ball. The indices of the points in these voxels are given by the second list. Since each voxel contains $O(1)$ points, this operation is of constant effort. Hence, the total complexity of a neighborhood search for every point is $O(n)$. The neighborhood information is saved in the third list.

3.2.3 Removing Close Points

The two closest points can be found in $O(n)$ time by looping over all points and for each point finding its closest neighbor by checking all points in its circular neighborhood. With the same procedure, one can find the k shortest distances, as well as all points closer than a given distance $d_{\min}$. Removing a point is negligible effort. Introducing a new point is of complexity $O(1)$, since it is one interpolation problem using all neighboring points.

3.2.4 Filling Holes

Finding a hole in the point cloud is a more tricky task, since provided is only information where points *are* and not where there are *none*. An important fact is that the center of a void circle of maximal radius must be a Voronoi point. The Voronoi cell [Vor07] of point $\mathbf{x}_i$ is the set of all points closer to $\mathbf{x}_i$ than to any other point. If the points are in general position, $d + 1$ Voronoi cells meet in one point, the Voronoi point.

Finding all Voronoi points in 2d can be done in $O(n \log n)$ time, e.g. by Fortune's plane-sweep algorithm [For87]. In 3d, the Voronoi points can be found in $O(n^2)$ time by converting the problem to finding a 4d convex hull. As for hole filling the Voronoi points are only to be detected locally, subdivision algorithms can reduce the effort to $O(n \log n)$ (see [She99] for details).

Obviously in 3d, but also in 2d, constructing the full Voronoi diagram can be too expensive for a large number of points. This is in some sense satisfying, since the complexity of meshing a point cloud in 3d is one major motivation for putting our focus on meshless methods. If the Voronoi diagram was easy to compute, so would be the Delaunay triangulation.

For the task of identifying holes, the existing voxel (or octree) structure can be used to construct local, partially overlapping, Voronoi diagrams. If the point cloud is not too deformed, this approach successfully identifies the largest holes in $O(n)$ time, since the number of points considered locally is of order $O(1)$.

Once the too largest holes are identified, new points are inserted, which is for each new point of complexity $O(1)$.

3.2.5 Varying Point Density

If the density of points varies within the domain Ω, also the radius of circular neighborhood r has to vary, in order to keep a constant number of points. The above voxel structure can then be replaced by a quadtree (2d) respectively octree (3d) structure, yielding again a total complexity of $O(n \log n)$. Further details on the case of varying neighborhood radius are provided by Kuhnert [Kuh99].

3.3 Point Cloud Generation

For the initial setup, a point cloud needs to be constructed inside Ω and on its boundary. Besides the geometry itself, one may have prescribed a desired density of points (possibly depending on position), a minimal distance of points and a maximal hole size. If all these requirements are given, constructing a desired point cloud directly can be difficult. Among various others, the following approaches have proven useful:

- **Successive insertion**
  Start at the domain boundary, then construct one layer in the inside, and successively insert points till the center is reached. If the domain boundary is defined by points, use these (thin if too dense respectively fill if too coarse). Approach the inside by inserting points with (approximately) desired distance to d points. For complex geometries and restrictive constraints, this approach may require successional thinning and filling, as described below.

- **Quasirandom methods**
  Especially if the desired point density should depend on position, quasirandom approaches, as described by Devroye [Dev86], can yield a good starting cloud. Towards the interior one can use quasiregular point clouds, for instance the lattices described by Sloan and Joe [SJ94]. For complex geometries, however, constructing the boundary points might be complicated. Additional constraints on the point cloud may require thinning or filling, as described in the following.

- **Thinning**
  Construct a too large point cloud without caring about the minimal distance (e.g. by (quasi)randomly constructing points). Then remove (or merge) points as described in Section 2.5.2, until a desired number of minimal distance is reached.

- **Filling**
  Construct a very coarse point cloud. Then insert points into the largest holes, as described in Section 2.5.1.

- **Moving**
  Construct a point cloud by any method. Then move points until the minimal distance and maximum hole size constraints are satisfied.

3.4 Point Cloud Manipulation

The FPM starts with a point cloud which satisfies given requirements, then moves the points according to their fluid velocity. This may violate the conditions, e.g. by points coming too close or by large holes appearing. Hence, one has to constantly manipulate the point cloud, i.e. the task is to change a given point cloud, which in several instances is non-satisfactory, into a point cloud, which satisfies the requirements. This constant *particle management* consists of two aspects: Merging close points and filling large holes. The former is described in Section 2.5.2, the latter is outlined in Section 2.5.1.

3.5 Neighborhood Relations

We consider completely meshless point clouds, i.e. no information is provided aside from the positions of the points and the role the point plays in the partial differential equations being solved (e.g. interior or boundary point). Finite difference methods require neighborhood information, i.e. for each point one has to define which of the other points are its neighbors. In the presence of a grid, the neighborhood relation is defined by the edges. In a meshless set of points, however, there is no straightforward concept of a neighborhood. Before applying a finite difference method, as described in Chapter 4, for every point the neighboring points have to be defined and determined.

The intention of a neighborhood of a point is that the neighboring points shall be incorporated into a finite difference approximation at the point. Obviously, a good neighborhood concept should depend on the equation to be approximated. For instance, for the Poisson equation it is desirable to have neighboring points distributed “nicely around” the central

point (we shall specify this aspect more precisely later). On the other hand, the same concept of a central point having a symmetric neighborhood around itself can be a source of instabilities if a hyperbolic equation is to be solved. If the approximated operator is isotropic (such as the Laplace operator), the neighborhood concept should mimic this property. If, on the other hand, the considered problem shows significant anisotropies, another neighborhood concept might be preferable. **The concept of neighborhood should depend on the equation being solved.**

In this thesis, one main task is the meshless finite difference approximation of the Poisson equation. In Chapter 6 we present a neighborhood concept which is specifically designed for approximating the Laplace operator. As this operator is homogeneous and isotropic, the resulting neighborhood concept will turn out to possess nice geometric properties.

Often times, the neighborhood concept is chosen without specific attention to the equation being solved, but is rather defined solely geometrically. This may lead to success, if the geometric properties are compatible with the operator being approximated.

An important property of neighborhood concepts is given by the following

Definition 3.2. A neighborhood concept is called *symmetric*, if a point $\mathbf{x}_j$ being a neighbor of point $\mathbf{x}_i$ implies point $\mathbf{x}_i$ being a neighbor of point $\mathbf{x}_j$.

Not considering the special role of Dirichlet boundary points, we shall see in Chapter 4 that a symmetric neighborhood relation leads to reciprocal finite difference matrices (see Definition 4.1).

3.5.1 Geometric Neighborhood Concepts

We present the most common neighborhood concepts, which are based on geometric criteria, i.e. can be defined for a point cloud without considering the equation being solved. Figures 3.8, 3.9, 3.11, 3.12, and 3.13 show one and the same point cloud with different neighborhood concepts applied to it.

Circular Neighborhood

The setup is shown in Figure 3.8. One prescribes a radius r. To a point $\mathbf{x}_i$ the neighborhood is defined as all points $\mathbf{x}_j \in X$, which satisfy

$$\|\mathbf{x}_j - \mathbf{x}_i\|_2 < r \ . \tag{3.11}$$

The Euclidian norm is isotropic, which is favorable for the Laplace operator being approximated. If the same radius r is chosen for all points, the circular neighborhood concept is

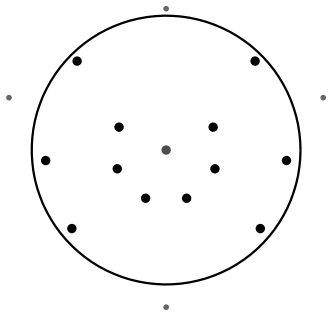

Figure 3.8: Circular neighborhood

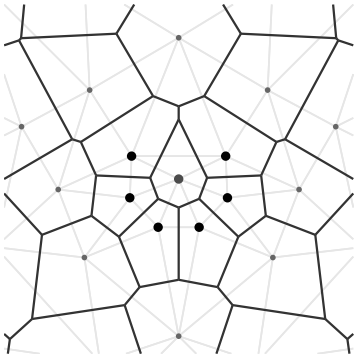

Figure 3.9: Delaunay neighborhood

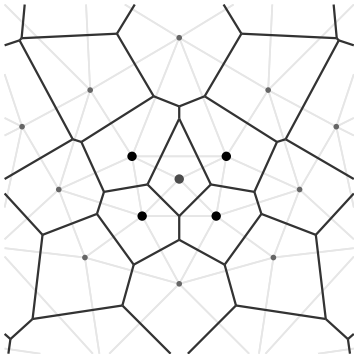

Figure 3.10: Delaunay neighborhood may yield too few neighbors

symmetric. If the radius r varies over the domain Ω, the concept is not necessarily symmetric. In this case, one can recover symmetry on the exchange of the neighborhoods not being exact circles anymore. Also note that the circular neighborhood concept does not necessarily guarantee the neighboring points to be distributed nicely around the central point. If, for instance, left of a point a bunch of other points are close by, while other points right of the point are a distance away, the circular neighborhood concept either yields a lot of neighbors or the neighbors all lie on one side.

With a circular neighborhood an appropriate choice of r is big problem. In some sense, the radius plays a similar role as the smoothing length in SPH, as introduced in Section 2.3.1. Hence, ideas known from SPH can be employed for the correct choice of r. Another concept is to prescribe a number m, and always defining the m closest points as neighbors [Lis84]. Thus the radius r would vary and equal the distance of the m^{th} closest point plus ε.

The circular neighborhood is essentially the neighborhood approach in least squares methods, as presented in Chapter 5, since these consider a weight function which depends on the distance only. If this function is compactly supported, the neighborhood is naturally circular.

Delaunay Neighborhood

To the point cloud X the Delaunay triangulation, respectively tetrahedrization, is constructed [She99]. To a point all points connected by edges are defined as neighbors. As outlined in Section 3.2.4, and presented by Shewchuk [She99], this can be done comparably efficient in 2d, while in 3d this construction can become cumbersome. Of course, for the task of finding neighbors, one can limit the maximum distance of neighbors and thus confide in local tesselations. Note also that in 2d the Delaunay triangles have a uniform lower bound on the minimum angle, which is not the case for the 3d Delaunay tetrahedra [She99, p. 21].

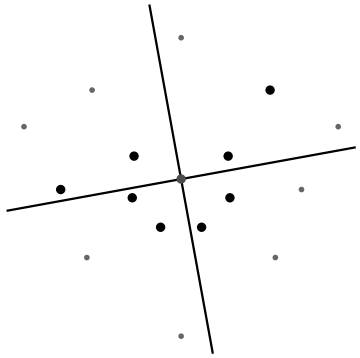

Figure 3.11: Four quadrant neighborhood

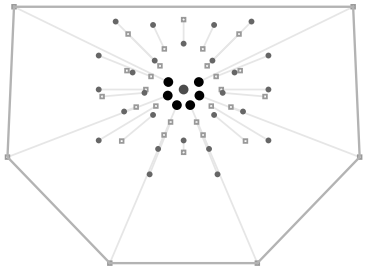

Figure 3.12: Inverse convex hull neighborhood

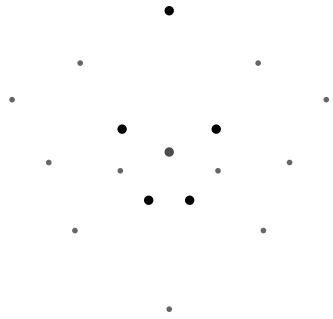

Figure 3.13: Positive stencil neighborhood

However, this fact is more a problem for finite element than for finite difference approximations, as long as the point cloud has a reasonable mesh size and separation, as defined in Section 3.1. Since the approach yields a tesselation, the Delaunay neighborhood concept is obviously symmetric.

Figure 3.9 shows the Delaunay neighborhood, which in this case yields 6 neighbors. These are the closest points, but this need not be the case in general. Figure 3.10 shows the Delaunay neighborhood, when the point cloud is slightly changed. In this case only 4 points are in the neighborhood. The number of neighboring points varies from point to point, and it can be smaller than 5. With respect to approximating the Laplace operator this is a problem, since in 2d at least 5 neighboring points are required for a consistent Laplace approximation (see Section 4.2). One can remedy this problem by detecting these cases and including additional neighbors, this however at the sacrifice of symmetry.

It has to be pointed out that one major motivation for considering meshless methods was that the generation of a mesh may be too expensive. In this case, of course, computing the Delaunay neighborhood will in general not be a good option. Conversely spoken, if one computes a tesselation anyhow, why consider meshless methods? Firstly, the mesh is still irregular, and finite difference approaches still require most of the ingredients presented in the following. Secondly, it may still be advantageous to use a meshless approach for the finite difference approximation, although a mesh is present (since, for instance, dealing solely with a neighborhood relation is significantly easier than dealing with 3d grids). This approach would be a hybrid method, in which a mesh is present, but the approximations are done in a meshless manner.

Four Quadrant Criterion

The *four quadrant criterion* (in 3d *eight sector*) was proposed by Liszka and is presented in [DLT96]. The selection idea is shown in Figure 3.11. In the central point a local cartesian coordinate system is considered, which defines four (2d) respectively eight (3d) sectors. In each such sector, the two points closest to the central point are chosen, yielding for every point a total of 8 respectively 16 neighbors. The construction guarantees the neighboring points to be in some sense distributed around the central points. The method is not completely meshless, since there is a slight dependence on the chosen coordinate system, which points are chosen as neighbors.

Inverse Convex Hull Neighborhood

If the task is to select neighbors which are distributed nicely around a central point, the convex hull of a point cloud is a useful construction. Unfortunately, the vertex points of a convex hull are typically the ones furthest away. However, reasonable neighborhood points should be nearby. Liszka communicated the following idea [Lis06]. For a central point $\mathbf{x_i}$, one considers it as the origin and mirrors all other points at the unit circle, i.e. the point $\mathbf{x_j}$ is moved to $\mathbf{z_j} = \mathbf{x_i} + \frac{\mathbf{x_j}-\mathbf{x_i}}{\|\mathbf{x_j}-\mathbf{x_i}\|^2}$. The convex hull to the new set of points $\mathbf{z}_i$ is constructed. The original points, whose mirrored points span the convex hull, are defined as neighbors to $\mathbf{x_i}$. Figure 3.12 shows the construction. The grey boxes denote the mirror images of the original points. The points spanning the convex hull are close to the central point. However, due to the properties of the convex hull, not only distance but also position of the points is important. Observe for example that the point exactly north of the central point would become a neighbor if it was moved slightly closer. The inverse convex hull neighborhood can yield any number of neighbors larger or equal $d + 1$. This might not be enough points, as for the Delaunay neighborhood.

Obviously, one will never construct the convex hull for all mirrored points, but only incorporate points close enough to have at least a chance to form the envelope. Still, the construction is fairly elaborate. It is unclear, whether this concept of neighborhood can be obtained in an inexpensive manner.

Positive Stencil Neighborhood

Figure 3.13 shows the stencil obtained with the minimal positive stencil method, which Chapter 6 is devoted to. As opposed to the preceding approaches it is not constructed by geometric means, but by formulating a minimization problem. For the moment note the particular choice of neighbors. The point is the north is preferred over the two much closer

points in the southwest and southeast. As we shall see later, the positive stencil approach selects independently for each point a minimum number of neighbors, hence the approach is not symmetric. In Section 6.9 we revisit the neighborhood concepts with respect to positivity of the arising stencils.

Chapter 4

Meshless Finite Differences for the Poisson Equation

In this chapter we outline how the Poisson equation can be discretized on a point cloud using a meshless finite difference approach. We derive the conditions for a consistent approximation and present how to set up the approximating linear system. The subsequent chapters are devoted to the specific computation of the matrix entries.

Consider the Poisson equation to be solved inside a suitable domain $\Omega \subset \mathbb{R}^d$

$$\begin{cases} \Delta u = f & \text{in } \Omega \\ u = g & \text{on } \Gamma_D \\ \dfrac{\partial u}{\partial n} = h & \text{on } \Gamma_N \end{cases} \tag{4.1}$$

where $\Gamma = \Gamma_D \cup \Gamma_N = \partial\Omega$. Let an arbitrary point cloud $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n\} \subset \overline{\Omega}$ be given, which consists of interior points $X_i = \{\mathbf{x}_1, \dots, \mathbf{x}_{n_i}\} \subset \Omega$ and boundary points $X_b = \{\mathbf{x}_{n_i+1}, \dots, \mathbf{x}_n\} \subset \partial\Omega$. The point cloud is considered to be purely meshless, i.e. no information about connection of points is provided. This meshless Poisson problem is an archetype problem in meshless methods, and various approaches apply to it, as outlined in Section 2.2.2. We are interested in the generalized finite difference method (GFDM), as presented by Liszka, et. al. [LO80, DKL84].

4.1 Matrix Generation

We use finite difference approaches to convert the above problem into a finite dimensional linear system

$$A\hat{\mathbf{u}} = \hat{\mathbf{f}}, \tag{4.2}$$

where the vector $\hat{\mathbf{u}}$ contains approximations to the values $u(\mathbf{x}_i)$. In finite difference methods, each row of the matrix A corresponds to a point $\mathbf{x}_i$. The entries in the i-th row are formed by a *stencil*, i.e. coefficients, which, taken as weights for a linear combination of function values, approximate the corresponding differential operator. In particular, for any interior point $\mathbf{x}_i \in X_i$, the corresponding row is given by a *Laplace stencil*, i.e. the Laplace operator at the point $\mathbf{x}_i$ is approximated. For a Neumann boundary point $\mathbf{x}_i \in X_b$, a stencil approximating the normal derivative is constructed. For a Dirichlet boundary point $\mathbf{x}_i \in X_b$ the corresponding stencil is the trivial identity stencil. Note that Dirichlet boundary points yield an identity submatrix and could thus be easily be shifted to the right hand side, yielding a smaller system $A\hat{\mathbf{u}} = \hat{\mathbf{f}}$. However, in order to have equally sized matrices $A \in \mathbb{R}^{n\times n}$ always, we consider Dirichlet points to be also incorporated into the system matrix.

The computation of stencil values is based on approximating derivatives of a smooth function, whose values are given at scattered data points. In Section 4.2 we derive a set of constraints for each point in the point cloud. These constraints need to be satisfied by the stencil entries. Formally, one could construct a stencil which uses every point in the point cloud for approximating a derivative. Obviously, this would not be a smart thing to do, since firstly the matrix A would have unnecessarily many nonzero entries, and secondly only points in the neighborhood (as outlined in Section 3.5) of a central point should constitute to an approximation of a derivative. In general, one chooses more neighboring points than constraints, so additional criteria are required to single out a unique stencil.

Let us for the moment assume that for any point $\mathbf{x}_i$ a stencil $\mathfrak{s}^{(i)}$ is obtained. Then the resulting system matrix A is constructed by inserting the stencils as rows, i.e. the entries are $a_{ij} = \mathfrak{s}_j^{(i)}$. Since for every point only a small amount of neighboring points is considered, the resulting finite difference matrix A will be sparse. The right hand side $\hat{\mathbf{f}}$ contains the values $f(\mathbf{x}_i)$ for every interior point, and the values $g(\mathbf{x}_i)$ respectively $h(\mathbf{x}_i)$ for boundary points.

Many approaches for the computation of the stencils at the scattered data points employ ideas originating in scattered data interpolation. A fundamental example are least squares method, as visited in more detail in Chapter 5. The minimal positive stencil method, as presented in Chapter 6, is another approach enforcing positive stencils and thus yielding an M-matrix structure.

4.1.1 Matrix Graph and Connectivity

Criteria for an M-matrix structure require to consider the connectivity of the system matrix A. Let us assume each row (and corresponding column) is associated with a value from an index set I.

Definition 4.1. Let $A \in \mathbb{R}^{I\times I}$ be a square matrix with respect to the index set I. The

graph of the matrix $G(A)$ is the subset of $I \times I$, defined by

$$G(A) = \{(i,j) \in I \times I : a_{ij} \neq 0\} \ . \tag{4.3}$$

Definition 4.2. A graph G is called *symmetric*, if with $(i,j) \in G$ also $(j,i) \in G$. A matrix with symmetric graph is called *reciprocal*.

For sparse matrices, reciprocity means symmetrically structured. Obviously, every symmetric matrix is reciprocal, but not vice versa. As we are confronted with non-symmetric matrices, reciprocity is a concept of interest.

Definition 4.3. For a matrix $A \in \mathbb{R}^{I \times I}$, the index $i \in I$ is called *connected* to index $j \in I$, if a chain $i = i_0, i_1, \ldots, i_{k-1}, i_k = j \ \in I$ exists, such that

$$(i_{\nu-1}, i_\nu) \in G(A) \quad \forall \nu = 1, \ldots, k \ . \tag{4.4}$$

Note that for non-symmetric matrices, i being connected to j does not imply j being connected to i. For a matrix which arises from a finite difference discretization, connectivity translates into the following: A point i is connected with a point j, if i depends at least indirectly on j via nonzero stencil entries.

Definition 4.4. A matrix $A \in \mathbb{R}^{I \times I}$ is denoted *irreducible*, if every $i \in I$ is connected to every $j \in I$, otherwise *reducible*.

Obviously, Dirichlet boundary points are not connected to any other point, hence preserving Dirichlet boundary points in the system matrix always yields a reducible matrix. One could remove the Dirichlet boundary points from the system. For the criteria for an M-matrix structure, however, it is sufficient to require every point to be connected with a Dirichlet boundary point. In order to generalize the concept to system matrices with Dirichlet rows retained in the matrix, we define

Definition 4.5. Let $A \in \mathbb{R}^{I \times I}$ be a matrix arising from a finite difference approximation of problem (4.1). Let $I_D \subset I$ be the set of indices of the Dirichlet points. We call a matrix *essentially irreducible*, every $i \in I$ is connected to a $j \in I_D$.

In other words, a matrix is essentially irreducible, iff every interior point and Neumann boundary point is at least indirectly connected via stencil entries to a Dirichlet boundary point. In Section 6.1.1 we show that the condition of essential irreducibility yields essential diagonal dominance for the arising finite difference matrices, which then is sufficient for an L-matrix to be an M-matrix.

4.2 Consistent Derivative Approximation

Consider a smooth function u be defined on a domain $\Omega \subset \mathbb{R}^d$. The Laplacian applied to the the function at a point $\mathbf{x_0} \in \Omega$ shall be approximated by using a finite number of function values in the neighborhood of $\mathbf{x}_0$.

Let $u \in C^2(\Omega, \mathbb{R})$ and analytic in $\mathbf{x}_0$. We consider a circular neighborhood

$$U(\mathbf{x}_0) = B(\mathbf{x}_0, r) = \{\mathbf{x} \in \overline{\Omega} : \|\mathbf{x} - \mathbf{x}_0\| < r\} \tag{4.5}$$

with r small enough, such that a Taylor expansion is valid. Other types of neighborhood relations can be considered, as outlined in Section 3.5. Consider points inside the neighborhood $(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_m) \in B(\mathbf{x}_0, r)$. Define the distance vectors $\bar{\mathbf{x}}_i = \mathbf{x}_i - \mathbf{x}_0 \ \forall i = 0, \ldots, m$. The function value at each neighboring point $u(\mathbf{x}_i)$ can be expressed by a Taylor expansion

$$u(\mathbf{x}_i) = u(\mathbf{x}_0) + \nabla u(\mathbf{x}_0) \cdot \bar{\mathbf{x}}_i + \frac{1}{2}\nabla^2 u(\mathbf{x}_0) : \left(\bar{\mathbf{x}}_i \cdot \bar{\mathbf{x}}_i^T\right) + e_i \tag{4.6}$$

Here the colon denotes the matrix scalar product $A : B = \sum_{i,j} A_{ij} B_{ij}$, and e_i is the error in the expansion, which is of order $e_i = O(r^3)$. A linear combination with coefficients $(\mathfrak{s}_0, \ldots, \mathfrak{s}_m)$ equals

$$\begin{aligned} \sum_{i=0}^{m} \mathfrak{s}_i u(\mathbf{x}_i) = u(\mathbf{x}_0) \left(\sum_{i=0}^{m} \mathfrak{s}_i\right) + \nabla u(\mathbf{x}_0) \cdot \left(\sum_{i=1}^{m} \mathfrak{s}_i \bar{\mathbf{x}}_i\right) \\ + \nabla^2 u(\mathbf{x}_0) : \left(\frac{1}{2} \sum_{i=1}^{m} \mathfrak{s}_i \left(\bar{\mathbf{x}}_i \cdot \bar{\mathbf{x}}_i^T\right)\right) + \left(\sum_{i=1}^{m} \mathfrak{s}_i e_i\right) \end{aligned} \tag{4.7}$$

Expression (4.7) shall approximate the Laplacian at the central point

$$\sum_{i=0}^{m} \mathfrak{s}_i u(\mathbf{x}_i) = \Delta u(\mathbf{x}_0) + O(r^3) \tag{4.8}$$

This is satisfied, if the following constraints hold

$$\sum_{i=0}^{m} \mathfrak{s}_i = 0 \tag{4.9}$$

$$\sum_{i=1}^{m} \bar{\mathbf{x}}_i \mathfrak{s}_i = 0 \tag{4.10}$$

$$\sum_{i=1}^{m} \left(\bar{\mathbf{x}}_i \cdot \bar{\mathbf{x}}_i^T\right) \mathfrak{s}_i = 2I \tag{4.11}$$

Here (4.11) follows, since $\nabla^2 u(\mathbf{x}_0) : I = \Delta u(\mathbf{x}_0)$. For a method to satisfy (4.8) for any choice of r, it is necessary for the constraints (4.9), (4.10) and (4.11) to be satisfied. Hence we define

Definition 4.6. A Laplace stencil $(\mathfrak{s}_0, \ldots, \mathfrak{s}_m)$ to a given set of points $(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_m) \in B(\mathbf{x}_0, r)$ is called *consistent*, if the constraints (4.9), (4.10) and (4.11) are satisfied.

Constraints (4.10) and (4.11) can be formulated as a linear system of equations

$$V \cdot \boldsymbol{\mathfrak{s}} = \Delta\mathbf{b} \tag{4.12}$$

where $V \in \mathbb{R}^{k \times m}$ is the Vandermonde matrix given by $\bar{\mathbf{x}}_1, \ldots, \bar{\mathbf{x}}_m$, and $\boldsymbol{\mathfrak{s}} \in \mathbb{R}^m$ is the sought vector of weights. In 2d the system reads as

$$V = \begin{pmatrix} \bar{x}_1 & \ldots & \bar{x}_m \\ \bar{y}_1 & \ldots & \bar{y}_m \\ \bar{x}_1\bar{y}_1 & \ldots & \bar{x}_m\bar{y}_m \\ \bar{x}_1^2 & \ldots & \bar{x}_m^2 \\ \bar{y}_1^2 & \ldots & \bar{y}_m^2 \end{pmatrix}, \quad \Delta\mathbf{b} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 2 \\ 2 \end{pmatrix} \tag{4.13}$$

Here the vector in component notation is $\bar{\mathbf{x}}_i = (\bar{x}_i, \bar{y}_i)$. In Chapter 5 we will motivate the notation $\Delta\mathbf{b}$ for the right hand side vector. The central weight $\mathfrak{s}_0$ is then given by (4.9) as

$$\mathfrak{s}_0 = -\sum_{i=1}^{m} \mathfrak{s}_i \ . \tag{4.14}$$

The number of constraints is

$$k = \frac{d(d+3)}{2} \ . \tag{4.15}$$

In case the number of neighboring points equals the number of constraints (i.e. $m = k$), the stencil is uniquely defined by $\boldsymbol{\mathfrak{s}} = V^{-1} \cdot \Delta\mathbf{b}$, given the V is regular (see Section 4.2.3 on this assumption). The linear system can be solved for instance by Gaußian elimination. Alternatively, one can use explicit formulae for the determinant of a multivariate Vandermonde matrix, as derived by Lorentz [Lor90, Lor92].

If on the other hand more neighboring points are considered than constraints are to be satisfied (i.e. $m > k$), then system (4.12) is underdetermined and will in general have infinitely many solutions. A unique stencil can be selected either by formulating a minimization problem (as in Section 5.1.3) or by deriving the stencil from an approximation method, which guarantees the stencil to be consistent (as done in Section 5.2).

In a similar manner as for the Laplace operator one can employ a Taylor expansion to derive a linear system of constraints for the approximation of first derivatives. The expression

$$\sum_{i=0}^{m} \mathfrak{s}_i u(\mathbf{x}_i) \tag{4.16}$$

approximates the directional derivative $\frac{\partial}{\partial \mathbf{n}}$ at the point $\mathbf{x}_0$, if the constraints

$$\sum_{i=0}^{m} \mathfrak{s}_i = 0 \tag{4.17}$$

$$\sum_{i=1}^{m} \bar{\mathbf{x}}_i \mathfrak{s}_i = \mathbf{n} \tag{4.18}$$

are satisfied. For instance, in 1d the resulting system for the approximation of $u'(x_0)$ reads as

$$\begin{pmatrix} \bar{x}_1 & \dots & \bar{x}_m \end{pmatrix} \cdot \mathfrak{s} = 1 \ , \tag{4.19}$$

and the central weight $\mathfrak{s}_0$ is obtained by relation (4.14).

4.2.1 Positive Stencils

In general, one chooses more neighboring points than constraints to be satisfied. Hence, there is for each point some freedom which particular stencil to choose. In the following chapters we present various methods on how the particular stencil is chosen. These single out a unique stencil by formulating a minimization problem. There are various properties which are desirable for stencils to satisfy. For instance, the stencil entries should not be too large in absolute value (which gives rise to the minimization formulations presented in Section 5.1.3 and Chapter 6). A good property for a stencil to satisfy, which is one focus in this thesis, is positivity.

Definition 4.7. A stencil $\mathfrak{s}$ is called *positive*, if either all neighboring entries $\mathfrak{s}_1, \dots, \mathfrak{s}_m$ are non-positive or all are non-negative, and if the central entry $\mathfrak{s}_0$ is of opposing sign.

Example 4.1. Consider in 1d the points $X = (0, -1, 1, 2)$, where $x_0 = 0$ is the central point. One can easily verify that all three stencils $\mathfrak{s}_a = (-2, 1, 1, 0)$, $\mathfrak{s}_b = (-\frac{3}{4}, -\frac{1}{2}, 0, \frac{1}{4})$ and $\mathfrak{s}_c = (3, -1, -3, 1)$ are consistent approximations to the second derivative. Stencil $\mathfrak{s}_a$ is positive, the other two are not. Of the two non-positive stencils, $\mathfrak{s}_b$ is preferable, since it has at least a negative central entry. Stencil $\mathfrak{s}_c$ has a positive central entry. A stencil with a positive central entry can never be positive, since relation (4.9) cannot be satisfied.

If a positive stencil is obtained for every point, then the resulting system matrix is an L-matrix, as defined in Section 6.1. In Chapter 6 we introduce an approach which guarantees positive stencils.

4.2.2 Higher Order Approximations

The constraints derived in Section 4.2 guarantee a consistent approximation to the corresponding derivatives, i.e. the approximations are at least first order accurate. Indeed, for

irregular point distributions, the approximation will in general only be of first order, if only the consistency constraints are imposed.

One can in principle obtain higher order approximations to the derivatives by enforcing higher order terms in the Taylor expansion to cancel out. For instance, a second order accurate approximation to the Laplace operator in 2d needs to satisfy the constraint (4.9) for the central point, and additionally the linear system $V \cdot \mathfrak{s} = \Delta\mathbf{b}$ with

$$V = \begin{pmatrix} \bar{x}_1 & \dots & \bar{x}_m \\ \bar{y}_1 & \dots & \bar{y}_m \\ \bar{x}_1^2 & \dots & \bar{x}_m^2 \\ \bar{x}_1\bar{y}_1 & \dots & \bar{x}_m\bar{y}_m \\ \bar{y}_1^2 & \dots & \bar{y}_m^2 \\ \bar{x}_1^3 & \dots & \bar{x}_m^3 \\ \bar{x}_1^2\bar{y}_1 & \dots & \bar{x}_m^2\bar{y}_m \\ \bar{x}_1\bar{y}_1^2 & \dots & \bar{x}_m\bar{y}_m^2 \\ \bar{y}_1^3 & \dots & \bar{y}_m^3 \end{pmatrix}, \quad \Delta\mathbf{b} = \begin{pmatrix} 0 \\ 0 \\ 2 \\ 0 \\ 2 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{4.20}$$

needs to be satisfied. If the Taylor expansion up to order l is considered, one obtains in d space dimensions altogether

$$k + 1 = \binom{l+d}{d} . \tag{4.21}$$

constraints, including the constant constraint (4.9). In 3d, one has $k = 19$ for second order, and $k = 34$ for third order. Since for points in general position at least as many neighboring points as constraints are required, this approach will not constitute in an efficient method for high order approximations. Additionally, it is well known from the 1d case that higher order conditions lead to oscillations in the stencil entries, i.e. one has both positive and negative entries, which significantly worsens stability. In particular, one can easily see that incorporating fourth other terms into the Vandermonde matrix will make it impossible for the resulting stencils to be positive. If the stencil satisfies the fourth order constraints, then it satisfies in particular

$$\sum_{i=1}^{m} \left(\bar{x}_i^4 + \bar{y}_i^4\right) \mathfrak{s}_i = 0 , \tag{4.22}$$

which can only be satisfied by $\mathfrak{s} = 0$, which is inconsistent.

Hence, higher order is incompatible with positive stencils. One major interest of this thesis lies in positive stencils. The whole Chapter 6 is devoted on how to enforce and guarantee positive stencils, i.e. stencils with entries of one and the same sign aside from the central point. In the context of meshless finite difference methods, higher order can be obtained by deferred correction approaches, as described by Orkisz and Milewski [OM05]. In general, however, meshless finite difference approaches are not the most favorable methods for high

order approximations to the Poisson equation. Such a task can be better achieved, for instance, by *h*-*p*-cloud methods [DO96].

Of a certain interest can be a second order approximation to the normal derivative at Neumann boundary points. In this case the constraints (4.17) and (4.18) will be complemented by the additional set of constraints

$$\sum_{i=1}^{m} \left(\bar{\mathbf{x}}_i \cdot \bar{\mathbf{x}}_i^T \right) \mathfrak{s}_i = 0 \ . \tag{4.23}$$

We will show in Section 6.5 that this condition is incompatible with positive stencils.

4.2.3 Special Configurations

In general at least as many points as constraints are required, and if more points than constraints are present, there will be infinitely many solutions to the system (4.12). However, if the points form specific configurations, both above statements can be false: Less points than constraints *can* yield a solution (Example 4.2), and enough points can yield *no* (Example 4.3) or *multiple* (Example 4.4) solutions.

In most meshless applications such special configurations are extremely unlikely to happen. For instance in the FPM the points are moving with the fluid velocity and thus might come close to the below configurations, but will not meet them exactly. Note that slightly moving the points in the given configurations will recover the setup which is true in general, i.e. no solution in Example 4.2, and exactly one solution in Examples 4.3 and 4.4. Additionally, special configurations of all points become even less likely by considering significantly more points than constraints.

For the generation of positive Laplace stencils (see Chapter 6) we will impose much stricter geometric conditions than merely avoiding the specific setups given in this section. However, when comparing different approaches to generate stencils, the special cases considered here give insight into various properties. In general, one prefers meshless methods to generate well known finite difference stencils in the case of regular grids.

Example 4.2 (Fewer points than constraints, still solution exists)**.** Consider 5 points given by $\mathbf{x}_0 = (0,0)$, $\mathbf{x}_1 = (1,0)$, $\mathbf{x}_2 = (0,-1)$, $\mathbf{x}_3 = (-1,0)$, $\mathbf{x}_4 = (0,1)$, as shown in Figure 4.1. The corresponding linear system is

$$\left(\begin{array}{cccc|c} 1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 2 \\ 0 & 1 & 0 & 1 & 2 \end{array}\right) \tag{4.24}$$

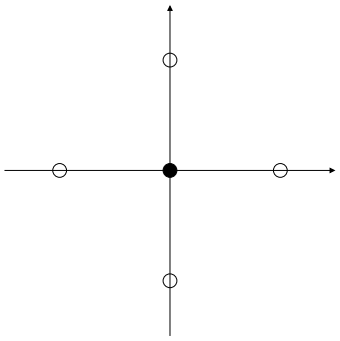

Figure 4.1: Solution with 5 points

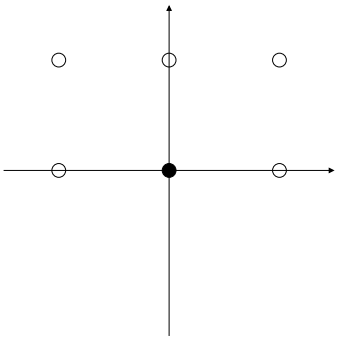

Figure 4.2: No solution with 6 points

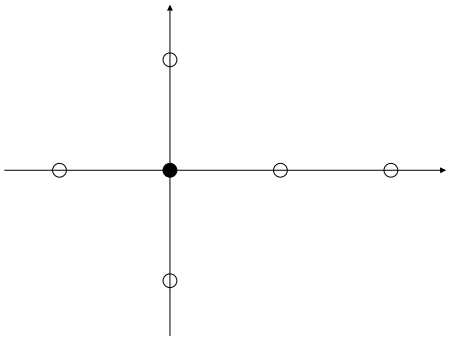

Figure 4.3: Multiple solutions with 6 points

Due to the particular point positions, the standard 5-point stencil $(-4, 1, 1, 1, 1)$ for regular grids satisfies the consistency constraints, although fewer points than constraints are present. In general, however, having fewer points available than constraints will make it impossible to obtain a consistent stencil.

Example 4.3 (Enough points, but no solution)**.** Consider 6 points given by $\mathbf{x}_0 = (0, 0)$, $\mathbf{x}_1 = (1, 0)$, $\mathbf{x}_2 = (1, 1)$, $\mathbf{x}_3 = (0, 1)$, $\mathbf{x}_4 = (-1, 1)$, $\mathbf{x}_5 = (-1, 0)$, as shown in Figure 4.2. The corresponding linear system is

$$\left(\begin{array}{ccccc|c} 1 & 1 & 0 & -1 & -1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 2 \\ 0 & 1 & 1 & 1 & 0 & 2 \end{array}\right) \tag{4.25}$$

Although enough points are available, this system has no solution (the second and fifth row contradict). In y-direction only two values are available ($y = 0$ and $y = 1$), which does not admit an approximation to the second derivative. In the language of Section 6.4.3 the neighboring points are not distributed well around the central point.

Example 4.4 (Exact number of points, still infinitely many solution)**.** Consider 6 points given by $\mathbf{x}_0 = (0, 0)$, $\mathbf{x}_1 = (1, 0)$, $\mathbf{x}_2 = (0, -1)$, $\mathbf{x}_3 = (-1, 0)$, $\mathbf{x}_4 = (0, 1)$, $\mathbf{x}_5 = (2, 0)$, as shown in Figure 4.3. The corresponding linear system is

$$\left(\begin{array}{ccccc|c} 1 & 0 & -1 & 0 & 2 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 4 & 2 \\ 0 & 1 & 0 & 1 & 0 & 2 \end{array}\right) \tag{4.26}$$

The linear system is singular, since the third row is zero. The system has solutions, since

$\mathbf{b} \in R(B)$. The solutions are of the form

$$\mathfrak{s} = \begin{pmatrix} 1 - 3\mathfrak{s}_5 \\ 1 - \mathfrak{s}_5 \\ 1 \\ 1 \\ \mathfrak{s}_5 \end{pmatrix} \tag{4.27}$$

In this case we would like uniqueness criteria to single out the well known 5-point stencil $(-4, 1, 1, 1, 1)$ for regular grids, which is obtained for $\mathfrak{s}_5 = 0$. We will show in Section 6.3.3 that for this setup the new approach of minimal positive stencils will indeed select the desired stencil. On the other hand, least squares methods, as presented in Chapter 5 will in general yield a stencil containing all points.

4.3 General Second Order Elliptic Problems

The presented approach can be generalized to more general second order elliptic problems.

$$\begin{cases} Lu = f & \text{in } \Omega \\ u = g & \text{on } \Gamma_D \\ \dfrac{\partial u}{\partial n} = h & \text{on } \Gamma_N \end{cases} \tag{4.28}$$

Here L is an elliptic differential operator, defined as

$$Lu = A : \nabla^2 u + \mathbf{b} \cdot \nabla u + cu \ . \tag{4.29}$$

As presented in [Eva98] we assume A to be symmetric and elliptic. We further assume the coefficients A, $\mathbf{b}$ and c to be constant or at least smooth and varying slower than the considered neighborhood radius r. Then a finite difference approximation of L is possible. We employ the same Taylor expansions as derived in Section 4.2, yielding expression (4.7) for a linear combination with neighboring points. Now, of course, expression (4.7) shall approximate the elliptic operator at the central point

$$\sum_{i=0}^{m} \mathfrak{s}_i u(\mathbf{x}_i) = Lu(\mathbf{x}_0) + O(r^3) \tag{4.30}$$

This is satisfied, if the following constraints hold

$$\sum_{i=0}^{m} \mathfrak{s}_i = c \tag{4.31}$$

$$\sum_{i=1}^{m} \bar{\mathbf{x}}_i \mathfrak{s}_i = \mathbf{b} \tag{4.32}$$

$$\sum_{i=1}^{m} \left(\bar{\mathbf{x}}_i \cdot \bar{\mathbf{x}}_i^T \right) \mathfrak{s}_i = 2A \tag{4.33}$$

Hence, we obtain a linear system with the same matrix as for the Laplace operator, i.e. in 2d the matrix is also given by (4.13). However, the right hand side vector differs. In 2d, it now is $(b_x, b_y, 2a_{11}, 2a_{12}, 2a_{22})^T$. The central entry now computes as

$$\mathfrak{s}_0 = c - \sum_{i=1}^{m} \mathfrak{s}_i \ . \tag{4.34}$$

Obviously, the special configuration presented in Section 4.2.3 can yield different results for a different operator. Since the Vandermonde matrix V does not change when going from Δ to L, for points in general position (i.e. when the Vandermonde matrix has full rank), the solution methods presented in the following for the Laplace operator can be applied in the same manner for the general elliptic operator.

The Laplace operator is homogeneous and isotropic. An general operator L can be far from these nice properties, causing problems when approaches tailored to the Laplacian are applied. Also, aspects about the signs of the stencil entries change. For instance, a $c > 0$ can shift a central weight $\mathfrak{s}_0$, which would be nicely negative for the Laplacian, to a positive value.

Chapter 5

Least Squares Methods

In this chapter we investigate methods to approximate derivatives, which are based on a weighted least squares approximation. There are various derivations and interpretations. However, as we will see, common methods which formulate a quadratic minimization problem boil down either to the local least squares or to the moving least squares method for approximating derivatives. These two approaches are not the same, although in specific cases they yield identical results. In most cases, however, the obtained approximations differ. Among both methods there is always an approximating as well as an interpolating approach. We derive the approaches and compare them and show equivalences. A large focus will be on specific cases and one dimensional situations, which provide a good inside into problems with various approximation approaches. Of particular interest are approximations which yield positive stencils. We compare various least squares methods with respect to this aspect.

5.1 Local Least Squares Method

Let a smooth function u be given on a domain $\Omega \subset \mathbb{R}^d$. Consider a central point $\mathbf{x}_0$ and neighboring points $(\mathbf{x}_1, \dots, \mathbf{x}_m)$. The function values $u(\mathbf{x}_i)$ on the points are given. Sought is an approximation to derivatives in the central point.

The local least squares method best approximates the function data by a polynomial in a weighted least squares sense and assigns the polynomial's derivatives as approximations to the derivatives of the function u.

Let P_ν denote the vector space of polynomials of order up to ν. Let $(b_0, \dots, b_k)$ be a basis

of this vector space. A canonical basis of P_1 in 1d is

$$\begin{aligned} b_0(x) &= 1 \\ b_1(x) &= x \end{aligned} \tag{5.1}$$

A canonical basis of P_2 in 2d would be

$$\begin{aligned} b_0(x,y) &= 1 \\ b_1(x,y) &= x \\ b_2(x,y) &= y \\ b_3(x,y) &= x^2 \\ b_4(x,y) &= xy \\ b_5(x,y) &= y^2 \end{aligned} \tag{5.2}$$

Any polynomial $p \in P_\nu$ can be written as

$$p(\mathbf{x}) = \mathbf{a}^T \cdot \mathbf{b}(\mathbf{x}) \ , \tag{5.3}$$

where $\mathbf{a}$ is the vector of coefficients and $\mathbf{b} = (b_0(\mathbf{x}), \ldots, b_k(\mathbf{x}))^T$.

5.1.1 Approximating Local Least Squares

One minimizes the functional

$$F(\mathbf{a}) = \sum_{i=0}^{m} w_i (p(\mathbf{x}_i) - u(\mathbf{x}_i))^2 \ . \tag{5.4}$$

Points closer to the central point should contribute more to the sought value, hence the weights are chosen to depend on the distance to the central point

$$w_i = w(\|\mathbf{x}_i - \mathbf{x}_0\|) \ , \tag{5.5}$$

and the distance weight $w(d)$ function should decay sufficiently fast with d increasing. Note that at this point one is completely free in the choice of the distance weight function. It need not be smooth, nor is the support or the behavior for small distances of importance, if solely one specific points setup is considered. However, in various instances, the LLS method will equal the MLS method, hence the considerations about the distance weight function given in Section 5.2 will be of importance.

Necessary conditions for a minimizer of (5.4) are

$$\frac{dF}{d\mathbf{a}} = 2 \sum_{i=0}^{m} (p(\mathbf{x}_i) - u(\mathbf{x}_i)) w_i \mathbf{b}(\mathbf{x}_i) = 0 \ , \tag{5.6}$$

which leads to the normal equations

$$\sum_{i=0}^{m} \mathbf{b}(\mathbf{x}_i) w_i \mathbf{b}(\mathbf{x}_i)^T \cdot \mathbf{a} = \sum_{i=0}^{m} \mathbf{b}(\mathbf{x}_i) w_i \mathbf{u} \ . \tag{5.7}$$

These are in matrix notation

$$\left(VWV^T\right) \mathbf{a} = (VW)\, \mathbf{u} \ , \tag{5.8}$$

where

$$V = \begin{pmatrix} b_0(\mathbf{x}_1) & \dots & b_0(\mathbf{x}_m) \\ \vdots & \ddots & \vdots \\ b_k(\mathbf{x}_1) & \dots & b_k(\mathbf{x}_m) \end{pmatrix} \in \mathbb{R}^{(k+1)\times m} \tag{5.9}$$

and

$$W = \begin{pmatrix} w_0 & & \\ & \ddots & \\ & & w_m \end{pmatrix} \in \mathbb{R}^{(m+1)\times(m+1)} \ . \tag{5.10}$$

The polynomial p minimizing (5.4) is smooth, and thus derivatives up to order ν can be calculated. For instance, for the 2d quadratic basis above, the Laplace operator applied to a polynomial $p(\mathbf{x}) = \mathbf{a}^T \cdot \mathbf{b}(\mathbf{x})$ is obtained by

$$\Delta p(\mathbf{x}) = \Delta \mathbf{b}^T \cdot \mathbf{a} \ , \tag{5.11}$$

where $\Delta \mathbf{b} = (0, 0, 0, 2, 0, 2)^T$. Solving system (5.8), and substituting into (5.11) yields the Laplace stencil

$$\mathfrak{s} = \left(WV^T\right) \left(VWV^T\right)^{-1} \Delta \mathbf{b} \ , \tag{5.12}$$

which forms the LLS approximation to the Laplacian in the point $\mathbf{x}_0$. The matrix is

$$V = \begin{pmatrix} 1 & \dots & 1 \\ x_1 & \dots & x_m \\ y_1 & \dots & y_m \\ x_1^2 & \dots & x_m^2 \\ x_1 y_1 & \dots & x_m y_m \\ y_1^2 & \dots & y_m^2 \end{pmatrix} \ , \tag{5.13}$$

where $\mathbf{x}_i = (x_i, y_i)^T$.

With the linear basis (5.1) in 1d one can obtain an approximation to the first derivative by

$$p'(x) = \left(\mathbf{b}'\right)^T \cdot \mathbf{a} \ , \tag{5.14}$$

where $\mathbf{b}' = (0, 1)^T$. In the same manner as for the Laplace operator we obtain a stencil

$$\mathfrak{s} = \left(WV^T\right) \left(VWV^T\right)^{-1} \mathbf{b}' \ , \tag{5.15}$$

which approximates $u'(x_0)$. Here the matrix is

$$V = \begin{pmatrix} 1 & \dots & 1 \\ x_1 & \dots & x_m \end{pmatrix} . \tag{5.16}$$

Remark 5.1. Note that the matrix (5.13) depends on the chosen basis. The solution, however, is independent of the basis, as long as the space P_ν is one and the same. When shifting the origin to a different position, matrix (5.13) changes. In comparison, in the consistency system (4.13) the entries were taken relatively to the point of consideration $\mathbf{x}_0$. However, one can easily transform the ALLS system with matrix (5.13) and right hand side $\Delta\mathbf{b}$ into (4.13) and vice versa: Subtracting x_0 and y_0 times the first (constant) row from the linear rows shifts the origin in these. Consequently, subtracting $2x_0$ times the linear and adding x_0^2 times the constant row from and to the quadratic row with x_0^2 (similarly for y_0^2 and x_0y_0) shifts the origin in those.

5.1.2 Interpolating Local Least Squares

The LLS approach presented above uses a weight w_0 for the central point. When increasing w_0, the polynomial minimizing (5.4) will approach the function value of u at the point $\mathbf{x}_0$. This gives rise to modifying the above approach by minimizing functional (5.4) only over the affine subspace of polynomials in P_ν which additionally satisfy $p(\mathbf{x}_0) = u(\mathbf{x}_0)$.

Instead of the basis (5.2) we choose a basis which satisfies

$$b_i(\mathbf{x}_0) = 0 \quad \forall i = 1, \dots, k \ , \tag{5.17}$$

e.g. by replacing $\mathbf{x}$ by $\bar{\mathbf{x}} = \mathbf{x} - \mathbf{x}_0$. A derivation analogous to the one above yields the Laplace stencil

$$\bar{\mathfrak{s}} = \left(\bar{W}\bar{V}^T\right)\left(\bar{V}\bar{W}\bar{V}^T\right)^{-1}\Delta\bar{\mathbf{b}} \ , \tag{5.18}$$

where

$$\bar{V} = \begin{pmatrix} b_1(\bar{\mathbf{x}}_1) & \dots & b_1(\bar{\mathbf{x}}_m) \\ \vdots & \ddots & \vdots \\ b_k(\bar{\mathbf{x}}_1) & \dots & b_k(\bar{\mathbf{x}}_m) \end{pmatrix} \in \mathbb{R}^{k\times m} \ , \quad \bar{W} = \begin{pmatrix} w_1 & & \\ & \ddots & \\ & & w_m \end{pmatrix} \in \mathbb{R}^{m\times m} \tag{5.19}$$

and $\Delta\bar{\mathbf{b}} = (0, 0, 2, 0, 2)^T$. According to relation (4.9) the central weight is

$$\mathfrak{s}_0 = \sum_{i=1}^{m} \mathfrak{s}_i \ . \tag{5.20}$$

Note that system (5.18) is of one dimension lower than system (5.12). For low polynomial order this certainly can yield some percent in computation speed. Also, the solution is different, as we will analyze in more detail in Section 5.5.2.

In comparison to the ALLS approach presented in Section 5.1.1, the matrix for the 2d Laplace operator is

$$\bar{V} = \begin{pmatrix} \bar{x}_1 & \dots & \bar{x}_m \\ \bar{y}_1 & \dots & \bar{y}_m \\ \bar{x}_1^2 & \dots & \bar{x}_m^2 \\ \bar{x}_1\bar{y}_1 & \dots & \bar{x}_m\bar{y}_m \\ \bar{y}_1^2 & \dots & \bar{y}_m^2 \end{pmatrix}, \tag{5.21}$$

The matrix for the 1d derivative becomes merely a row vector

$$\bar{V} = \begin{pmatrix} \bar{x}_1 & \dots & \bar{x}_m \end{pmatrix}, \tag{5.22}$$

resulting in system (5.18) to be trivial to solve.

5.1.3 Minimization Formulation

A useful interpretation for the difference between approximating and interpolating approach is provided by formulating constrained minimization problems, which have the desired stencils as solutions.

As derived in Section 4.2, the constraints for a stencil to be consistent are (4.9), (4.10) and (4.11). These can be formulated into a linear system. This system can be expressed in two ways:

1. **Full system**
   One places all constraints into the system. For instance, in 2d the system is given by

$$V = \begin{pmatrix} 1 & 1 & \dots & 1 \\ 0 & \bar{x}_1 & \dots & \bar{x}_m \\ 0 & \bar{y}_1 & \dots & \bar{y}_m \\ 0 & \bar{x}_1\bar{y}_1 & \dots & \bar{x}_m\bar{y}_m \\ 0 & \bar{x}_1^2 & \dots & \bar{x}_m^2 \\ 0 & \bar{y}_1^2 & \dots & \bar{y}_m^2 \end{pmatrix}, \quad \Delta\mathbf{b} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 2 \\ 2 \end{pmatrix} \tag{5.23}$$

   and the solution is the full stencil $\mathfrak{s}$.

2. **Reduced system**
   The constant constraint does not enter the system. The 2d system is given by

$$\bar{V} = \begin{pmatrix} \bar{x}_1 & \dots & \bar{x}_m \\ \bar{y}_1 & \dots & \bar{y}_m \\ \bar{x}_1\bar{y}_1 & \dots & \bar{x}_m\bar{y}_m \\ \bar{x}_1^2 & \dots & \bar{x}_m^2 \\ \bar{y}_1^2 & \dots & \bar{y}_m^2 \end{pmatrix}, \quad \Delta\bar{\mathbf{b}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 2 \\ 2 \end{pmatrix} \tag{5.24}$$

The solution is the reduced stencil $\bar{\mathfrak{s}}$. The value $\mathfrak{s}_0$ is then computed by relation (4.14).

We now consider the two constrained minimization problems

1. **Minimize over all stencil entries**

$$\min \sum_{i=0}^{n} \frac{\mathfrak{s}_i^2}{w_i}, \quad \text{s.t. } V \cdot \mathfrak{s} = \Delta \mathbf{b} \tag{5.25}$$

   This requires a central weight w_0 being assigned.

2. **Minimize over all but the central stencil entry**

$$\min \sum_{i=1}^{n} \frac{\mathfrak{s}_i^2}{w_i}, \quad \text{s.t. } V \cdot \mathfrak{s} = \Delta \mathbf{b} \tag{5.26}$$

   Obviously, one can replace the constraints in (5.26) by the reduced system

$$\bar{V} \cdot \bar{\mathfrak{s}} = \Delta \bar{\mathbf{b}} \tag{5.27}$$

   and compute the central entry $\mathfrak{s}_0$ afterwards. In this sense, problem (5.26) is of exactly the same form as problem (5.25), just of one dimension less.

The solution to the constrained minimization problem is obtained by the method of Lagrange multipliers.

Theorem 5.1. *Assume V has full rank and $w_i > 0$ $\forall i$. Then the minimization problem (5.25) has the unique solution*

$$\mathfrak{s} = WV^T(VWV^T)^{-1} \cdot \Delta \mathbf{b}, \tag{5.28}$$

where $W = \mathrm{diag}(w_i)$.

Proof. We minimize

$$f(\mathfrak{s}) = \sum_{i=0}^{n} \frac{\mathfrak{s}_i^2}{w_i} \tag{5.29}$$

subject to the constraints

$$g_i(\mathfrak{s}) = \sum_{j=0}^{n} v_{ij}\mathfrak{s}_j - b_i = 0. \tag{5.30}$$

Using the method of Lagrange multipliers we know that a solution $\mathfrak{s}$ satisfies the relation

$$\nabla f(\mathfrak{s}) = \sum_{i=1}^{n} \lambda_i \nabla g_i(\mathfrak{s}) \tag{5.31}$$

Here

$$2W^{-1} \cdot \mathfrak{s} = V^T \cdot \boldsymbol{\lambda} \tag{5.32}$$

Hence

$$\mathfrak{s} = \frac{1}{2} W V^T \cdot \boldsymbol{\lambda} \tag{5.33}$$

Substituting (5.33) into (5.30) yields

$$\frac{1}{2} V W V^T \cdot \boldsymbol{\lambda} = \Delta \mathbf{b} \tag{5.34}$$

As V has full rank and W is regular, VWV^T is regular, and we can solve for λ

$$\boldsymbol{\lambda} = 2(VWV^T)^{-1} \cdot \Delta \mathbf{b} \tag{5.35}$$

Substituting (5.35) into (5.33) yields

$$\mathfrak{s} = WV^T \cdot (VWV^T)^{-1} \cdot \Delta \mathbf{b} \tag{5.36}$$

□

Applying Theorem 5.1 to the reduced problem (5.26) yields the solution

$$\bar{\mathfrak{s}} = \bar{W}\bar{V}^T \cdot (\bar{V}\bar{W}\bar{V}^T)^{-1} \cdot \Delta \bar{\mathbf{b}} \tag{5.37}$$

Obviously, these expressions equal the LLS formulae (5.12) and (5.18), i.e. the minimization formulation and the LLS method are equivalent.

The minimization problem admits the following reasoning: From the class of stencils which satisfy the constraints, one is selected, for which the weighted Euclidian norm of the components is minimized. In the approximating case, the central entry is also minimized, whereas in the interpolating case, it is not. Hence, in the interpolating case, the central entry will always be larger (or at most equal) in absolute value. Desired Laplace stencils have a negative central entry and non-negative entries aside. Then, due to (4.14) the central entry equals in absolute value the sum of the other entries. Hence, the interpolating approach, which does not minimize the central entry, has better chances of keeping the neighboring stencil entries positive.

5.1.4 Approximating and Interpolating Approach in 1d

We derive the ALLS and the ILLS approximations for the first and second derivative in one space dimension. The point of evaluation is x_0. The ILLS expressions contain the expression relative to the point of evaluation $\bar{x}_i = x_i - x_0$. In the following, the expressions involving

$\bar{x}_i$, $\bar{s}_i$, $\bar{r}_i$ are the ILLS approximations, while terms without an overbar belong to the ALLS approach.

The matrices for the linear basis are

$$V = \begin{pmatrix} 1 & \dots & 1 \\ x_0 & \dots & x_m \end{pmatrix} \quad \text{and} \quad \bar{V} = \begin{pmatrix} \bar{x}_1 & \dots & \bar{x}_m \end{pmatrix} , \tag{5.38}$$

and for the quadratic basis

$$V = \begin{pmatrix} 1 & \dots & 1 \\ x_0 & \dots & x_m \\ x_0^2 & \dots & x_m^2 \end{pmatrix} \quad \text{and} \quad \bar{V} = \begin{pmatrix} \bar{x}_1 & \dots & \bar{x}_m \\ \bar{x}_1^2 & \dots & \bar{x}_m^2 \end{pmatrix} . \tag{5.39}$$

This yields the system matrix $S = VWV^T$ to become for the linear basis

$$S = \begin{pmatrix} s_0 & s_1 \\ s_1 & s_2 \end{pmatrix} \quad \text{and} \quad \bar{S} = \begin{pmatrix} \bar{s}_2 \end{pmatrix} , \tag{5.40}$$

and for the quadratic basis

$$S = \begin{pmatrix} s_0 & s_1 & s_2 \\ s_1 & s_2 & s_3 \\ s_2 & s_3 & s_4 \end{pmatrix} \quad \text{and} \quad \bar{S} = \begin{pmatrix} \bar{s}_0 & \bar{s}_1 \\ \bar{s}_1 & \bar{s}_2 \end{pmatrix} . \tag{5.41}$$

Here

$$s_k = \sum_{i=0}^{m} w_i x_i^k \quad \text{and} \quad \bar{s}_k = \sum_{i=1}^{m} w_i \bar{x}_i^k \tag{5.42}$$

Note that the ALLS sum s_0 includes the central weight w_0, whereas this is not the case for the ILLS sum $\bar{s}_k$. According to formula (5.12) respectively (5.18), the k^{th} stencil component for the first derivative with the linear basis is

$$\mathfrak{s}_k = \frac{-s_1 w_k + s_0 w_k x_k}{s_0 s_2 - s_1^2} \quad \text{and} \quad \bar{\mathfrak{s}}_k = \frac{w_k \bar{x}_k}{\bar{s}_2} \tag{5.43}$$

With the notations

$$r_k = \sum_{i=0}^{m} w_i x_i^k u_i \quad \text{and} \quad \bar{r}_k = \sum_{i=1}^{m} w_i \bar{x}_i^k (u_i - u_0) \tag{5.44}$$

we obtain the approximations

$$u(x_0)' \approx \frac{s_0 r_1 - s_1 r_0}{s_0 s_2 - s_1^2} \quad \text{and} \quad u(x_0)' \approx \frac{\bar{r}_1}{\bar{s}_2} \tag{5.45}$$

Analogously, with the quadratic basis, we can calculate approximations to the first derivative as

$$u(x_0)' \approx \frac{(s_2 s_3 - s_1 s_4) r_0 + (s_0 s_4 - s_2^2) r_1 + (s_1 s_2 - s_0 s_3) r_2}{s_0 s_2 s_4 + 2 s_1 s_2 s_3 - s_2^3 - s_0 s_3^2 - s_1^2 s_4} \tag{5.46}$$

and

$$u(x_0)' \approx \frac{\bar{s}_4\bar{r}_1 - \bar{s}_3\bar{r}_2}{\bar{s}_2\bar{s}_4 - \bar{s}_3^2} \ . \tag{5.47}$$

The second derivative yields

$$u(x_0)'' \approx 2\frac{(s_1s_3 - s_2^2)r_0 + (s_1s_2 - s_0s_3)r_1 + (s_0s_2 - s_1^2)r_2}{s_0s_2s_4 + 2s_1s_2s_3 - s_2^3 - s_0s_3^2 - s_1^2s_4} \tag{5.48}$$

and

$$u(x_0)'' \approx 2\frac{\bar{s}_2\bar{r}_2 - \bar{s}_3\bar{r}_1}{\bar{s}_2\bar{s}_4 - \bar{s}_3^2} \ . \tag{5.49}$$

When analyzing the moving least squares approach in one space dimension, we will recover corresponding expressions in Section 5.3.2 for the approximating, and in Section 5.4 for the interpolating case.

5.2 Moving Least Squares Method

The Moving Least Squares (MLS) approach was developed by Lancaster and Salkauskas [LS81] as a method for reconstructing a function from values given at scattered data points. It locally best approximates the function in some polynomial basis, weighted with respect to the interpolation point. While the polynomial basis remains fixed, the weights move over the domain Ω, thus the method is also referred to as *moving weighted least squares* (MWLS) method. Due to the weights changing over the domain, the global approximating function is not a polynomial. The method has been analyzed by Levin [Lev98]. It found use as a building block for derivative approximation by Duarte, Liszka and Tworzyako [DLT96]. Recent investigation regarding the approximation of derivatives in the interpolating case has been presented by Sonar [Son05].

Let a function u be given on a domain $\Omega \subset \mathbb{R}^d$. Consider a point cloud $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. The function values $u_1, \ldots, u_n$ are given at the data points by $u_i = u(\mathbf{x}_i)$. The task is to approximate the function value $u(\mathbf{x})$ at an arbitrary point $\mathbf{x} \in \Omega$, i.e. in the first place we are confronted with a meshless interpolation problem.

5.2.1 Distance Weight Function

The first step of the MLS method is to assign weights to the points $\mathbf{x}_i$ via a *distance weight function* w

$$w_i(\mathbf{x}) = w\left(\|\mathbf{x} - \mathbf{x}_i\|_2\right) \ . \tag{5.50}$$

The Euclidian norm is chosen, since it is rotationally invariant. The distance weight function is chosen smooth and decaying with increasing distance. One distinguishes distance weight functions with respect to two features

1. **Regular vs. singular**
   A regular weight function has an existing limit

$$w(0) = \lim_{d \to 0} w(d) \ , \tag{5.51}$$

   whereas a singular weight function has a pole at zero distance

$$w(d) \to \infty \quad \text{as } d \to 0 \ . \tag{5.52}$$

   A regular weight function leads to the *approximating moving least squares* (AMLS) method, while a singular weight function leads to the *interpolating moving least squares* (IMLS) method. If the distance weight function is regular, the function $w(\|\mathbf{x}\|)$ must be twice differentiable in the origin, hence one needs to require $w'(0) = 0$.

2. **Compactly supported vs. global**
   A compactly supported distance weight function has a $d_0 > 0$ such that

$$w(d) = 0 \quad \forall d > d_0 \ , \tag{5.53}$$

   while a global one does not vanish even for large distances. However, the chosen global functions typically decay fast enough such that for computational purposes one can neglect points further away than a given distance. A distance weight function which requires to consider all points would obviously not result in a computationally efficient method.

Analysis of the MLS method is often times carried out for a class of global distance weight functions, in the regular case

$$w(d) = \exp\left(-\frac{\alpha}{2} d^2\right) \ , \tag{5.54}$$

which is also suggested in [LS81], and in the singular case

$$w(d) = d^{-\alpha} \tag{5.55}$$

with $\alpha \geq 2$ even. If one wishes to have strictly compactly supported distance weight functions, one often chooses chooses

$$w(d) = \begin{cases} \exp\left(-(d^2 - d_0^2)^{-2}\right) & d < d_0 \\ 0 & d \geq d_0 \end{cases} \tag{5.56}$$

or splines

$$w(d) = \begin{cases} 1 - 6\left(\frac{d}{d_0}\right)^2 + 8\left(\frac{d}{d_0}\right)^3 - 3\left(\frac{d}{d_0}\right)^4 & d < d_0 \\ 0 & d \geq d_0 \end{cases} \ , \tag{5.57}$$

The above parameter would then be $\alpha = \frac{1}{d_0}$. Singular compactly supported weight functions can be obtained as products of (5.55) with (5.56) or (5.57). Also products of (5.55) with (5.54) have been considered.

5.2.2 Approximation in Polynomial Basis

Let P_ν the the vector space of polynomials up to order ν. [1] Consider a basis $(b_0, \ldots, b_k)$ of this vector space. For instance, for P_2 in 2d one can consider the canonical basis (5.2). Any polynomial $p \in P_\nu$ can be written as

$$p(\mathbf{x}) = \sum_{j=0}^{k} a_j b_j(\mathbf{x}) = \mathbf{b}(\mathbf{x})^T \cdot \mathbf{a} \tag{5.58}$$

with basis vector $\mathbf{b}(\mathbf{x}) = (b_0(\mathbf{x}), \ldots, b_k(\mathbf{x}))^T$ and coefficient vector $\mathbf{a} = (a_0, \ldots, a_k)^T$. A specific point $\hat{\mathbf{x}}$ is considered and the functional

$$E_{\hat{\mathbf{x}}}(\mathbf{a}) = \sum_{i=1}^{n} w_i(\hat{\mathbf{x}}) \left(p(\mathbf{x}_i) - u_i\right)^2 \tag{5.59}$$

is minimized, i.e. one selects the polynomial $p \in P_\nu$, defined by the coefficient vector $\mathbf{a}$, which minimizes the weighted least squares distance at the data points $\mathbf{x}_i$. Since the weights w_i are taken with respect to the point $\hat{\mathbf{x}}$, for every point $\hat{\mathbf{x}}$ one obtains a different polynomial, and thus a different coefficient vector $\mathbf{a}(\hat{\mathbf{x}})$.

The coefficient vector $\mathbf{a}(\hat{\mathbf{x}})$ is obtained by the normal equations

$$\left(VW(\hat{\mathbf{x}})V^T\right) \mathbf{a}(\hat{\mathbf{x}}) = \left(VW(\hat{\mathbf{x}})\right) \mathbf{u} \tag{5.60}$$

where

$$V = \begin{pmatrix} b_0(\mathbf{x}_1) & \ldots & b_0(\mathbf{x}_n) \\ \vdots & \ddots & \vdots \\ b_k(\mathbf{x}_1) & \ldots & b_k(\mathbf{x}_n) \end{pmatrix} \in \mathbb{R}^{(k+1)\times n} \tag{5.61}$$

and

$$W = \begin{pmatrix} w_1(\hat{\mathbf{x}}) & & \\ & \ddots & \\ & & w_n(\hat{\mathbf{x}}) \end{pmatrix} \in \mathbb{R}^{n\times n} \ . \tag{5.62}$$

Hence, for each point $\hat{\mathbf{x}}$ a small linear system has to be solved with matrix $VW(\hat{\mathbf{x}})V^T \in \mathbb{R}^{(k+1)\times(k+1)}$. For every point $\hat{\mathbf{x}}$ one has a local polynomial approximation

$$L_{\hat{\mathbf{x}}} u(\mathbf{x}) = p_{\hat{\mathbf{x}}}(\mathbf{x}) = \mathbf{b}(\mathbf{x})^T \cdot \mathbf{a}(\hat{\mathbf{x}}) \tag{5.63}$$

The global MLS interpolating function Πu is then obtained by assigning $\Pi u(\hat{\mathbf{x}})$ the value of the polynomial (5.58) at the same point $\hat{\mathbf{x}}$. In other words, for the approximation there is

[1] Indeed, the MLS method can be applied for more general function spaces. However, with respect to the approximation of derivatives we shall confine with polynomial bases here.

no difference between the two points, thus the MLS function Πu is given for every point $\mathbf{x}$ as

$$\Pi u(\mathbf{x}) = p_{\mathbf{x}}(\mathbf{x}) = \mathbf{b}(\mathbf{x})^T \cdot \mathbf{a}(\mathbf{x}) \tag{5.64}$$

If this approach is done for all points $\mathbf{x}$, one obtains from the values $(u_1, \ldots, u_n)$ a new weighted least squares function Πu defined on the whole domain Ω. Formally, the operator Π acts on a function u and yields a new function which is constructed only from the values of u at the data points of the cloud X.

Remark 5.2. Note that the operator Π depends on the point cloud X, the order of the polynomial subspace P_ν, and on the distance weight function w, i.e. $\Pi u = \Pi_{X,\nu,w} u$. However, for the simplicity of notation, we omit the X, ν and w dependence, whenever there is exactly one point cloud and exactly one distance weight function specified, and the polynomial order is clear.

Remark 5.3. Also note that only in the case of a singular distance weight function, the resulting MLS function Πu actually interpolates the data. In the case of a regular distance weight function, the function Πu approximates the data.

5.2.3 Shepard's Method

When minimizing over P_0, i.e. considering constant polynomials only, one ends up with *Shepard's method* [She68]. In this case the matrix in system (5.60) is a real number

$$VW(\mathbf{x})V^T = \sum_{j=1}^{n} w_i(\mathbf{x}) \tag{5.65}$$

and the resulting function is

$$Su(\mathbf{x}) = \Pi_0 u(\mathbf{x}) = \frac{\sum_{i=1}^n w_i(\mathbf{x}) u_i}{\sum_{i=1}^n w_i(\mathbf{x})} = \sum_{i=1}^{n} u_i \varphi_i(\mathbf{x}) \ . \tag{5.66}$$

where φ_i are the basis functions

$$\varphi_i(\mathbf{x}) = \frac{w_i(\mathbf{x})}{\sum_{j=1}^n w_j(\mathbf{x})} \ . \tag{5.67}$$

Assumed the supports of the distance weight functions centered at the points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ cover the whole domain Ω, the basis functions form a partition of unity. If the distance weight function is non-negative, the basis functions are bounded

$$0 < \varphi_i(\mathbf{x}) \leq 1 \quad \forall i = 1, \ldots, n \ , \tag{5.68}$$

thus the Shepard function Su does not overshoot the data. Obviously, smoothness properties of the distance weight function transfer to the basis functions and thus carry over to the

Shepard function Su. In case of a singular distance weight function, the basis functions satisfy

$$\lim_{\mathbf{x}\to\mathbf{x}_j} \varphi_i(\mathbf{x}) = \delta_{ij} \ , \tag{5.69}$$

hence one defines $\varphi_i(\mathbf{x}_j) = \delta_{ij}$. Consequently, the function Su interpolates the data. For a regular distance weight function, the basis functions satisfy $\varphi_i(\mathbf{x}_i) < 1$. Consequently, the function Su approximates the data.

5.2.4 Derivatives by the Moving Least Squares Method

The moving least squares method uses function values at the point cloud X to construct a function on the whole domain Ω. The MLS function Πu is at least as smooth as the distance weight function. Thus, the MLS method can be used to approximate derivatives. While in principle one can employ the method to approximate a derivate at any point $\mathbf{x} \in \Omega$, for the given Poisson problem (4.1) we are interested in approximating derivatives at the points of the cloud X. In particular we interested in approximating the Laplace operator in the interior and first derivative at Neumann boundary points. The general ideas presented in the following, however, apply to derivatives of any order. In this section we derive the first and second derivatives of the MLS functions and show that the LLS approach, as presented in Section 5.1, can be interpreted as a special case of a specific type of MLS differentiation.

Let u be a function on Ω. Let a point cloud X be given, a polynomial basis be prescribed and a distance weight function be specified. Let Πu the corresponding MLS function. The MLS approach gives rise to different approaches for approximating derivatives:

1. **Local differentiation**
   The MLS function is based on a local weighted polynomial least squares fit. At a fixed point $\hat{\mathbf{x}}$ consider the local polynomial approximation

   $$L_{\hat{\mathbf{x}}} u(\mathbf{x}) = \mathbf{b}(\mathbf{x})^T \cdot \mathbf{a}(\hat{\mathbf{x}}) \ . \tag{5.70}$$

   Let ∂ denote a differential operator. Differentiation with respect to $\mathbf{x}$ yields

   $$\partial_{\mathbf{x}} \left(L_{\hat{\mathbf{x}}} u \right) (\mathbf{x}) = \left(\partial \mathbf{b}(\mathbf{x}) \right)^T \cdot \mathbf{a}(\hat{\mathbf{x}}) \ . \tag{5.71}$$

   Now define the *locally differentiated MLS function* $\Pi^{\partial} u$, analogously to the MLS function, by evaluating (5.71) at the same point, which we denote $\mathbf{x}$ for simplicity

   $$\Pi^{\partial} u(\mathbf{x}) = \left(\partial \mathbf{b}(\mathbf{x}) \right)^T \cdot \mathbf{a}(\mathbf{x}) \ . \tag{5.72}$$

The function $\Pi^{\partial} u$ is the locally differentiated approximation to the function ∂u. Similarly one can define other local differentiation operators, such as

$$\Pi^{\partial^2} u(\mathbf{x}) = \left(\partial^2 \mathbf{b}(\mathbf{x})\right)^T \cdot \mathbf{a}(\mathbf{x}) \tag{5.73}$$

$$\Pi^{\frac{\partial}{\partial \mathbf{n}}} u(\mathbf{x}) = \left(\frac{\partial \mathbf{b}}{\partial \mathbf{n}}(\mathbf{x})\right)^T \cdot \mathbf{a}(\mathbf{x}) \tag{5.74}$$

$$\Pi^{\nabla} u(\mathbf{x}) = \sum_{j=0}^{k} \nabla b_j(\mathbf{x}) a_j(\mathbf{x}) \tag{5.75}$$

$$\Pi^{\Delta} u(\mathbf{x}) = \left(\Delta \mathbf{b}(\mathbf{x})\right)^T \cdot \mathbf{a}(\mathbf{x}) \tag{5.76}$$

This approach is identical to the LLS method, as presented in Section 5.1. Hence, the distance weight function need not be smooth. The MLS function Πu is not differentiated, rather are new operators defined.

2. **Global differentiation**
Consider the derivative of the global MLS interpolant

$$\Pi u(\mathbf{x}) = \mathbf{b}(\mathbf{x})^T \cdot \mathbf{a}(\mathbf{x}) \tag{5.77}$$

as an approximation to the derivative of $u(\mathbf{x})$. Since both $\mathbf{a}$ and $\mathbf{b}$ depend on $\mathbf{x}$, one obtains additional terms by the chain rule

$$\nabla \left(\Pi u\right)(\mathbf{x}) = \sum_{j=0}^{k} \left(\nabla b_j(\mathbf{x}) a_j(\mathbf{x}) + b_j(\mathbf{x}) \nabla a_j(\mathbf{x})\right) \tag{5.78}$$

$$\frac{\partial \left(\Pi u\right)}{\partial \mathbf{n}}(\mathbf{x}) = \left(\frac{\partial \mathbf{b}}{\partial \mathbf{n}}(\mathbf{x})\right)^T \cdot \mathbf{a}(\mathbf{x}) + \mathbf{b}(\mathbf{x})^T \cdot \frac{\partial \mathbf{a}}{\partial \mathbf{n}}(\mathbf{x}) \tag{5.79}$$

$$\Delta \left(\Pi u\right)(\mathbf{x}) = \sum_{j=0}^{k} \left(\Delta b_j(\mathbf{x}) a_j(\mathbf{x}) + 2 \nabla b_j(\mathbf{x}) \cdot \nabla a_j(\mathbf{x}) + b_j(\mathbf{x}) \Delta a_j(\mathbf{x})\right) \tag{5.80}$$

In both expressions (5.80) and (5.79), the global MLS derivative consists of the LLS derivative plus additional terms due to the chain rule.

3. **Mixed local and global differentiation**
Differentiating the locally differentiated MLS function yields a mixed local–global approximation. Of particular interest with respect to the Poisson problem (4.1) is the approximation to the Laplacian

$$\begin{aligned} \nabla \cdot \left(\Pi^{\nabla} u\right)(\mathbf{x}) &= \nabla \cdot \left(\sum_{j=0}^{k} \nabla b_j(\mathbf{x}) a_j(\mathbf{x})\right) \\ &= \sum_{j=0}^{k} \left(\Delta b_j(\mathbf{x}) a_j(\mathbf{x}) + \nabla b_j(\mathbf{x}) \cdot \nabla a_j(\mathbf{x})\right) \end{aligned} \tag{5.81}$$

This expression best reflects the weak form of the Laplace operator.

In one space dimension the functions of interest are for the approximation of first derivatives

$$(\Pi' u) = (\mathbf{b}')^T \cdot \mathbf{a} \qquad \text{local} \tag{5.82}$$

$$(\Pi u)' = (\mathbf{b}')^T \cdot \mathbf{a} + \mathbf{b}^T \cdot \mathbf{a}' \qquad \text{global} \tag{5.83}$$

and for second derivatives

$$(\Pi'' u) = (\mathbf{b}'')^T \cdot \mathbf{a} \qquad \text{local} \tag{5.84}$$

$$(\Pi' u)' = (\mathbf{b}'')^T \cdot \mathbf{a} + (\mathbf{b}')^T \cdot \mathbf{a}' \qquad \text{mixed} \tag{5.85}$$

$$(\Pi u)'' = (\mathbf{b}'')^T \cdot \mathbf{a} + 2\,(\mathbf{b}')^T \cdot \mathbf{a}' + \mathbf{b}^T \cdot \mathbf{a}'' \qquad \text{global} \tag{5.86}$$

Generally, the global approximations equal the local approximations, plus additional terms due to the chain rule.

5.3 Derivatives by Approximating Moving Least Squares

In this section we consider the AMLS method, i.e. the MLS method with regular distance weight functions. If the distance weight function is smooth, the MLS approximant Πu can be differentiated at data points. The resulting expression will be a linear combination of function values at the data points $u(x_i)$. The corresponding coefficients form a stencil $\mathfrak{s}$, which – if consistent – yields an approximation to the corresponding derivative of the function u.

We will show that the AMLS method yields consistent derivative approximations only if the polynomial basis is of order of at least the derivative to be taken.

5.3.1 Consistency of First and Second Derivative

The AMLS function is

$$(\Pi u)(\mathbf{x}) = \mathbf{b}(\mathbf{x})^T \cdot \mathbf{a}(\mathbf{x}) \ , \tag{5.87}$$

where the coefficient vector $\mathbf{a}(\mathbf{x})$ is the solution to the system

$$S(\mathbf{x})\mathbf{a}(\mathbf{x}) = \mathbf{r}(\mathbf{x}) \ . \tag{5.88}$$

Here the matrix S and right hand side $\mathbf{r}$ are

$$S(\mathbf{x}) = VW(\mathbf{x})V^T \tag{5.89}$$

and the right hand side

$$\mathbf{r}(\mathbf{x}) = VW(\mathbf{x})\mathbf{u} \tag{5.90}$$

Note that both S and $\mathbf{r}$ depend on $\mathbf{x}$. Consequently, $\mathbf{a}$ as well as the basis vector $\mathbf{b}$ depend on $\mathbf{x}$. In the following we omit the $\mathbf{x}$-dependence in the expressions.

For equation (5.88) the coefficient vector $\mathbf{a}$ calculates as

$$\mathbf{a} = S^{-1}\mathbf{r} \ . \tag{5.91}$$

Let in the following the prime $'$ denote a differential operator (such as $\frac{\partial}{\partial x_k}$), and the double-prime $''$ denote the operator applied twice. Applying the chain rule to equation (5.88) yields

$$S\mathbf{a}' + S'\mathbf{a} = \mathbf{r}' \ , \tag{5.92}$$

and thus

$$\mathbf{a}' = S^{-1}(\mathbf{r}' - S'S^{-1}\mathbf{r}) \ . \tag{5.93}$$

Differentiating equation (5.88) twice yields

$$S\mathbf{a}'' + 2S'\mathbf{a}' + S''\mathbf{a} = \mathbf{r}'' \ , \tag{5.94}$$

and thus

$$\mathbf{a}'' = S^{-1}\left(\left(\mathbf{r}'' - S''S^{-1}\mathbf{r}\right) - 2S'S^{-1}(\mathbf{r}' - S'S^{-1}\mathbf{r})\right) \tag{5.95}$$

Using (5.93), we obtain the MLS function's first derivative as

$$\begin{aligned} (\Pi u)' &= (\mathbf{b}')^T \cdot \mathbf{a} + \mathbf{b}^T \cdot \mathbf{a}' && (5.96)\\ &= (\mathbf{b}')^T \cdot S^{-1}\mathbf{r} + \mathbf{b}^T \cdot S^{-1}(\mathbf{r}' - S'S^{-1}\mathbf{r}) && (5.97) \end{aligned}$$

Similarly, using (5.93) and (5.95), we obtain the MLS function's second derivative as

$$\begin{aligned} (\Pi u)'' &= (\mathbf{b}'')^T \cdot \mathbf{a} + 2\,(\mathbf{b}')^T \cdot \mathbf{a}' + \mathbf{b}^T \cdot \mathbf{a}'' && (5.98)\\ &= (\mathbf{b}'')^T \cdot S^{-1}\mathbf{r} + 2\,(\mathbf{b}')^T \cdot S^{-1}(\mathbf{r}' - S'S^{-1}\mathbf{r}) && (5.99)\\ &\quad + \mathbf{b}^T \cdot S^{-1}\left(\left(\mathbf{r}'' - S''S^{-1}\mathbf{r}\right) - 2S'S^{-1}(\mathbf{r}' - S'S^{-1}\mathbf{r})\right) \end{aligned}$$

Theorem 5.2. *If the distance weight function is sufficiently smooth, the AMLS method yields a consistent approximation to derivatives of order $k = 0, \ldots, \nu$, where ν is the highest polynomial order of vector space P_ν.*

Proof. In relations (5.97) and (5.99) the matrix S and its derivatives are

$$S^{(k)} = VW^{(k)}V^T \tag{5.100}$$

Evaluating w.l.o.g. the derivatives at the origin $\mathbf{x} = 0$, we obtain from Definition 4.6 that consistency means nothing else than the relation

$$V \cdot \mathfrak{s}^{(k)} = \mathbf{b}^{(k)} \tag{5.101}$$

must hold, where $\mathfrak{s}^{(k)}$ is the stencil approximating the k^{th} derivative. The stencil $\left(\mathfrak{s}^{(k)}\right)^T$ is obtained by not inserting the function values $\mathbf{u}$ into the expressions $\mathbf{r}$ and its derivatives in formulae (5.97) and (5.99). Consistency means that inserting the matrix V^T instead of the vector $\mathbf{u}$ must yield the vector $\left(\mathbf{b}^{(k)}\right)^T$. However, replacing $\mathbf{u}$ by V^T transform any vector $\mathbf{r}^{(k)}$ into the matrix $S^{(k)}$. This inserted into (5.97) this yields for the first derivative

$$\begin{aligned}
&\left(\mathbf{b}'\right)^T \cdot S^{-1}S + \mathbf{b}^T \cdot S^{-1}(S' - S'S^{-1}S) && (5.102)\\
=&\left(\mathbf{b}'\right)^T + \mathbf{b}^T \cdot S^{-1}(S' - S') && (5.103)\\
=&\left(\mathbf{b}'\right)^T && (5.104)
\end{aligned}$$

and for the second derivative

$$\begin{aligned}
&\left(\mathbf{b}''\right)^T \cdot S^{-1}S + 2\left(\mathbf{b}'\right)^T \cdot S^{-1}(S' - S'S^{-1}S) && (5.105)\\
&+ \mathbf{b}^T \cdot S^{-1}\left(\left(S'' - S''S^{-1}S\right) - 2S'S^{-1}(S' - S'S^{-1}S)\right) && (5.106)\\
=&\left(\mathbf{b}''\right)^T + 2\left(\mathbf{b}'\right)^T \cdot S^{-1}(S' - S') && (5.107)\\
&+ \mathbf{b}^T \cdot S^{-1}\left((S'' - S'') - 2S'S^{-1}(S' - S')\right) && (5.108)\\
=&\left(\mathbf{b}''\right)^T && (5.109)
\end{aligned}$$

This procedure applies in the same manner to higher derivatives. However, here we confine with second derivatives. □

5.3.2 Special Cases in One Space Dimension

The analysis in one space dimension can provide a good understanding of the various terms appearing in the approximations. The fact that in 1d the points always lie on some natural "grid" (namely the coordinate axis) simplifies the geometric considerations. For special cases of symmetric (i.e. to any point x_i there is a point $-x_i$) and regular (i.e. equidistant) grids, various expressions can be worked out analytically and thus admit a deeper analysis.

In one space dimension, system (5.88) takes a special structure

$$S(x) = \begin{pmatrix} s_0(x) & \dots & s_m(x) \\ \vdots & \ddots & \vdots \\ s_m(x) & \dots & s_{2m}(x) \end{pmatrix}, \tag{5.110}$$

where

$$s_k(x) = \sum_{i=1}^{m} w_i(x) x_i^k \tag{5.111}$$

and the right hand side is

$$\mathbf{r}(x) = \begin{pmatrix} r_0(x) \\ \vdots \\ r_m(x) \end{pmatrix}, \tag{5.112}$$

where

$$r_k(x) = \sum_{i=1}^{m} w_i(x) x_i^k u_i \ . \tag{5.113}$$

Consequently, the derivatives of the matrix $S(x)$ and the vector $\mathbf{r}$ have the components

$$s_k'(x) = \sum_{i=1}^{m} w_i'(x) x_i^k \tag{5.114}$$

$$s_k''(x) = \sum_{i=1}^{m} w_i''(x) x_i^k \tag{5.115}$$

$$r_k'(x) = \sum_{i=1}^{m} w_i'(x) x_i^k u_i \tag{5.116}$$

$$r_k''(x) = \sum_{i=1}^{m} w_i''(x) x_i^k u_i \tag{5.117}$$

Remark 5.4. Note that the the expression $w_i'(x)$ evaluated at the origin (as done in later sections) equals $w_i'(0) = w'(0 - x_i) = w'(-x_i)$. Since w' is an odd function, the minus sign is of importance (as opposed to w_i and w_i'', which are even, thus the sign is not important).

In order to gain more insight into the AMLS approximations to derivatives, we consider the following special cases:

- **The point setup is a symmetric grid** (Section 5.3.3)
  In this case various cancellations let the expressions take a special structure. We consider the first derivative to the AMLS function with linear basis. Interesting here are non-Gaussian distance weight functions. We consider the example of a Runge function.

- **The distance weight function is a Gaussian function** (Section 5.3.4)
  In this case the derivatives of the distance weight function can be expressed in terms of function values only. Furthermore, various expressions take a specific structure.

- **The point setup is symmetric and the distance weight function is Gaussian** (Section 5.3.5)
  In this special case we can calculate the stencil entries of the first and second derivative of the AMLS function with firstly linear and secondly quadratic basis. In case of a quadratic basis, the AMLS stencils differ from the LLS stencils. We consider the stencil values depending on the ratio between width of the weight function and mesh size.

Also, the analysis for the symmetric point setup equips us with a handy tool for showing inconsistency in the case of low polynomial degree, as we will do in Section 5.3.6.

5.3.3 Special Case: First Derivatives on Symmetric Grid

Consider the grid $X = (-h_m, \ldots, -h_1, 0, h_1, \ldots, h_m)$. We calculate the AMLS approximation to the first derivative, when a linear basis is used. Due to the symmetry, we obtain various cancellations. The expressions (5.111), (5.113) and their derivatives, evaluated at the origin, become

$$\begin{array}{lll} s_0 = w_0 + 2\sum w_i & s_0' = 0 & s_0'' = w_0'' + 2\sum w_i'' \\ s_1 = 0 & s_1' = 2\sum w_1' h_1 & s_1'' = 0 \\ s_2 = 2\sum w_i h_i^2 & s_2' = 0 & s_2'' = 2\sum w_i'' h_i^2 \\ r_0 = w_0 u_0 + \sum w_i u_i^+ & r_0' = \sum w_i' u_i^- & r_0'' = w_0'' u_0 + \sum w_i'' u_i^+ \\ r_1 = \sum w_i h_i u_i^- & r_1' = \sum w_i' h_i u_i^+ & r_1'' = \sum w_i'' h_i u_i^- \end{array} \tag{5.118}$$

For simplicity we omit to express the evaluation point (0). The terms for the function values are

$$u_i^+ = u(h_i) + u(-h_i) \quad \text{and} \quad u_i^- = u(h_i) - u(-h_i)\ . \tag{5.119}$$

With these notations and cancellations, the AMLS function's first derivative becomes

$$(\Pi u)'(0) = (\mathbf{b}')^T S^{-1}\mathbf{r} + \mathbf{b}^T S^{-1}(\mathbf{r}' - S' S^{-1}\mathbf{r}) \tag{5.120}$$

$$= \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{1}{s_0} & 0 \\ 0 & \frac{1}{s_2} \end{pmatrix} \begin{pmatrix} r_0 \\ r_1 \end{pmatrix} \tag{5.121}$$

$$+ \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{s_0} & 0 \\ 0 & \frac{1}{s_2} \end{pmatrix} \left(\begin{pmatrix} r_0' \\ r_1' \end{pmatrix} - \begin{pmatrix} 0 & \frac{s_1'}{s_2} \\ \frac{s_1'}{s_0} & 0 \end{pmatrix} \begin{pmatrix} r_0 \\ r_1 \end{pmatrix} \right) \tag{5.122}$$

$$= \frac{r_1}{s_2} + \frac{r_0' - \frac{s_1'}{s_2} r_1}{s_0} \tag{5.123}$$

$$= \frac{\sum w_i h_i u_i^-}{2\sum w_i h_i^2} + \frac{\sum w_i' u_i^- - \frac{\sum w_1' h_1}{\sum w_i h_i^2}(\sum w_i h_i u_i^-)}{w_0 + 2\sum w_i} \tag{5.124}$$

Gaussian Weight Function

For the popular choice of a Gaussian distance weight function, the terms r_0' and $\frac{s_1'}{s_2} r_2$ in expression (5.123) cancel, due to the property $w_i' = \alpha h_i w_i$. Thus, the AMLS derivative approximation is merely

$$(\Pi u)'(0) = \frac{r_1}{s_2} = \frac{\sum w_i h_i u_i^-}{2\sum w_i h_i^2}\ , \tag{5.125}$$

which equals the LLS approximation derived in Section 5.1.4. We will consider the case of Gaussian weight functions in more detail in Section 5.3.4 for arbitrary grids and in Section 5.3.5 for symmetric grids.

Runge Weight Function

However, for non-Gaussian distance weight functions, this cancellation need not take place. For instance, consider as a distance weight function a *Runge function*, which is

$$w(d) = \frac{1}{1 + \frac{\alpha}{2} d^2} \tag{5.126}$$

It satisfies $w'(d) = -\alpha d w(d)^2$, hence in the above notation $w_i' = \alpha h_i w_i^2$. In this case the above terms do not cancel out. Instead, the second summand in (5.123) yields a contribution additional to the expression obtained by the LLS method. In order to obtain simpler expressions for this additional term, we consider the regular five point grid $X = (-2h, -h, 0, h, 2h)$. In this case, the derivative becomes

$$(\Pi u)'(0) = \frac{r_1}{s_2} + \frac{s_2 r_0' - s_1' r_1}{s_0 s_2} \tag{5.127}$$

$$= \frac{w_1 u_1^- + 2 w_2 u_2^-}{2(w_1 + 4w_2)} \cdot \frac{1}{h} + \frac{2 w_1 w_2 (w_1 - w_2)}{w_0 + 2(w_1 + w_2)} \cdot \frac{2u_1^- - u_2^-}{w_1 + 4w_2} \cdot \alpha h \tag{5.128}$$

When scaling the parameter α accordingly to the mesh size h, i.e.

$$\alpha = \frac{\mu}{h^2} \ , \tag{5.129}$$

one obtains

$$(\Pi u)'(0) = \frac{1}{h} \left((k_1 + 2l\mu) u_1^- + (k_2 - l\mu) u_2^- \right) \ , \tag{5.130}$$

where

$$k_1 = \frac{w_1}{2(w_1 + 4w_2)} = \frac{2\mu + 1}{8\mu + 10} \quad \text{and} \quad k_2 = \frac{2w_2}{2(w_1 + 4w_2)} = \frac{\mu + 2}{8\mu + 10} \tag{5.131}$$

and

$$l = \frac{2 w_1 w_2 (w_1 - w_2)}{w_0 + 2(w_1 + w_2)} \cdot \frac{1}{w_1 + 4w_2} = \frac{6\mu}{(4\mu + 5)(2\mu^2 + 15\mu + 10)} \tag{5.132}$$

For $\mu = 1$ these take the values

$$k_1 = \frac{1}{6} \ , \quad k_2 = \frac{1}{6} \ , \quad l = \frac{2}{81} \tag{5.133}$$

Hence the LLS stencil is

$$\mathfrak{s} = \frac{1}{h} \left(-\frac{1}{6}, -\frac{1}{6}, 0, \frac{1}{6}, \frac{1}{6} \right)^T \ , \tag{5.134}$$

whereas the MLS stencil is

$$\mathfrak{s} = \frac{1}{h} \left(-\frac{1}{6} + \frac{2}{81}, -\frac{1}{6} - \frac{4}{81}, 0, \frac{1}{6} + \frac{4}{81}, \frac{1}{6} - \frac{2}{81} \right)^T \ . \tag{5.135}$$

5.3.4 Special Case: Gaussian Weight Function on Arbitrary Grid

For the AMLS method, a Gaussian distance weight function $w(d) = \exp(-\frac{\alpha}{2}d^2)$ is a popular choice. In this section we assume the distance weight function to satisfy

$$w'(d) = -\alpha d \; w(d) \; . \tag{5.136}$$

Obviously, a Gaussian function satisfies this property. Using the above property, it follows

$$w_i'(x) = -\alpha(x - x_i)w_i(x) \tag{5.137}$$
$$w_i''(x) = \left(\alpha^2(x - x_i)^2 - \alpha\right) w_i(x) \; , \tag{5.138}$$

hence

$$s_k' = \sum_{i=1}^{m} w_i' x_i^k = \alpha \sum_{i=1}^{m} (x - x_i) w_i x_i^k = \alpha(s_{k+1} - x s_k) \tag{5.139}$$
$$s_k'' = \alpha^2 x^2 s_k - 2\alpha^2 x s_{k+1} + \alpha^2 s_{k+2} - \alpha s_k \tag{5.140}$$

Similarly

$$r_k' = \alpha(r_{k+1} - x r_k) \tag{5.141}$$
$$r_k'' = \alpha^2 x^2 r_k - 2\alpha^2 x r_{k+1} + \alpha^2 r_{k+2} - \alpha r_k \tag{5.142}$$

Introducing the notations

$$S^+ = \begin{pmatrix} s_1 & \dots & s_{m+1} \\ \vdots & \ddots & \vdots \\ s_{m+1} & \dots & s_{2m+1} \end{pmatrix} \quad \text{and} \quad S^{++} = \begin{pmatrix} s_2 & \dots & s_{m+2} \\ \vdots & \ddots & \vdots \\ s_{m+2} & \dots & s_{2m+2} \end{pmatrix} \tag{5.143}$$

as well as

$$\mathbf{r}^+ = \begin{pmatrix} r_1 \\ \vdots \\ r_{m+1} \end{pmatrix} \quad \text{and} \quad \mathbf{r}^{++} = \begin{pmatrix} r_2 \\ \vdots \\ r_{m+2} \end{pmatrix} \tag{5.144}$$

we obtain for the derivatives of the matrix S and the right hand side $\mathbf{r}$

$$S' = \alpha(S^+ - xS) \tag{5.145}$$
$$\mathbf{r}' = \alpha(\mathbf{r}^+ - x\mathbf{r}) \tag{5.146}$$

and

$$S'' = \alpha^2 x^2 S - \alpha S - 2\alpha^2 x S^+ + \alpha^2 S^{++} \tag{5.147}$$
$$\mathbf{r}'' = \alpha^2 x^2 \mathbf{r} - \alpha\mathbf{r} - 2\alpha^2 x \mathbf{r}^+ + \alpha^2 \mathbf{r}^{++} \tag{5.148}$$

Some algebra yields that for the first and second derivative various terms, in particular all terms involving x, cancel out and one obtains

$$\mathbf{a}' = \alpha S^{-1}(\mathbf{r}^+ - S^+S^{-1}\mathbf{r}) \tag{5.149}$$
$$\mathbf{a}'' = \alpha^2 S^{-1}\left(\left(\mathbf{r}^{++} - S^{++}S^{-1}\mathbf{r}\right) - 2S^+S^{-1}(\mathbf{r}^+ - S^+S^{-1}\mathbf{r})\right) \tag{5.150}$$

According to Section 5.2.4 this yields the derivatives

$$(\Pi' u) = (\mathbf{b}')^T S^{-1}\mathbf{r} \tag{5.151}$$
$$(\Pi u)' = (\mathbf{b}')^T S^{-1}\mathbf{r} + \alpha \mathbf{b}^T S^{-1}(\mathbf{r}^+ - S^+S^{-1}\mathbf{r}) \tag{5.152}$$

and

$$(\Pi'' u) = (\mathbf{b}'')^T S^{-1}\mathbf{r} \tag{5.153}$$
$$(\Pi' u)' = (\mathbf{b}'')^T S^{-1}\mathbf{r} + \alpha\,(\mathbf{b}')^T S^{-1}(\mathbf{r}^+ - S^+S^{-1}\mathbf{r}) \tag{5.154}$$
$$\begin{aligned}(\Pi u)'' = {} & (\mathbf{b}'')^T S^{-1}\mathbf{r} + 2\alpha\,(\mathbf{b}')^T S^{-1}(\mathbf{r}^+ - S^+S^{-1}\mathbf{r}) \\ & + \alpha^2 \mathbf{b}^T S^{-1}\left(\left(\mathbf{r}^{++} - S^{++}S^{-1}\mathbf{r}\right) - 2S^+S^{-1}(\mathbf{r}^+ - S^+S^{-1}\mathbf{r})\right)\end{aligned} \tag{5.155}$$

Since the matrix S^+ is the same as the matrix S with the first row removed and an additional row attached at the bottom, multiplication with S^{-1} yields the structure

$$S^+S^{-1} = \begin{pmatrix} 0 & 1 & & \\ & \ddots & 1 & \\ & & 0 & 1 \\ * & \dots & \dots & * \end{pmatrix}, \tag{5.156}$$

Hence, the difference term takes the structure

$$\mathbf{r}^+ - S^+S^{-1}\mathbf{r} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ * \end{pmatrix}. \tag{5.157}$$

Similarly, the expression involving $^{++}$ has the structure

$$\mathbf{r}^{++} - S^{++}S^{-1}\mathbf{r} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ * \\ * \end{pmatrix}. \tag{5.158}$$

5.3.5 Special Case: Gaussian Weight Function on Symmetric Grid

As in Section 5.3.3 we consider the symmetric grid $X = (-h_m, \dots, -h_1, 0, h_1, \dots, h_m)$. Further, as considered in Section 5.3.4, we assume the distance weight function to be Gaussian, i.e. derivatives of expressions can be expressed in terms of higher order expressions. In this special case, we calculate the AMLS expressions for a linear and a quadratic basis and consider the first and second derivative. Again, due to symmetry, we have

$$\begin{array}{llllll} s_0 = & w_0 + 2\sum w_i & r_0 = & w_0 u_0 + \sum w_i u_i^+ & & \\ s_k = & 0 & r_k = & \sum w_i h_i^k u_i^- & \forall k = 1, 3, \dots & (5.159) \\ s_k = & 2\sum w_i h_i^k & r_k = & \sum w_i h_i^k u_i^+ & \forall k = 2, 4, \dots & \end{array}$$

Employing the notations from Section 5.3.4 we obtain for the linear basis

$$S^+ S^{-1} = \begin{pmatrix} 0 & 1 \\ \frac{s_2}{s_0} & 0 \end{pmatrix} \tag{5.160}$$

$$\mathbf{r}^+ - \left(S^+ S^{-1}\right) \mathbf{r} = \begin{pmatrix} 0 \\ r_2 - \frac{s_2}{s_0} r_0 \end{pmatrix} \tag{5.161}$$

$$\left(S^+ S^{-1}\right) \left(\mathbf{r}^+ - \left(S^+ S^{-1}\right) \mathbf{r}\right) = \begin{pmatrix} r_2 - \frac{s_2}{s_0} r_0 \\ 0 \end{pmatrix} \tag{5.162}$$

$$S^{++} S^{-1} = \begin{pmatrix} \frac{s_2}{s_0} & 0 \\ 0 & \frac{s_4}{s_2} \end{pmatrix} \tag{5.163}$$

$$\mathbf{r}^{++} - \left(S^{++} S^{-1}\right) \mathbf{r} = \begin{pmatrix} r_2 - \frac{s_2}{s_0} r_0 \\ r_3 - \frac{s_3}{s_1} r_1 \end{pmatrix} \tag{5.164}$$

and for the quadratic basis

$$S^+ S^{-1} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & \frac{s_4}{s_2} & 0 \end{pmatrix} \tag{5.165}$$

$$\mathbf{r}^+ - \left(S^+ S^{-1}\right) \mathbf{r} = \begin{pmatrix} 0 \\ 0 \\ r_3 - \frac{s_4}{s_2} r_1 \end{pmatrix} \tag{5.166}$$

$$\left(S^+ S^{-1}\right) \left(\mathbf{r}^+ - \left(S^+ S^{-1}\right) \mathbf{r}\right) = \begin{pmatrix} 0 \\ r_3 - \frac{s_4}{s_2} r_1 \\ 0 \end{pmatrix} \tag{5.167}$$

$$S^{++}S^{-1} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & \frac{s_4}{s_2} & 0 \\ \frac{s_4^2 - s_2 s_6}{s_0 s_4 - s_2^2} & 0 & \frac{s_0 s_6 - s_2 s_4}{s_0 s_4 - s_2^2} \end{pmatrix} \tag{5.168}$$

$$\mathbf{r}^{++} - \left(S^{++}S^{-1}\right)\mathbf{r} = \begin{pmatrix} 0 \\ r_3 - \frac{s_4}{s_2} r_1 \\ r_4 - \frac{s_4^2 - s_2 s_6}{s_0 s_4 - s_2^2} r_0 - \frac{s_0 s_6 - s_2 s_4}{s_0 s_4 - s_2^2} r_2 \end{pmatrix} \tag{5.169}$$

This yields by equation (5.152) for the AMLS function's first derivative, for the linear basis

$$(\Pi u)'(0) = (\mathbf{b}')^T S^{-1}\mathbf{r} + \alpha \mathbf{b}^T S^{-1}(\mathbf{r}^+ - S^+ S^{-1}\mathbf{r}) \tag{5.170}$$

$$= \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{1}{s_0} & 0 \\ 0 & \frac{1}{s_2} \end{pmatrix} \begin{pmatrix} r_0 \\ r_1 \end{pmatrix} + \alpha \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{s_0} & 0 \\ 0 & \frac{1}{s_2} \end{pmatrix} \begin{pmatrix} 0 \\ r_2 - \frac{s_2}{s_0} r_0 \end{pmatrix} \tag{5.171}$$

$$= \frac{r_1}{s_2} \tag{5.172}$$

and for the quadratic basis, respectively

$$(\Pi u)'(0) = (\mathbf{b}')^T S^{-1}\mathbf{r} + \alpha \mathbf{b}^T S^{-1}(\mathbf{r}^+ - S^+ S^{-1}\mathbf{r}) \tag{5.173}$$

$$= \frac{1}{s_2(s_0 s_4 - s_2^2)} \left(\begin{pmatrix} 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} s_2 s_4 & 0 & -s_2^2 \\ 0 & s_0 s_4 - s_2^2 & 0 \\ -s_2^2 & 0 & s_0 s_2 \end{pmatrix} \begin{pmatrix} r_0 \\ r_1 \\ r_2 \end{pmatrix} \right. \tag{5.174}$$

$$\left. + \alpha \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} s_2 s_4 & 0 & -s_2^2 \\ 0 & s_0 s_4 - s_2^2 & 0 \\ -s_2^2 & 0 & s_0 s_2 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ r_3 - \frac{s_4}{s_2} r_1 \end{pmatrix} \right)$$

$$= \frac{r_1}{s_2} + \alpha \frac{s_4 r_1 - s_2 r_3}{s_0 s_4 - s_2^2} \tag{5.175}$$

For the second derivative we obtain by equation (5.155) for the linear basis

$$(\Pi u)''(0) = (\mathbf{b}'')^T S^{-1}\mathbf{r} + 2\alpha (\mathbf{b}')^T S^{-1}(\mathbf{r}^+ - S^+ S^{-1}\mathbf{r}) \tag{5.176}$$

$$+ \alpha^2 \mathbf{b}^T S^{-1}\left(\left(\mathbf{r}^{++} - S^{++}S^{-1}\mathbf{r}\right) - 2S^+ S^{-1}(\mathbf{r}^+ - S^+ S^{-1}\mathbf{r})\right)$$

$$= 2\alpha \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{1}{s_0} & 0 \\ 0 & \frac{1}{s_2} \end{pmatrix} \begin{pmatrix} 0 \\ r_2 - \frac{s_2}{s_0} r_0 \end{pmatrix} + \alpha^2 \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{s_0} & 0 \\ 0 & \frac{1}{s_2} \end{pmatrix} \left(\begin{pmatrix} r_2 - \frac{s_2}{s_0} r_0 \\ r_3 - \frac{s_3}{s_1} r_1 \end{pmatrix} - 2 \begin{pmatrix} r_2 - \frac{s_2}{s_0} r_0 \\ 0 \end{pmatrix} \right) \tag{5.177}$$

$$= 2\alpha \frac{s_0 r_2 - s_2 r_0}{s_0 s_2} - \alpha^2 \frac{s_0 r_2 - s_2 r_0}{s_0^2} \tag{5.178}$$

$$= \alpha(2s_0 - \alpha s_2) \frac{s_0 r_2 - s_2 r_0}{s_0^2 s_2} \tag{5.179}$$

and for the quadratic basis, respectively

$$
\begin{aligned}
(\Pi u)''(0) =& \left(\mathbf{b}''\right)^T S^{-1}\mathbf{r} + 2\alpha \left(\mathbf{b}'\right)^T S^{-1}(\mathbf{r}^+ - S^+S^{-1}\mathbf{r}) \\
&+ \alpha^2 \mathbf{b}^T S^{-1}\left(\left(\mathbf{r}^{++} - S^{++}S^{-1}\mathbf{r}\right) - 2S^+S^{-1}(\mathbf{r}^+ - S^+S^{-1}\mathbf{r})\right) && (5.180) \\
=& \frac{1}{s_2(s_0s_4 - s_2^2)} \left(2 \begin{pmatrix} -s_2^2 & 0 & s_0s_2 \end{pmatrix} \begin{pmatrix} r_0 \\ r_1 \\ r_2 \end{pmatrix} \right. \\
&+ 2\alpha \begin{pmatrix} 0 & s_0s_4 - s_2^2 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ r_3 - \frac{s_4}{s_2}r_1 \end{pmatrix} && (5.181) \\
&\left. + \alpha^2 \begin{pmatrix} s_2s_4 & 0 & -s_2^2 \end{pmatrix} \begin{pmatrix} 0 \\ -r_3 + \frac{s_4}{s_2}r_1 \\ r_4 - \frac{s_4^2 - s_2s_6}{s_0s_4 - s_2^2}r_0 - \frac{s_0s_6 - s_2s_4}{s_0s_4 - s_2^2}r_2 \end{pmatrix} \right) \\
=& 2\frac{s_0r_2 - s_2r_0}{s_0s_4 - s_2^2} - \alpha^2 \frac{s_2}{s_0s_4 - s_2^2}\left(r_4 - \frac{s_4^2 - s_2s_6}{s_0s_4 - s_2^2}r_0 - \frac{s_0s_6 - s_2s_4}{s_0s_4 - s_2^2}r_2 \right) && (5.182)
\end{aligned}
$$

Some remarks can be given regarding the derived expressions:

Remark 5.5. Expression (5.172) is identical to the result obtained by the LLS method (see Section 5.1.4), i.e. $(\Pi u)'(0) = (\Pi' u)(0)$. This is due to the special properties of a Gaussian distance weight function and due to the symmetry of the geometry. The example in Section 5.3.3 shows that for other distance weight functions, the AMLS and the LLS approximation can very well differ. Also, note that the approximation is actually second order accurate due to the symmetry of the geometry.

Remark 5.6. The first derivative of the AMLS function with a quadratic basis, as given by expression (5.175), differs from the LLS approximation, which is the first part, $(\Pi' u)(0) = \frac{r_1}{s_2}$.

Remark 5.7. The second derivative of the AMLS function with linear basis is inconsistent. In Theorem 5.4 we show that for no distance weight function a consistent approximation is obtained. Here, in the particular case of a Gaussian weight function, there is no α which would let expression (5.179) satisfy the consistency requirement (4.11).

Remark 5.8. When a quadratic basis is employed, the AMLS function's second derivative is given by expression (5.182). It consists of a term which is obtained by the LLS method, plus an additional term. Note that the middle term cancels, thus the mixed second derivative equals the LLS derivative $(\Pi' u)'(0) = (\Pi'' u)(0)$.

We evaluate expressions (5.175) and (5.182) for the case of a regular symmetric five point grid $X = (-2h, -h, 0, h, 2h)$. We introduce a scaling factor μ, which represents the ratio between the width of the weight function $\sqrt{\alpha}$ and the mesh size h

$$\mu = \alpha h^2 \tag{5.183}$$

Since the grid is regular, we can introduce $h = h_1$ and $w = w_1 = \exp\left(\frac{\mu}{2}\right)$. Then $h_2 = 2h$ and $w_2 = w^4$. Consequently, the expressions s_k and r_k become

$$\begin{array}{llll} s_0 = & 1 + 2w + 2w^4 & r_0 = & u_0 + wu_1^+ + w^4 u_2^+ \\ s_k = & 0 & r_k = & wh^k(u_1^- + 2^k w^3 u_1^-) \quad \forall k = 1, 3, \ldots \\ s_k = & 2wh^k(1 + 2^k w^3) & r_k = & wh^k(u_1^+ + 2^k w^3 u_1^+) \quad \forall k = 2, 4, \ldots \end{array} \tag{5.184}$$

Inserting these terms into the (5.175) yields for the first derivative of the AMLS function with quadratic basis

$$(\Pi u)'(0) = \frac{r_1}{s_2} + \alpha \frac{s_4 r_1 - s_2 r_3}{s_0 s_4 - s_2^2} \tag{5.185}$$

$$= \frac{1}{h}\left(\frac{u_1^- + 2w^3 u_2^-}{2(1 + 4w^3)} + \mu \frac{6w^4(2u_1^- - u_2^-)}{1 + 16w^3 + 18w^4}\right) \tag{5.186}$$

$$= \frac{1}{h}\left(\mathfrak{s}_L + \mu \mathfrak{s}_\Delta\right)^T \mathbf{u} \tag{5.187}$$

Here the stencils (without the $\frac{1}{h}$ scaling) are

$$\mathfrak{s}_L = \frac{1}{2(1 + 4w^3)} \begin{pmatrix} -2w^3 \\ -1 \\ 0 \\ 1 \\ 2w^3 \end{pmatrix} \tag{5.188}$$

and

$$\mathfrak{s}_\Delta = \frac{6w^4}{1 + 16w^3 + 18w^4} \begin{pmatrix} 1 \\ -2 \\ 0 \\ 2 \\ -1 \end{pmatrix} \tag{5.189}$$

Figure 5.1 shows the stencil entries (without the $\frac{1}{h}$ scaling) corresponding to the positions $x = 0$, $x = h$ and $x = 2h$. The other two components at $x = -h$ and $x = -2h$ equal minus the entries at $x = h$ and $x = 2h$. The values are plotted in dependence of the scaling factor $\mu = \alpha h^2$. The solid lines show the components of the vector $\mathfrak{s}_L$, i.e. the LLS approximation, as given by (5.188). The dashed line shows $\mathfrak{s}_L + \mu \mathfrak{s}_\Delta$, i.e. the AMLS approximation, where the difference is given by (5.189).

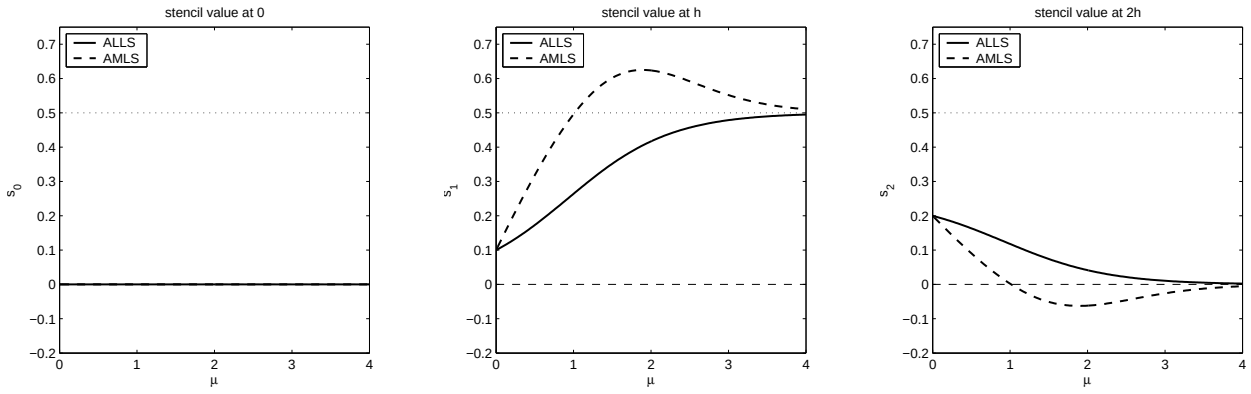

Figure 5.1: Stencil entries for first derivative of AMLS function with quadratic basis

Inserting the terms (5.184) into (5.182) yields for the second derivative

$$(\Pi u)''(0) = 2\frac{s_0 r_2 - s_2 r_0}{s_0 s_4 - s_2^2} - \alpha^2 \frac{s_2}{s_0 s_4 - s_2^2}\left(r_4 - \frac{s_4^2 - s_2 s_6}{s_0 s_4 - s_2^2} r_0 - \frac{s_0 s_6 - s_2 s_4}{s_0 s_4 - s_2^2} r_2\right) \tag{5.190}$$

$$= \frac{1}{h^2}\left(\frac{-2(1+4w^3)w_0 + (1-6w^4)u_1^+ + 2w^3(2+3w)u_2^+}{1+16w^3+18w^4}\right. \tag{5.191}$$

$$\left. + \mu^2 \frac{12w^4(1+4w^3)(-12u_0 + 4u_1^+ - u_2^+)}{(1+16w^3+18w^4)^2}\right)$$

$$= \frac{1}{h^2}\left(\mathfrak{s}_L + \mu \mathfrak{s}_\Delta\right)^T \mathbf{u} \tag{5.192}$$

The corresponding stencils (without the $\frac{1}{h^2}$ scaling) are

$$\mathfrak{s}_L = \frac{1}{1+16w^3+18w^4}\begin{pmatrix} 2w^3(2+3w) \\ 1-6w^4 \\ -2(1+4w^3) \\ 1-6w^4 \\ 2w^3(2+3w) \end{pmatrix} \tag{5.193}$$

and

$$\mathfrak{s}_\Delta = \frac{12w^4(1+4w^3)}{(1+16w^3+18w^4)^2}\begin{pmatrix} -1 \\ 4 \\ -12 \\ 4 \\ -1 \end{pmatrix} \tag{5.194}$$

Similarly to the first derivative, Figure 5.2 shows the stencil entries corresponding to the positions $x = 0$, $x = h$ and $x = 2h$. The other two components at $x = -h$ and $x = -2h$ equal the entries at $x = h$ and $x = 2h$. The values are plotted in dependence of the scaling factor $\mu = \alpha h^2$. The solid lines show the components of the vector $\mathfrak{s}_L$, i.e. the

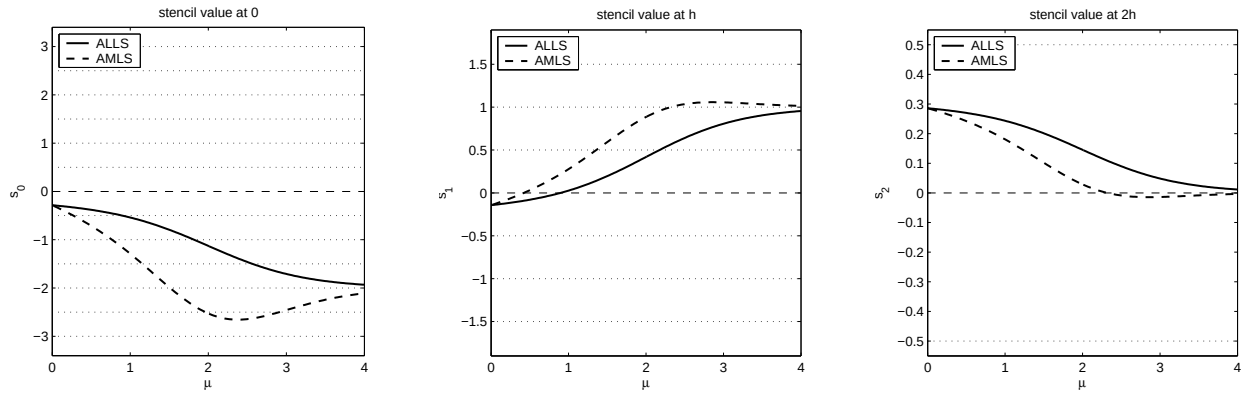

Figure 5.2: Stencil entries for second derivative of AMLS function with quadratic basis

LLS approximation, as given by (5.193). The dashed line shows $\mathfrak{s}_L + \mu\mathfrak{s}_\Delta$, i.e. the AMLS approximation, where the difference is given by (5.194).

For both the first derivative as well as the second derivative approximations we can observe:

- The stencil values depend smoothly on the scaling parameter μ.
- For narrow weight functions ($\mu \to \infty$), all approximations approach the standard central three point stencils $(0, -1, 0, 1, 0)\frac{1}{h}$, respectively $(0, 1, -2, 1, 0)\frac{1}{h^2}$. For wide (almost constant) weight functions ($\mu \to 0$), the approximations approach a non-weighted least squares stencil $\frac{1}{10}(-2, -1, 0, 1, 2)\frac{1}{h}$, respectively $\frac{1}{7}(2, -1, -2, -1, 2)\frac{1}{h^2}$.
- The transfer from wide to narrow takes place monotonically for LLS approaches, while MLS approaches "overshoot" for moderate values of μ.
- In general, it is desirable to have positive values for the stencil values at $x = h$ and $x = 2h$ for the first derivative, while for the second derivative it is advantageous to have positive values at $x = h$ and $x = 2h$ and a negative value at $x = 0$, in other words, the stencil should be positive. For the first derivative, the LLS approach satisfies this condition, while the MLS approach yields a change in sign for the component $x = 2h$ for $\mu > 1.014$. Similarly, for second derivative, the LLS approach yields a positive stencil if the weight function is tight enough ($\mu > \frac{1}{2}\log(6)$). However, the MLS approach yields negative stencil values at $x = 2h$ for $\mu > 2.33$. We will focus deeper on the aspect of positive stencils in Chapter 6.

Due to the monotonicity and the positivity of the stencil entries, the ALLS approaches seem preferable over AMLS approaches for approximating first and second derivatives.

5.3.6 Inconsistency for Insufficient Polynomial Degree

For any choice of degree of the polynomial subspace P_ν, the AMLS function is as smooth as the distance weight function w. Thus, even for low polynomial degrees, higher derivatives of the MLS approximant can be obtained. However, considering derivatives higher than the polynomial degree leads to inconsistent approximations to the real function derivatives. In the following, we consider a regular three point grid $X = (-h, 0, h)$ and derive the corresponding stencils obtained by the first derivative of the Shepard approximant and the second derivative of the AMLS function with linear basis. Both will turn out to yield a correct ratio between stencil entries, but with an incorrect constant.

Theorem 5.3. *The approximating Shepard's method leads to inconsistent first derivatives.*

Proof. We show that Shepard's method yields inconsistent stencils for the regular three point grid $X = (-h, 0, h)$. Using the notations defined in Section 5.3.2, the Shepard approximant is

$$Su(x) = \frac{r_0}{s_0} \ . \tag{5.195}$$

Its first derivative equals

$$(Su)'(x) = \frac{s_0 r_0' - s_0' r_0}{s_0^2} = \frac{r_0'}{s_0} \ , \tag{5.196}$$

where the last equality holds, since $s_0' = 0$ (we require $w'(0) = 0$). Hence

$$(Su)'(0) = \frac{w'(-h)(u(h) - u(-h))}{w(0) + 2w(h)} \ . \tag{5.197}$$

Thus the obtained stencil is $\mathfrak{s} = \frac{w'(-h)}{w(0)+2w(h)}(-1, 0, 1)^T$. For this stencil to be consistent, it must satisfy

$$1 = (-h, 0, h) \cdot \mathfrak{s} = -\frac{2hw'(h)}{w(0) + 2w(h)} \ , \tag{5.198}$$

which is equivalent to the ordinary differential equation for the distance weight function

$$w'(h) = -\frac{w(0) + 2w(h)}{2h} \ . \tag{5.199}$$

However, this ODE is inconsistent with the requirements $0 < w(0) < \infty$, $w'(0) = 0$. Thus for no reasonable distance weight function, the Shepard approximant yields a consistent approximation of first derivatives. □

Theorem 5.4. *The AMLS method with linear basis leads to inconsistent second derivatives.*

Proof. We consider the regular three point grid $X = (-h, 0, h)$. Here the linear system (4.12) yields a unique stencil $\mathfrak{s} = \frac{1}{h^2}(1, -2, 1)^T$ for the second derivative at the origin. This stencil

has to be obtained from the AMLS function with linear basis for it to have a consistent second derivative.

We consider the expression for the second derivative of the the AMLS function (5.99). Using the abbreviations defined in Section 5.3.3, we obtain for the specific three point grid

$$\begin{array}{lll} s_0 = w_0 + 2w & s_0' = 0 & s_0'' = w_0'' + 2w'' \\ s_1 = 0 & s_1' = 2w'h & s_1'' = 0 \\ s_2 = 2wh^2 & s_2' = 0 & s_2'' = 2w''h^2 \\ r_0 = w_0u_0 + wu^+ & r_0' = w'u^- & r_0'' = w_0''u_0 + w''u^+ \\ r_1 = whu^- & r_1' = w'hu^+ & r_1'' = w''hu^- \end{array} \tag{5.200}$$

Additionally to u^+ and u^-, we define

$$u^c = u(h) - 2u(0) + u(-h) \ . \tag{5.201}$$

This leads to the following expressions

$$S'S^{-1} = \begin{pmatrix} 0 & \frac{s_1'}{s_2} \\ \frac{s_1'}{s_0} & 0 \end{pmatrix} = \begin{pmatrix} 0 & \frac{w'}{wh} \\ \frac{2w'h}{w_0+2w} & 0 \end{pmatrix} \tag{5.202}$$

$$S''S^{-1} = \begin{pmatrix} \frac{s_0''}{s_0} & 0 \\ 0 & \frac{s_2''}{s_2} \end{pmatrix} = \begin{pmatrix} \frac{w_0''+2w''}{w_0+2w} & 0 \\ 0 & \frac{w''}{w} \end{pmatrix} \tag{5.203}$$

$$\mathbf{r}' - \left(S'S^{-1}\right)\mathbf{r} = \begin{pmatrix} 0 \\ \frac{w_0w'h}{w_0+2w}u^c \end{pmatrix} \tag{5.204}$$

$$\left(S'S^{-1}\right)\left(\mathbf{r}' - \left(S'S^{-1}\right)\mathbf{r}\right) = \begin{pmatrix} \frac{w_0(w')^2}{w(w_0+2w)}u^c \\ 0 \end{pmatrix} \tag{5.205}$$

$$\mathbf{r}'' - \left(S''S^{-1}\right)\mathbf{r} = \begin{pmatrix} \frac{w_0w''-w_0''w}{w_0+2w}u^c \\ 0 \end{pmatrix} \tag{5.206}$$

Altogether, the vectors $\mathbf{a}'$ and $\mathbf{a}''$ become

$$\mathbf{a}' = S^{-1}\left(\mathbf{r}' - \left(S'S^{-1}\right)\mathbf{r}\right) = \begin{pmatrix} 0 \\ \frac{w_0w'}{2wh(w_0+2w)}u^c \end{pmatrix} \tag{5.207}$$

$$\mathbf{a}'' = S^{-1}\left(\left(\mathbf{r}'' - \left(S''S^{-1}\right)\mathbf{r}\right) - 2\left(S'S^{-1}\right)\left(\mathbf{r}' - \left(S'S^{-1}\right)\mathbf{r}\right)\right) \tag{5.208}$$

$$= \begin{pmatrix} \frac{w_0ww''-w_0''w^2-2w_0(w')^2}{w(w_0+2w)^2}u^c \\ 0 \end{pmatrix} \tag{5.209}$$

This leads to the second derivative

$$
\begin{aligned}
(\Pi u)''(0) =& 2 \begin{pmatrix} 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ \frac{w_0 w'}{2wh(w_0+2w)} u^c \end{pmatrix} && (5.210)\\
&+ \begin{pmatrix} 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} \frac{w_0 w w'' - w_0'' w^2 - 2w_0 (w')^2}{w(w_0+2w)^2} u^c \\ 0 \end{pmatrix} \\
=& 2 \frac{w_0 w'}{2wh(w_0+2w)} u^c + \frac{w_0 w w'' - w_0'' w^2 - 2w_0 (w')^2}{w(w_0+2w)^2} u^c && (5.211)\\
=& \frac{w_0 w'(w_0+2w) + h(w_0 w w'' - w_0'' w^2 - 2w_0 (w')^2)}{wh(w_0+2w)^2} u^c && (5.212)
\end{aligned}
$$

The approximation can only be consistent, if this expression equals $\frac{1}{h^2} u^c$, in particular for small values of h. For this case we have the asymptotic expansions

$$
w \cong w_0 \qquad w' \cong -w_0'' h \qquad w'' \cong w_0'' \tag{5.213}
$$

Inserting these into (5.212) yields the expression

$$
\begin{aligned}
(\Pi u)''(0) &\cong \frac{w_0(-w_0'' h)(3w_0) + h(w_0^2 w_0'' - w_0'' w_0^2 - 2w_0(-w_0'' h)^2)}{w_0 (3w_0)^2 h} u^c && (5.214)\\
&= \frac{(-w_0'' h)(3w_0) h - 2(-w_0'' h)^2 h^2}{(3w_0)^2} \frac{1}{h^2} u^c && (5.215)\\
&= \left(\lambda - 2\lambda^2\right) \frac{1}{h^2} u^c \ , && (5.216)
\end{aligned}
$$

where

$$
\lambda = \frac{-w_0'' h^2}{3w_0} \ . \tag{5.217}
$$

However, there is no $\lambda \in \mathbb{R}$ which satisfies $\lambda - 2\lambda^2 = 1$. Hence the stencil is inconsistent for any possible distance weight function. □

We have shown that if the polynomial degree of P_ν is lower than the derivative taken, an inconsistent approximation is obtained, even for a regular one dimensional three point grid. As in any higher space dimensions the one dimensional problem can be recovered, there is no hope that derivatives higher than the polynomial degree of P_ν can be approximated.

Theorem 5.2 together with Theorems 5.3 and 5.4 can be summarized in the following

Corollary 5.5. *If the distance weight function is sufficiently smooth, the AMLS method with polynomial degree n approximates derivatives up to order n consistently, and only these.*

5.3.7 Special Case: One Sided Derivative

With respect to Neumann boundary conditions, but also for upwinding for advection terms, when applying the presented methods to other problems, it is of importance to consider one sided grids $X = (0, x_1, \ldots, x_m)$, where $x_i > 0$, and the derivative is approximated in the origin. We consider the grid $X = (0, h, 2h)$ and the one-sided first derivative at the origin.

Linear Basis

We consider the AMLS function with linear basis. Using the abbreviations as in the previous sections, we have

$$\begin{array}{ll} s_0 = 1 + w + w^4 & r_0 = u_0 + wu_1 + w^4 u_2 \\ s_1 = hw(1 + 2w^3) & r_1 = hw(u_1 + 2w^3 u_2) \\ s_2 = h^2 w^2 (1 + 4w^3) & r_2 = h^2 w(u_1 + 4w^3 u_2) \end{array} , \tag{5.218}$$

where $w = w(h) = \exp\left(-\frac{\alpha}{2}h^2\right)$.

According to equation (5.152), the first derivative of the AMLS function with linear basis is

$$(\Pi u)'(0) = (\mathbf{b}')^T S^{-1}\mathbf{r} + \alpha \mathbf{b}^T S^{-1}(\mathbf{r}^+ - S^+ S^{-1}\mathbf{r}) \tag{5.219}$$

$$\begin{aligned} = &\frac{1}{s_0 s_2 - s_1^2}\left(\begin{pmatrix} 0 & 1 \end{pmatrix}\begin{pmatrix} s_2 & -s_1 \\ -s_1 & s_0 \end{pmatrix}\begin{pmatrix} r_0 \\ r_1 \end{pmatrix}\right. \\ &\left. + \alpha \begin{pmatrix} 1 & 0 \end{pmatrix}\begin{pmatrix} s_2 & -s_1 \\ -s_1 & s_0 \end{pmatrix}\begin{pmatrix} 0 \\ q \end{pmatrix}\right) \end{aligned} \tag{5.220}$$

where

$$q = r_2 - \begin{pmatrix} s_2 & s_3 \end{pmatrix} S^{-1}\begin{pmatrix} r_0 \\ r_1 \end{pmatrix} \tag{5.221}$$

$$= \frac{(s_0 s_2 - s_1^2)r_2 + (s_1 s_2 - s_0 s_3)r_1 + (s_1 s_3 - s_2^2)r_0}{s_0 s_2 - s_1^2} \tag{5.222}$$

Inserting the expressions (5.218) into (5.220) we obtain

$$s_0 s_2 - s_1^2 = h^2 w(1 + 4w^3 + w^4) \tag{5.223}$$

$$s_1 s_3 - s_2^2 = 2h^4 w^5 \tag{5.224}$$

$$s_1 s_2 - s_0 s_3 = h^3 w(1 + 8w^3 + 3w^4) \tag{5.225}$$

$$-s_1 r_0 + s_0 r_1 = hw \begin{pmatrix} -(1 + 2w^3) \\ 1 - w^4 \\ 2w^3 + w^4 \end{pmatrix}^T \mathbf{u} \tag{5.226}$$

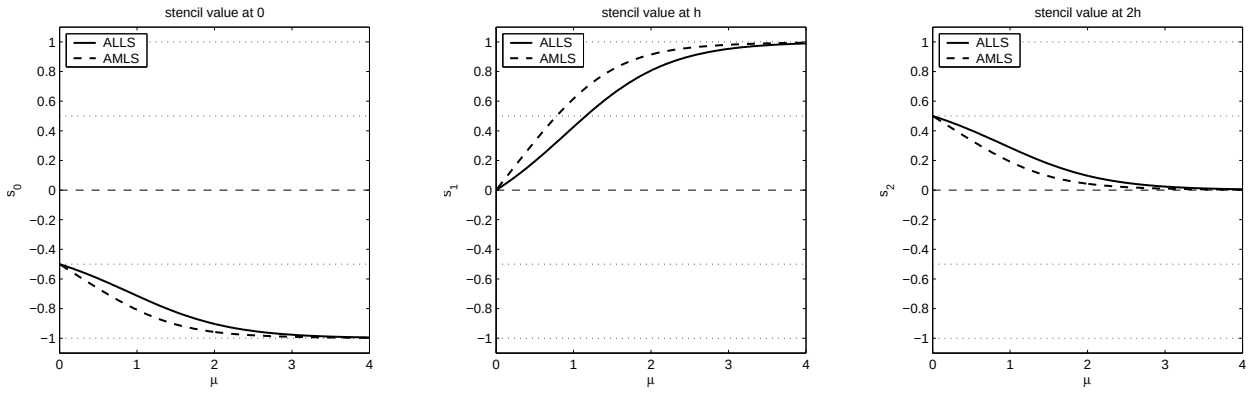

Figure 5.3: Stencil entries for one sided first derivative of AMLS function with linear basis

$$q = \frac{2h^2w^4}{1+4w^3+w^4} \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix}^T \mathbf{u} \tag{5.227}$$

Altogether

$$(\Pi u)'(0) = \frac{1}{h(1+4w^3+w^4)} \left(\begin{pmatrix} -(1+2w^3) \\ 1-w^4 \\ 2w^3+w^4 \end{pmatrix} - \alpha h^2 \frac{2w^4(1+2w^3)}{1+4w^3+w^4} \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix} \right)^T \mathbf{u} \tag{5.228}$$

As in Section 5.3.5, we introduce the scaling parameter $\mu = \alpha h^2$, and thus obtain

$$(\Pi u)'(0) = \frac{1}{h} \left(\mathfrak{s}_L + \mu \mathfrak{s}_\Delta \right)^T \mathbf{u} \ , \tag{5.229}$$

where

$$\mathfrak{s}_L = \frac{1}{1+4w^3+w^4} \begin{pmatrix} -(1+2w^3) \\ 1-w^4 \\ 2w^3+w^4 \end{pmatrix} \tag{5.230}$$

and

$$\mathfrak{s}_\Delta = -\frac{2w^4(1+2w^3)}{(1+4w^3+w^4)^2} \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix} \tag{5.231}$$

Figure 5.3 shows the value of the first derivative stencil multiplied with the mesh size h, in dependence on the scaling factor μ. As in Section 5.3.5, the larger the value of μ, the more localized is the approach. The solid lines show the LLS stencil $\mathfrak{s}_L$, as given by (5.230), the dashed lines represent the MLS stencil $\mathfrak{s}_L + \mu\mathfrak{s}_\Delta$, where the difference $\mathfrak{s}_\Delta$ is given by equation (5.231).

For $\mu \to \infty$, the approach goes to the standard two point stencil $\frac{1}{h}(-1, 1, 0)$. For $mu = 0$, one obtains the de-localized $\frac{1}{h}(-\frac{1}{2}, 0, \frac{1}{2})$ stencil. Obviously, such stencils can yield unstable

checkerboard effects. Thus, a reasonable decay of the distance weight function is necessary. As opposed to the symmetric grid considered in Section 5.3.5, the difference between MLS and LLS approach is not as significant here. The moving approach yields slightly more significant values, but here all values at $x = h$ and $x = 2h$ stay positive for all μ.

Quadratic Basis

Including quadratic terms into the basis forces the stencil to satisfy 3 constraints. As our grid contains only 3 points, the stencil is uniquely defined. It is the second order one sided first derivative stencil $\frac{1}{h}(-\frac{3}{2}, 2, -\frac{1}{2})$. Since it is uniquely defined, there is no difference between local and moving approach. Note that the stencil is not positive. As shown in Section 4.2.2, it is impossible for a higher order one sided first derivatives to yield a positive stencil.

5.4 Derivatives by Interpolating Moving Least Squares

The IMLS method as a method for approximating derivatives requires more care due to the singularity in the distance weight function. In [Kun01], Kunle presented how to obtain derivative approximations in a stable manner, by considering the asymptotical behavior of the singular systems at the data points. The IMLS method as a method for approximating derivatives in one space dimension has been investigated by Sonar in [Son05]. Explicit formulae for the derivatives at the data points are provided, and their values for specific regular three to seven point grids are computed. The quality of the derivative approximations are investigated numerically.

Typically in the IMLS method, one considers distance weight functions of the form

$$w(d) = d^{-\alpha} , \quad \alpha = 2, 4, \dots . \tag{5.232}$$

In [Kle98, Chap. III] as well as in [Son05] the authors point out the importance of α being an even integer. However, with a view on the ILLS method, as presented in Section 5.1.2, for the pure approximation of derivatives from function values, distance weight functions

$$w(d) = |d|^{-\alpha} \tag{5.233}$$

with any $\alpha > 0$ are admissible. In general, one will require at least $\alpha \geq 2$, since otherwise far away points will obtain too much weight.

Also, most results about consistency of the derivative approximations apply for any singular function w satisfying

$$d^{\alpha} w(d) \to \infty \quad \text{as } d \to 0 . \tag{5.234}$$

Of course, in order to differentiate the IMLS function, we assume the distance weight function to be sufficiently smooth.

Remark 5.9. The IMLS method requires more care than the AMLS case due to the singularity of the distance weight function. Function values as well as approximations to derivatives are in the first place not defined at data points $\mathbf{x}_k$. However, for function values and also for values of derivatives (see [Kun01]) the limit for $\mathbf{x} \to \mathbf{x}_k$ exists, and thus defines values at the data points themselves.

5.4.1 Consistency for Varying Polynomial Degree

As for the AMLS method, the IMLS function can be differentiated as often as the distance weight function can be. However, consistent approximations to derivatives are only obtained, if the polynomial degree of P_ν equals at least the derivative taken. This result mirrors the AMLS case, although the reasoning is slightly different in the interpolating case.

In the following, we outline the results for the one dimensional case, as presented in [Son05]. The consistency results transfer also to the multidimensional case. The analysis of specific stencils, of course, touched 1d phenomena only.

Abbreviations

As for the AMLS method (see Section 5.3.2), we introduce abbreviatory expressions. The point of approximation is x_0. For further simplicity, we place w.l.o.g. x_0 in the origin. For the case $x_0 \neq 0$, one has to consider the distances to the point x_0 in the expressions $\bar{x}_i$. The expressions are

$$\bar{s}_k = \sum_{i=1}^{m} w_i \bar{x}_i^k \qquad \bar{r}_k = \sum_{i=1}^{m} w_i \bar{x}_i^k u_i^0 \tag{5.235}$$

$$\bar{s}'_k = \sum_{i=1}^{m} w'_i \bar{x}_i^k \qquad \bar{r}'_k = \sum_{i=1}^{m} w'_i \bar{x}_i^k u_i^0 \tag{5.236}$$

where

$$\bar{x}_i = x_i - x_0 \tag{5.237}$$

$$u_i^0 = u(x_i) - u(x_0) \tag{5.238}$$

$$w_i = w(0 - x_i) = w(x_i) \tag{5.239}$$

$$w'_i = w'(0 - x_i) = w'(-x_i) \tag{5.240}$$

Note that the expressions $\bar{s}_k$ and $\bar{r}_k$ differ from their AMLS counterparts (5.111) and (5.113) in two aspects:

- The sum is not taken over the point of evaluation (here $x_0 = 0$), since in the IMLS method the weight is singular there.
- The r_k contain the difference $u_i^0 = u(x_i) - u(x_0)$, instead of solely the values u_i as in the AMLS method.

Also note the analogy to the ILLS case presented in Section 5.1.2.

Shepard's Method

Shepard's method, i.e. interpolating moving least squares with a constant polynomial basis, does not constitute in a consistent approximation of first or higher derivatives.

Theorem 5.6. *Let $\alpha \in \mathbb{N}$ be even and $w(d) = d^{-\alpha}$. Then the Shepard interpolant satisfies*

$$\frac{d^\nu}{dx^\nu} Su(x_k) = 0 \quad \forall \nu = 1, 2, \ldots, \alpha - 1 \tag{5.241}$$

Proof. The proof is given in [Son05]. The derivative at a point $x_k + h$ is calculated by applying the quotient rule to (5.65) directly. Asymptotically (for h small) various cancellations take place, yielding the desired result for the first and second derivative. Analogous cancellations appear for higher order derivatives by an interactive application of the same procedure. □

In particular, this result contains the well known *flat spot phenomenon* of the Shepard interpolant (see [LS86]). As we are interested in second derivatives, one could ask for the Shepard interpolant's second derivative in the case $\alpha = 2$. Indeed, this second derivative exists, but it is inconsistent.

Remark 5.10. In the case $\alpha = 2$ the second derivative of the Shepard interpolant at a data point is

$$(Su)''(x_k) = 2 \sum_{i \neq k} \frac{u_i - u_k}{(x_i - x_k)^2} \tag{5.242}$$

For the three points $(-h, 0, h)$ this expression yields the stencil $\frac{1}{h^2}(2, -4, 2)$, which is an inconsistent approximation to the second derivative.

Remark 5.11. The Shepard interpolant can only be differentiated up to order α. All derivatives up to order $\alpha - 1$ equal zero. As opposed the Shepard approximant obtained via the AMLS method can be differentiated as often as the distance weight function can be. Derivatives yield nonzero values. However, for both the AMLS and the IMLS method, the Shepard function yields an inconsistent approximation to any derivative.

Linear Basis

If the polynomial basis in enriched by linear functions, first order derivatives are approximated consistently. Second order derivatives, however, are not consistently approximated.

Theorem 5.7. *The first derivative of the interpolant Πu with linear basis exists and is given by*

$$(\Pi u)'(x_0) = \frac{\sum_{i=1}^m w_i(x_0) \cdot (x_i - x_0)(u_i - u_0)}{\sum_{i=1}^m w_i(x_0) \cdot (x_i - x_0)^2} \ , \tag{5.243}$$

or using the above abbreviations

$$(\Pi u)'(0) = \frac{\bar{r}_1}{\bar{s}_2} \ . \tag{5.244}$$

It is a consistent approximation to $u'(x_0)$.

Proof. The formula is proved in [Son05] and consistency is checked. □

Note that this equals the result for the local least squares approach presented in Section 5.1.2. There are no additional terms due to the differentiation of the IMLS coefficient vector $\mathbf{a}$.

Theorem 5.8. *The second derivative of the interpolant with linear basis is inconsistent.*

Proof. The proof is given in [Son05]. Indeed, for the three points $(-h, 0, h)$ this expression yields the stencil $\frac{\alpha}{h^2}(1, -2, 1)$, which for any valid value of α is an inconsistent approximation to the second derivative. □

Quadratic Basis

The interpolating moving least squares method with a polynomial basis up to second order yields consistent approximations to first and second derivatives. Again we consider w.l.o.g. $x_0 = 0$.

Theorem 5.9. *The first and second derivatives of the interpolant Πu with quadratic basis exist and are given by*

$$(\Pi u)'(0) = \beta_2 \tag{5.245}$$

$$(\Pi u)''(0) = 2\left(\beta_3 + \beta_2'\right) \ , \tag{5.246}$$

where

$$\beta_2 = \frac{\bar{s}_4\bar{r}_1 - \bar{s}_3\bar{r}_2}{\bar{s}_2\bar{s}_4 - \bar{s}_3^2} \tag{5.247}$$

$$\beta_3 = \frac{-\bar{s}_3\bar{r}_1 + \bar{s}_2\bar{r}_2}{\bar{s}_2\bar{s}_4 - \bar{s}_3^2} \tag{5.248}$$

$$\beta_2' = \frac{\bar{s}_4'\bar{r}_1 + \bar{s}_4\bar{r}_1' - \bar{s}_3'\bar{r}_2 - \bar{s}_3\bar{r}_2'}{\bar{s}_2\bar{s}_4 - \bar{s}_3^2} - \frac{(\bar{s}_2'\bar{s}_4 + \bar{s}_2\bar{s}_4' - 2\bar{s}_3\bar{s}_3')(\bar{s}_4\bar{r}_1 - \bar{s}_3\bar{r}_2)}{(\bar{s}_2\bar{s}_4 - \bar{s}_3^2)^2} \tag{5.249}$$

with the above notations for $\bar{s}_k$, $\bar{r}_k$, $\bar{s}'_k$, $\bar{r}'_k$. *They are consistent approximations to* $u'(0)$, *respectively to* $u''(0)$.

Proof. The proof is sketched in [Son05]. The expressions β_2 and β_3, which are evaluated at 0 here, are functions of x. Obviously, β'_2 is the derivative of $\beta_2(x)$, evaluated at 0. The derivative $\beta'_2(0)$ is obtained by considering a point x close to x_0 (in which the distance weight function of x_0 is regular) and considering the limit $x \to x_0$. □

A comparison with the LLS method presented in Section 5.1.4 yields that the expression for the first derivative equals the ILLS approximation. For the second derivative, the term β_3 is the LLS approximation, while the term β'_2 is an additional term due to the differentiation of the MLS coefficient vector $\mathbf{a}$.

Unlike the AMLS method, in the IMLS case, derivatives for the linear basis equal the LLS expressions. Only the second derivative to the IMLS function with quadratic basis differs from the ILLS expression. Thus we consider this case in more detail.

Specific Distance Weight Function

If the distance weight function is $w(d) = |d|^{-\alpha}$, then its derivative is

$$w'(d) = -\alpha \frac{1}{d} |d|^{-\alpha} = -\alpha \frac{1}{d} w(d) \tag{5.250}$$

Consequently $w'_i = w'(-x_i) = \alpha \frac{1}{x_i} w$, and thus

$$\bar{s}'_k = \alpha \bar{s}_{k-1} \quad \forall k \geq 1 \tag{5.251}$$

$$\bar{r}'_k = \alpha \bar{r}_{k-1} \quad \forall k \geq 1 \tag{5.252}$$

Employing these relations, expression (5.249) in the second derivative of the IMLS function with quadratic basis becomes

$$\beta'_2 = \alpha \left(\frac{\bar{s}_4 \bar{r}_0 - \bar{s}_2 \bar{r}_2}{\bar{s}_2 \bar{s}_4 - \bar{s}_3^2} - \frac{(\bar{s}_1 \bar{s}_4 - \bar{s}_2 \bar{s}_3)(\bar{s}_4 \bar{r}_1 - \bar{s}_3 \bar{r}_2)}{(\bar{s}_2 \bar{s}_4 - \bar{s}_3^2)^2} \right) \tag{5.253}$$

5.4.2 Special Case: Symmetric Grid

Consider the grid $X = (-h_m, \ldots, -h_1, 0, h_1, \ldots, h_m)$. We calculate the IMLS approximation to the second derivative, when a quadratic basis is used. Similarly to the AMLS case, as analyzed in Section 5.3.3, we obtain various cancellations. In particular

$$\begin{array}{llll} \bar{s}_k = & 0 & \bar{r}_k = & \sum w_i h_i^k u_i^- \quad \forall k = 1, 3, \ldots \\ \bar{s}_k = & 2 \sum w_i h_i^k & \bar{r}_k = & \sum w_i h_i^k u_i^c \quad \forall k = 2, 4, \ldots \end{array} \tag{5.254}$$

where

$$u_i^- = u(h_i) - u(-h_i) \quad \text{and} \quad u_i^c = u(h_i) - 2u(0) + u(-h_i) \ . \tag{5.255}$$

This yields

$$\beta_3 = \frac{\bar{r}_2}{\bar{s}_4} \tag{5.256}$$

$$\beta_2' = \alpha \left(\frac{\bar{s}_4 \bar{r}_0 - \bar{s}_2 \bar{r}_2}{\bar{s}_2 \bar{s}_4} \right) = \alpha \left(\frac{\bar{r}_0}{\bar{s}_2} - \frac{\bar{r}_2}{\bar{s}_4} \right) \tag{5.257}$$

and thus

$$(\Pi u)''(0) = 2 \left(\frac{\bar{r}_2}{\bar{s}_4} + \alpha \left(\frac{\bar{r}_0}{\bar{s}_2} - \frac{\bar{r}_2}{\bar{s}_4} \right) \right) \tag{5.258}$$

For the special case of a regular symmetric five point grid $X = (-2h, -h, 0, h, 2h)$, the weights are $w_1 = h^{-\alpha}$ and $w_2 = 2^{-\alpha} w_1$. We denote $t = 2^{-\alpha}$. With these notations, the terms become

$$\frac{\bar{r}_0}{\bar{s}_2} = \frac{1}{2h^2} \frac{u_1^c + t u_2^c}{1 + 4t} \tag{5.259}$$

$$\frac{\bar{r}_2}{\bar{s}_4} = \frac{1}{2h^2} \frac{u_1^c + 4t u_2^c}{1 + 16t} \tag{5.260}$$

yielding the derivative

$$(\Pi u)''(0) = \frac{1}{h^2} \left(\frac{u_1^c + 4t u_2^c}{1 + 16t} + \alpha \left(\frac{u_1^c + t u_2^c}{1 + 4t} - \frac{u_1^c + 4t u_2^c}{1 + 16t} \right) \right) \tag{5.261}$$

$$= \frac{1}{h^2} \left(\frac{u_1^c + 4t u_2^c}{1 + 16t} + \alpha \left(\frac{3t(4u_1^c - u_2^c)}{(1 + 4t)(1 + 16t)} \right) \right) \tag{5.262}$$

$$= \frac{1}{h^2} \left(\mathfrak{s}_L + \alpha \mathfrak{s}_\Delta \right)^T \mathbf{u} \tag{5.263}$$

Here the stencils (without the $\frac{1}{h^2}$ scaling) are

$$\mathfrak{s}_L = \frac{1}{1 + 16t} \begin{pmatrix} 4t \\ 1 \\ -2(1 + 4t) \\ 1 \\ 4t \end{pmatrix} \tag{5.264}$$

and

$$\mathfrak{s}_\Delta = \frac{3t}{(1 + 4t)(1 + 16t)} \begin{pmatrix} -1 \\ 4 \\ -6 \\ 4 \\ -1 \end{pmatrix} \tag{5.265}$$

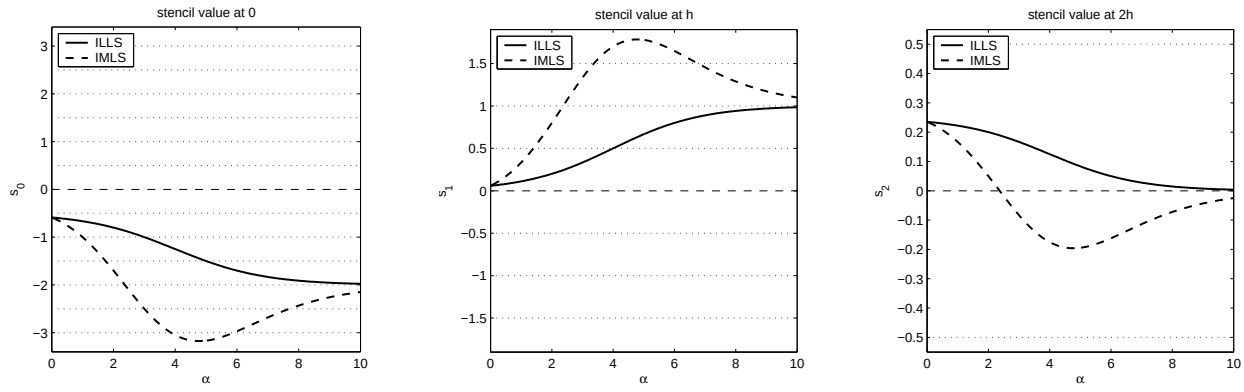

Figure 5.4: Stencil entries for second derivative of IMLS function with quadratic basis

Figure 5.4 shows the stencil entries corresponding to the positions $x = 0$, $x = h$ and $x = 2h$ for the second derivative of the IMLS function with quadratic basis, in dependence of the parameter α. Similarly to the AMLS case with a Gaussian distance weight function, the parameter α measures the "locality" of w. The larger the values of α, the more weight is given to closer points. One can see this fact, since the stencil entry at $x = 2h$ decays to zero for α increasing. For $\alpha \to 0$, the limiting stencil is $\frac{1}{17}(4, 1, -10, 1, 4)\frac{1}{h^2}$. The solid lines show the components of the vector $\mathfrak{s}_L$, i.e. the ILLS approximation, as given by (5.264). The dashed line shows $\mathfrak{s}_L + \alpha\mathfrak{s}_\Delta$, i.e. the IMLS approximation, where the difference is given by (5.265).

One can also observe:

- The stencil values depend smoothly on the parameter α. Thus, if for both $\alpha = 4$ and $\alpha = 6$ the obtained stencils are satisfying, nothing speaks against considering intermediate values $4 < \alpha < 6$. There is no need to restrict α to even integers.
- For narrow weight functions ($\alpha \to \infty$), the approximations approach the standard central three point stencils $(0, 1, -2, 1, 0)\frac{1}{h^2}$.
- The transfer from wide to narrow takes place monotonically for the ILLS approaches, while the IMLS approach "overshoots" for moderate values of α.
- The ILLS method yields a positive stencil for every value of α. The IMLS approach, however, yields negative stencil values at $x = 2h$ for $\alpha > 2.37$.

The comparison between ILLS and IMLS approach to approximate second derivatives show the same phenomena as the approximating case, as compared in Section 5.3.5. For the same reasons, the ILLS approach seems preferable over the IMLS approach for approximating derivatives (where they make a difference at all).

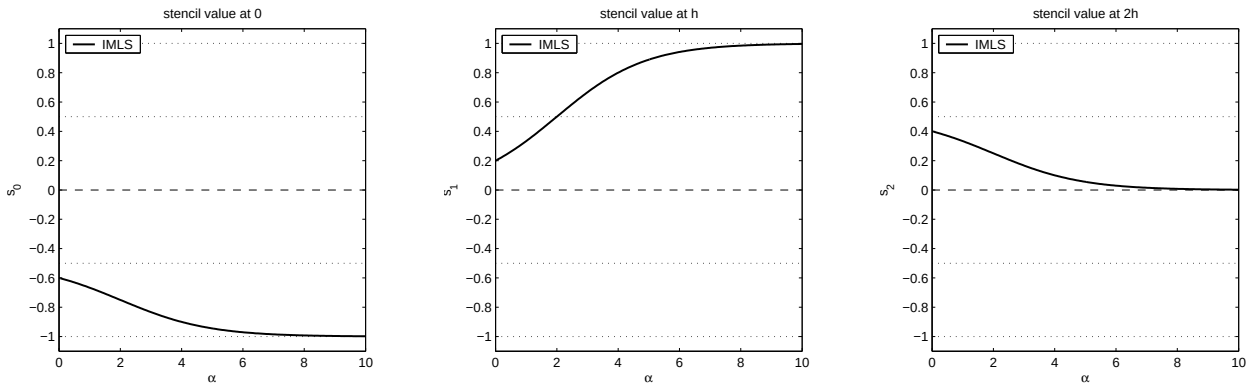

Figure 5.5: Stencil entries for one sided first derivative of IMLS function with linear basis

5.4.3 Special Case: One Sided Derivative

As in Section 5.3.7 for the AMLS method, we consider first derivatives of the one sided three point grid $X = (0, h, 2h)$.

Linear Basis

The expression for the first derivative of the linear basis IMLS function is given by (5.244). For the given three point grid it is

$$(\Pi u)'(0) = \frac{\bar{r}_1}{\bar{s}_2} = \frac{u_1^0 + 2tu_2^0}{1+4t} \,, \tag{5.266}$$

where as in Section 5.4.2, $t = 2^{-\alpha}$. Hence, the stencil is

$$\mathfrak{s} = \frac{1}{1+4t} \begin{pmatrix} -(1+2t) \\ 1 \\ 2t \end{pmatrix} \tag{5.267}$$

As stated in Section 5.4.1, these is no difference between moving and local approach.

Figure 5.5 shows the value of the first derivative stencil multiplied with the mesh size h, in dependence on the parameter α. The behavior is similar to results for the AMLS approach, as shown in Section 5.3.7. One significant difference is that here for small values of α, i.e. wide open weight functions, the stencil approaches $\frac{1}{h}(-\frac{3}{2}, \frac{1}{5}, \frac{2}{5})$. As for the AMLS approach, the stencil approximation is fine, given the value of α is sufficiently large.

Quadratic Basis

As for the AMLS case (Section 5.3.7), the case of a quadratic basis yields the uniquely defined second order stencil $\frac{1}{h}(-\frac{3}{2}, 2, -\frac{1}{2})$, for both moving and local approach.

5.4.4 IMLS in Matrix Notation

Due to the singularity of the distance weight function, differentiating the IMLS function at the data points requires a rigorous asymptotic treatment, as outlined by Sonar [Son05]. However, one can at least formally put the IMLS derivative approximation in the framework presented in Section 5.3.1 for the AMLS method.

Let $\mathbf{b}(\mathbf{x})$ be the vector of the polynomial basis. As outlined in Section 5.1.2 for the ILLS approximation, and as it can also seen in the proofs by Sonar, the IMLS derivatives firstly lack the first, constant, component of the basis, and secondly in any summation the central point is not considered. For instance, in 1d, for the quadratic basis, the basis vector is $\mathbf{b}(\mathbf{x}) = (x, x^2)^T$. Consequently, also the derivatives $\mathbf{b}'$ and $\mathbf{b}''$ are reduced by their first component. Thus, for the IMLS derivatives, the coefficient vector $\mathbf{a}(\mathbf{x})$ is computed like in the AMLS method, however lacking the first component.

The approximations to the first and second derivatives (5.82) to (5.86), as derived in Section 5.2.4, become here

$$(\Pi' u) = (\mathbf{b}')^T \cdot \mathbf{a} \qquad \text{local} \tag{5.268}$$

$$(\Pi u)' = (\mathbf{b}')^T \cdot \mathbf{a} \qquad \text{global} \tag{5.269}$$

and for second derivatives

$$(\Pi'' u) = (\mathbf{b}'')^T \cdot \mathbf{a} \qquad \text{local} \tag{5.270}$$

$$(\Pi' u)' = (\mathbf{b}'')^T \cdot \mathbf{a} + (\mathbf{b}')^T \cdot \mathbf{a}' \qquad \text{mixed} \tag{5.271}$$

$$(\Pi u)'' = (\mathbf{b}'')^T \cdot \mathbf{a} + 2\,(\mathbf{b}')^T \cdot \mathbf{a}' \qquad \text{global} \tag{5.272}$$

Since the expressions are evaluated at the origin $\mathbf{x}_0 = 0$ the basis vector $\mathbf{b}(0) = (0, 0)$ does not yield any contribution, as it does for the AMLS method. As derived in Section 5.4.1, we see that for the first derivative the local and moving approach are equivalent. The second derivatives just have one additional term compared to the ILLS approximation. Unlike the AMLS method, the IMLS method does not include any second derivatives of the distance weight function.

The lengthy expression for the second derivative of the IMLS function, as presented in Theorem 5.9 can in the matrix notion be recovered as

$$\begin{aligned}(\Pi u)''(0) &= (\mathbf{b}'')^T \cdot \mathbf{a} + 2\,(\mathbf{b}')^T \cdot \mathbf{a}' \\ &= (0,2) \cdot \bar{S}^{-1}\bar{\mathbf{r}} + 2(1,0) \cdot \bar{S}^{-1}\left(\bar{\mathbf{r}}' - \bar{S}'\bar{S}^{-1}\bar{\mathbf{r}}\right)\end{aligned} \tag{5.273}$$

where

$$\bar{S} = \begin{pmatrix} \bar{s}_2 & \bar{s}_3 \\ \bar{s}_3 & \bar{s}_4 \end{pmatrix} \quad \text{and} \quad \bar{\mathbf{r}} = \begin{pmatrix} \bar{r}_1 \\ \bar{r}_2 \end{pmatrix} . \tag{5.274}$$

The mixed derivative approximation lies in between the local and the global approximation. In the plots presented in Figure 5.4 it would yield a non-monotonic behavior, as for the IMLS approach, however, it would not overshoot such significantly.

5.5 Comparison of the Approaches

In the previous sections we have analyzed the values of stencils for specific configurations. For both the approximating and the interpolating case we have compared local and moving approaches. The general outcome of this analysis was that the local least squares approach shows a nicer behavior in terms of stencil values. In Section 5.3.5 we have seen that the first and second AMLS derivative overshoot and can yield non-positive stencils, where the ALLS stencils are positive. In Section 5.4.2 we have seen that the second IMLS derivative shows the same features. Of course, the stencil values alone are not the whole truth, hence we will investigate the approximations in a Poisson problem in Section 5.5.2. However, with respect to the property of positive stencils, which we will investigate in more detail in Chapter 6, the local approaches are clearly preferable over the moving methods.

Not investigated so far has been the question, whether to prefer the approximating or the interpolating approach in the context of the Finite Pointset Method. Kunle observes [Kun01] that the approximating approach performs weak for large gradients, and proceeds with the interpolating approach thereafter. Kuhnert and Tiwari employ the ALLS approach [KT01, KT02a, KT03]. Fürst and Sonar use interpolating least squares for approximating derivatives [FS01].

A strict comparison between the AMLS and the IMLS method is difficult, since the methods use incomparable distance weight functions. Both the Gaussian (AMLS) and the inverse polynomial (IMLS) functions have a scaling parameter, which in some sense measures the locality of the approximation. Thus one could compare the graphs presented in Figure 5.2 and Figure 5.4, which both show the second derivative for the case of a quadratic polynomial basis. However, a reasonable comparison should use the same distance weight function. This is a problem for the MLS approaches, since the approaches are based on the regularity respectively singularity of the distance weight function. What can be done without problems is to compare the LLS approaches, as derived in Section 5.1. These do not put any requirements on the weight function. In Section 5.5.1 we compare these approaches for a regular symmetric grid.

Of more interest is of course the application of the approach for solving a Poisson test problem, which we will do in Section 5.5.2. In order to obtain an exhaustive comparison, we test all methods, including the moving approaches, without worrying too much about the somewhat “incorrect” behavior for wide stretched weight functions.

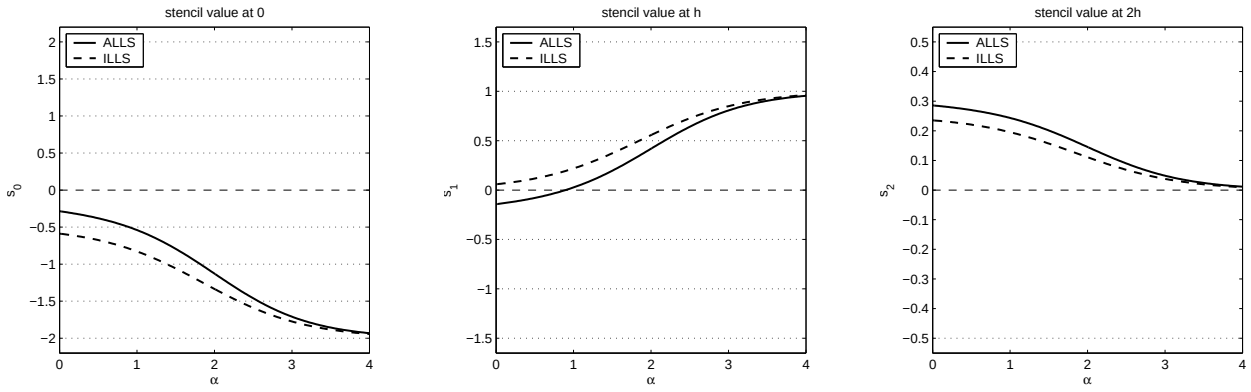

Figure 5.6: Stencil entries for second derivative of AMLS vs. IMLS function with quadratic basis

5.5.1 Comparison for Regular Symmetric Grid

We consider the local derivative approximations to the second derivative. These are given by equations (5.12) respectively (5.18). As in the previous sections, we consider the one dimensional, regular symmetric five point grid $X = (-2h, -h, 0, h, 2h)$. Although not intended for the interpolating approach, for the comparison, we use a Gaussian distance weight function $w(d) = \exp\left(-\frac{\alpha}{2}d^2\right)$ for both approximations. The approximation for the ALLS method is given by (5.182) and yields the stencil given by (5.193) as

$$\mathfrak{s} = \frac{1}{h^2}\frac{1}{h^2}\frac{1}{1+16w^3+18w^4}\begin{pmatrix} 4w^3+6w^4 \\ 1-6w^4 \\ -2(1+4w^3) \\ 1-6w^4 \\ 4w^3+6w^4 \end{pmatrix} \tag{5.275}$$

As in Section 5.3.5, we have the value $w = \exp\left(-\frac{\alpha}{2}\right)$. As calculated in Section 5.4.2, the IMLS approximation is

$$(\Pi u)''(0) = 2\frac{\bar{r}_2}{\bar{s}_4} = \mathfrak{s}_I \ , \tag{5.276}$$

which yields the stencil

$$\mathfrak{s}_I = \frac{1}{h^2}\frac{1}{1+16w^3}\begin{pmatrix} 4w^3 \\ 1 \\ -2(1+4w^3) \\ 1 \\ 4w^3 \end{pmatrix} \tag{5.277}$$

Figure 5.6 shows the stencil entries (without the $\frac{1}{h^2}$ scaling) corresponding to the positions $x = 0$, $x = h$ and $x = 2h$ for the second derivatives of the LLS functions with quadratic basis.

Both approaches are computed using a Gaussian distance weight, the values are computed in dependence of the parameter α. The solid lines show the components of the ALLS stencil $\mathfrak{s}_A$, as given by (5.275), the dashed lines show the corresponding ILLS components of $\mathfrak{s}_I$, as given by (5.277).

For large values of α, both approaches yield reasonable results. However, for small values of α, the ILLS approach is preferable, since the ALLS component at $x = h$ becomes negative for $\alpha < \frac{1}{2}\log(6)$, while the ILLS component stays positive. Also, the central entry is larger in absolute value in the interpolating case, which generally results in more stability.

5.5.2 Comparison for Poisson Test Problem

Up to now we have only investigated the various approaches in terms of approximating the derivative of a given function. Our main interest, however, is the inverse problem: Given a function f, find a function u, satisfying given boundary conditions, whose second derivative equals f.

We consider a one dimensional Poisson problem on $\Omega = (0, 1) \subset \mathbb{R}$

$$\begin{cases} u_{xx} = f & \text{in } (0,1) \\ u = 0 & \text{on } \{0,1\} \end{cases} \tag{5.278}$$

Obviously, the problem is of type (4.1), and can be discretized to a linear system

$$A \cdot \hat{\mathbf{u}} = \mathbf{f}, \tag{5.279}$$

where the components of the solution vector $\mathbf{u}$ are approximations to function values of the solution u at the data points.

We consider an equidistant point cloud $X = (0, h, \ldots, 1-h, 1)$, where $h = \frac{1}{n}$, i.e. we have $n-1$ interior and 2 boundary points. For every interior point we set up Laplace stencils. For both the approximating and the interpolating approach we consider a local (equations (5.84) and (5.270)), a mixed (equations (5.85) and (5.271)), and a global (equations (5.86) and (5.272)) approach. For each such case we compute a family of Laplace stencils, with the locality of the distance weight function μ as parameter. The thus obtained stencils, together with Dirichlet boundary conditions imposed at the two boundary points, form the system matrix A, which approximates the operator $\frac{\partial^2}{\partial x^2}$. Note that we use all points for the derivative approximation. For large values of μ, the points further away can be neglected, and the stencil will be close to the five point stencils considered previously. For small values of μ, however, the previous investigations for five point stencils are not valid here.

Each such constructed system matrix we check with respect to the following aspects:

- The **approximation error** when solving a system
  The matrix A is an approximation to the Laplace operator. Unless the solution u is a quadratic function, the discrete solution $\hat{\mathbf{u}}$ will not equal the correct solution. We consider the right hand side function

$$f(x) = -\pi^2 x \sin(\pi x) + 2\pi \cos(\pi x) \ . \tag{5.280}$$

  Together with the homogeneous Dirichlet boundary conditions, problem (5.278) has the unique solution

$$u(x) = x \sin(\pi x) \ . \tag{5.281}$$

  Due to their small size we can solve the arising linear systems exactly (up to machine accuracy), and consider the approximation error

$$\|\mathbf{u} - A^{-1}\mathbf{f}\|_2 \ . \tag{5.282}$$

  Here $\mathbf{u}$ is the vector of function values $u(x_i)$ of the correct solution, and $\mathbf{f}$ contains the function values $f(x_i)$.

- Its **condition number**:
  The condition number $\kappa(A) = \|A\|_2 \cdot \|A^{-1}\|_2$ measures to which extent errors in the system right hand side f (or in the matrix entries) are transfered to the solution of the system $A\hat{\mathbf{u}} = \hat{\mathbf{f}}$ (see [QSS00]). Large condition numbers mean that small disturbances in the data can yield large errors in the numerical solution. If a matrix has a large condition number, solving requires preconditioning. There is a large arsenal of preconditioning method available, which solve ill-conditioned linear system efficiently (see Chapter 7 on preconditioning). While sophisticated preconditioning techniques exist (such as algebraic multigrid, see Section 7.4.3), if one has the choice between two discretization matrices of the same partial differential equation, which yield comparable discretization errors, but have very different condition numbers, one will certainly prefer the better conditioned system.

- Whether it is an **M-matrix**:
  As described in Section 6.1, the M-matrix structure is in various respects favorable for matrices approximating the operator $-\Delta$. For every system matrix constructed, we check whether it is an M-matrix or not, by directly considering the entries of the matrix itself (L-matrix structure) and the entries of its inverse (nonnegative entries).

We discretize problem (5.278) with a grid of $n = 7$ intervals, as well as with a grid of $n = 70$ intervals. For the small system, Figure 5.7 shows the condition numbers of the system matrices, and Figure 5.8 shows the approximation errors. Analogously, Figure 5.9 and Figure 5.9 show condition numbers and approximation errors for the larger system. In each plot the solid line corresponds to the local least squares approach, the dashed line

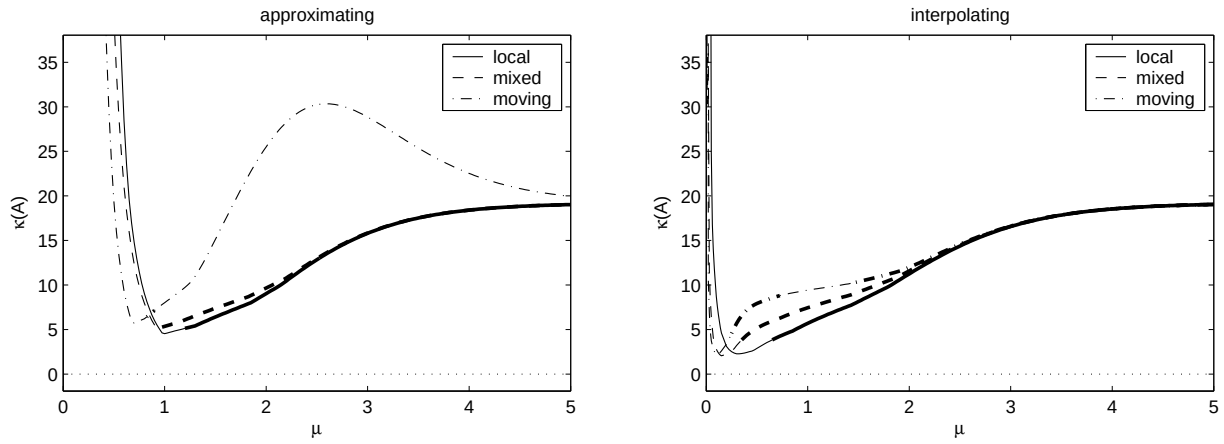

Figure 5.7: Condition numbers for 7 point least squares system

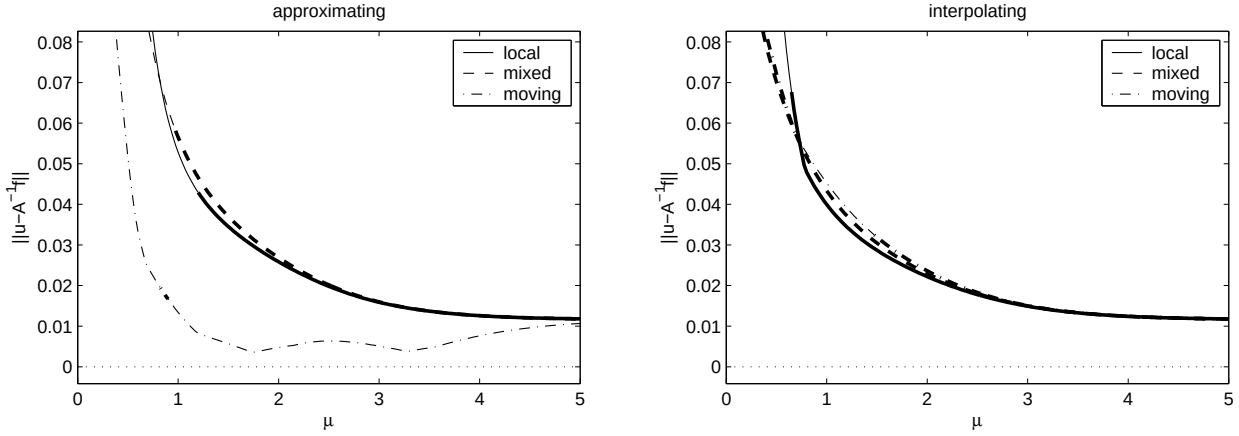

Figure 5.8: Errors for 7 point least squares system

belongs to the mixed local–global approach, and the dash–dotted line shows results for the moving least squares approach. In each plot, the line is fat, if the resulting matrix is an M-matrix, and thin, if it is not.

For each stencil construction, the distance weight function

$$w(d) = \exp\left(-\frac{\alpha}{2}d^2\right) = \exp\left(-\frac{\mu}{2}\left(\frac{d}{h}\right)^2\right) \tag{5.283}$$

is chosen. All quantities are plotted in dependence of the mesh size independent parameter μ. The limit $\mu \to 0$ represents constant distance weight functions. In this case the stencil approximation does not depend on the point of evaluation anymore. The resulting system matrices for $\mu = 0$ are singular, since multiple rows are linearly dependent. Consequently, for small values of μ, condition numbers and errors explode. Also note that in this case the results for the IMLS method are not really correct, since these normally rely on the singularity of the distance weight function, which is not present here. The limit $\mu \to \infty$ corresponds to

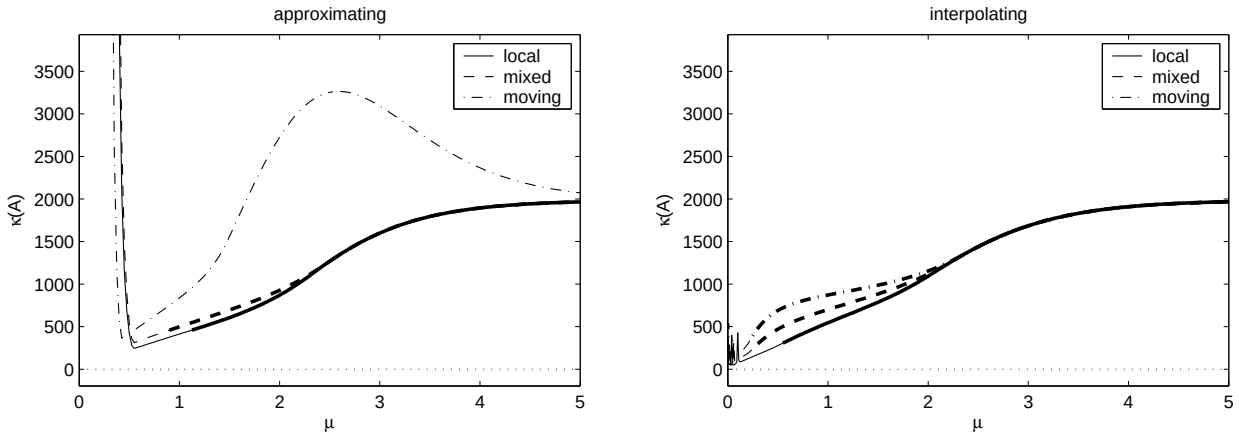

Figure 5.9: Condition numbers for 70 point least squares system

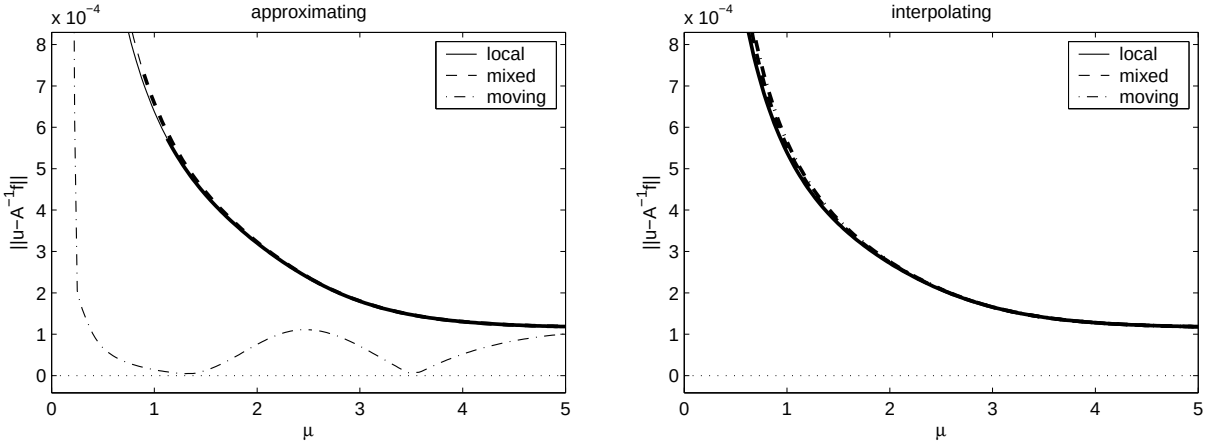

Figure 5.10: Errors for 70 point least squares system

a completely localized distance weight function. Consequently, all approximations converge to one and the same system matrix, consisting of the standard three point Laplace stencil $\frac{1}{h^2}(1, -2, 1)$.

Note that for all approximations, the stencil values depend continuously on the parameter μ. Consequently, the condition number (with respect to the spectral norm, analogous results can be observed for the maximum norm) and the approximation error depend continuously on μ, as long as the matrices do not become singular for some $\mu > 0$.

The results give rise to the following observations:

- The results are, up to the scaling, nearly independent of the mesh size h. The locality parameter $\mu = \alpha h^2$ is already scaled, the condition numbers scale roughly with $\frac{1}{h^2}$, while the approximation errors scale with h^2. Significant dependences on the mesh size are

  - For larger systems, reasonable condition numbers are achieved for smaller values of μ.
  - The different interpolating approaches become very similar for larger systems.
  - The approximating moving least squares method performs even better for larger systems.

- The interpolating approaches do not differ significantly.

- All system matrices are best conditioned for μ between 0.5 and 1. For larger μ the condition numbers increase, however, stay in a reasonable area. For smaller μ, the systems become quickly ill conditioned, yielding a minimum reasonable value for μ.

- The minimum reasonable values for μ are significantly smaller for the interpolating approaches. Approximating approaches require more localized distance weight function to yield well conditioned system matrices. This effect is mainly due to the fact that matrices by the interpolating approaches have (in absolute value) larger diagonal values than the matrices by the approximating approaches (see Section 5.1.3).

- For medium to large values of μ the condition number and approximation error go to different directions. While the approximation error becomes smaller for μ increasing, the condition numbers increase (with exception of the AMLS approach, see below). The systems with μ finite consider more points and are more smoothed than the $\mu = \infty$ system. This results in better condition at the price of higher approximation error. Obviously, for small and regular problems as the one here, the approximation error dominates and should hence be minimized. One will select the standard three point stencil matrix $\mu = \infty$. However, for larger and more irregular systems, one generally solves the linear systems only to a given accuracy, and often times better conditioned systems result in smaller run times. Hence, one will select a finite μ for best performance.

- For sufficiently large values of μ, the matrices obtained by the local and the mixed approaches are M-matrices. The IMLS approach requires a slightly larger value of μ to yield M-matrices. The AMLS approach does (up to a tiny interval) not result in an M-matrix structure.

- The AMLS approach differs significantly from all other approaches. It yields significantly lower approximation errors, even lower than the standard three point stencil ($\mu = \infty$). On the other hand, the approach does not yield an M-matrix structure. The reason can be seen in Figure 5.11. It shows the central 4^{th} row of the system matrix with $n = 7$ intervals, scalarly multiplied with with the vector with components $(x_i - x_4)^4$, plotted in dependence of μ. One can observe that for the MLS stencil this function becomes zero exactly where the MLS error is minimal. The approximation

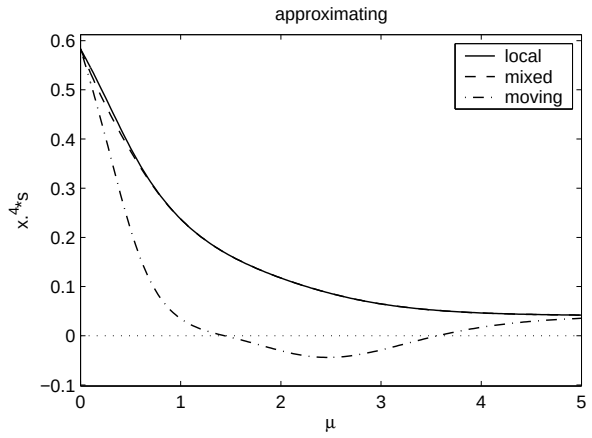

Figure 5.11: Central stencil acting on fourth order polynomial

to the Laplace operator is of higher order, yielding fourth order accuracy. The error values shown in Figure 5.8 and Figure 5.9 exactly fit to this observation. Also the fact that the matrix is no M-matrix can be explained: The above fourth order vector is non-negative. For the scalar product with the stencil entries to become zero, the stencil must have entries of both positive and negative sign. As shown in Section 4.2.2, a higher order approximation is incompatible with M-matrix structure. The M-matrix structure implies a good numerical stability. Note the bad condition numbers for moderate values of μ.

5.5.3 Numerical Effort of Stencil Computations

In this Chapter we have derived various stencil approximations to derivatives of a function on a point cloud. We have analytically calculated the stencils for various special cases in one space dimension. While this gives some insight in the general behavior of the various method, in real meshless applications in two or three space dimensions, the stencils have to be computed numerically. Since this has to be done for every interior (and Neumann boundary) point of the cloud, the effort of setting up the linear system which approximates the Poisson equation (4.1) must not be neglected. On the contrary, the numerical investigations in Section 8.2 indicate that if efficient solvers are used, setting up the systems can easily be more elaborate than a single solve.

The local approaches, as presented in Section 5.1, are the most straightforward approaches to implement. Equations (5.12) and (5.18) provide direct formulas on how to compute the stencils. We estimate the complexity in terms of the number of constraints k and the number of neighbors m. For any interior point the following operations are to be done.

1. **Set up matrix** $S = VWV^T$

Since the matrix W is diagonal, it is typically not efficient to use a black box matrix multiplication system. The fastest way is to first compute the products of polynomial basis function $b_\mu(\mathbf{x}_i)b_\nu(\mathbf{x}_i)$ for all considered neighboring points and then obtain the matrix entries $s_{\mu\nu} = \sum_{i=1}^m w_i b_\mu(\mathbf{x}_i)b_\nu(\mathbf{x}_i)$ directly. This requires m evaluations of the distance weight function. For inverse polynomial functions, the effort is small. Evaluating more complex functions (e.g. involving exponentials) might in the total balance require a significant amount of computation time. Assumed the evaluation time can be neglected, each matrix entry requires $2m$ floating point operations (flops) to be computed. Due to symmetry, one has a total of about mk^2 flops for the matrix setup. In 3d, with 40 neighboring points considered, this step takes more than 3200 flops.

2. **Solve linear system $S\mathbf{y} = \mathbf{b}$**
   The matrix S is small (10×10 in 3d) but full. It is symmetric positive definite. Hence, a Cholesky factorization [QSS00] can be used to solve the system, which requires about $\frac{k^3}{3}$ flops, i.e. this step is independent of the number of neighboring points. In 3d, it takes about 240 floating point operations to solve.

3. **Compute stencil $\mathfrak{s} = WV^T\mathbf{y}$**
   Most required values have been computed in the first step. Left in this step are mk flops, which in 3d, with 40 neighboring points, are 360 flops.

Altogether, computing a Laplace stencil with a LLS method requires

$$c = k(k+1)m + \frac{k^3}{3} \tag{5.284}$$

flops. For a reasonable amount of neighbors, the first term dominates. The effort is $6m$ in 1d, $30m$ in 2d, and $90m$ in 3d.

Moving approaches

The moving approaches as given by (5.99) and (5.273) require the evaluation of additional terms. Firstly, there are derivatives of the distance weight function. Assumed these are directly available (in which case they still would have to be evaluated, see above), one has to set up a matrix S' (and possibly S''), which is almost as expensive as computing S. Also, the third step has to be repeated involving derivatives of the weight function. Roughly speaking, Laplace stencil computations in the mixed approximating and the moving interpolating are twice as expensive as the local approach. The moving approximating approach requires more than three times the effort of the local approach.

The Poisson test problem in Section 5.5.2 indicates that the AMLS approach can yield accurate results. However, since these are due to a close to higher order stencil approximation,

one can also instead directly impose higher order in the systems, which is only slightly more expensive to compute (in 3d, the system would become 35×35) and yields correct fourth order accuracy.

Another possible approach, which we do not investigate further in this thesis, could be to evaluate the MLS function at regularly set up points close (but not too close) to the point of approximation and use a finite difference scheme using these values to approximate the derivative of the MLS function.

Approximating vs. Interpolating

To every approximating there is an interpolating approach, which is always one dimension smaller. For the local approach, the reduction of floating point operations will be about 10% in 3d, 15% in 2d, and 30% in 1d, for the operations depending on the number of neighboring points, and about 25% in 3d, 40% in 2d, and 70% in 1d, for solving the linear system.

5.5.4 Conclusions

In this chapter, we have investigated various meshless approaches for approximating derivatives on point clouds. The methods apply in any space dimension. We have considered specific examples in one space dimension, which admit an easy analysis. Since for problems in two or three space dimensions, the 1d problems can always be recovered as a subproblem (e.g. in a directional derivative), a failure or bad performance in 1d typically also disqualifies the method for multidimensional applications, too. Other observations, which have been presented in 1d for simplicity, transfer in the same or a similar manner to the multidimensional case.

The approaches considered in this chapter all select a stencil by formulating a weighted quadratic minimization problem:

- The **local least squares** (LLS) method, as presented in Section 5.1, locally approximates the function by a polynomial and differentiates the polynomial.
- The **moving least squares** (MLS) method, as presented in Section 5.2, sweeps with a polynomial approximation through the whole data space and thus constructs an approximating function. This function is then differentiated.
- At a point of approximation, a **constrained weighted least squares** problem can be formulated, as done in Section 5.1.3. This is shown to be equivalent to the LLS method.

- **Alternative derivations**. In [KT01, MT02, KT03] the ALLS stencils are derived by minimizing the weighted L^2 norm of the error in the Taylor expansion. In [FS01] the ILLS derivative approximation is derived by formulating a system of constraints for the first and second derivate at once. Numerous other interpretations exist.

The various derivations and interpretations which involve a weighted least squares formulation boil down to two methods: The LLS and the MLS method. We have investigated the difference in various examples. The general conclusion was that the moving approaches have a tendency to "overswing" in the stencil values. The local approaches yield a monotonic behavior in dependence on the width of the distance weight functions. For the Poisson test problem, the AMLS method showed the interesting behavior of being close to a higher order approximation, thus yielding much better approximation results. However, the additional computational effort, as outlined in Section 5.5.3, does not really justify the moving approaches. Since in this thesis, we are more interested in stability and positivity of stencils, we conclude

In this context, local approaches are preferable to moving approaches.

All approaches have in common that one can decide between an approximating and an interpolating approach. These can have very different interpretations depending on the derivation. Generally, results obtained by an approximating approach are more "flat" and more averaged, while results by interpolating approaches show larger gradients and differences. For instance, the approximating polynomials in the AMLS method pass in between different function values, while the IMLS polynomials swing up to actually interpolate the values. The stencils obtained by the interpolating approach have a central entry which is larger in absolute value compared to the AMLS approach (see Section 5.1.3). In our comparison interpolating versus approximating, the ILLS method wins 4:0

- The ALLS method can yield negative stencil entries, where the ILLS stencils are positive (see Section 5.5.1).
- The ALLS includes the stencil entry of the point of approximation into the minimization problem (see Section 5.1.3). However, generally the central entry should be large in absolute value compared to the other entries.
- Applied to a Poisson model problem (see Section 5.5.2), the ILLS approach yields slightly smaller errors and slightly better conditioned matrices than the ALLS approach. The locality parameter μ can be chosen much smaller in the interpolating case.
- The ILLS systems are by one dimension smaller than the ALLS systems, and thus are less expensive to compute (see Section 5.5.3).

Hence we conclude

> **In this context, interpolating approaches are preferable to approximating approaches**.

The ILLS is a good method for obtaining meshless finite difference approximations of the Poisson equation. Among all methods which perform a weighted quadratic least squares approximation, it shows the best performance. However, why restrict to methods which perform a quadratic least squares approximation? Any stencil which satisfies the constraints (4.12) yields a consistent derivative approximation. Why should the minimizer of a quadratic minimization problem be the preferable one? Due to the freedom in the distance weight function there is no unique answer anyhow. Why not demand other properties, such as positivity of the stencil? Why not linear minimization? We will investigate these questions in Chapter 6.

Chapter 6

The Minimal Positive Stencil Method

In Chapter 5 we have presented and analyzed various approaches based on the formulation of a weighted least squares problem. The ILLS approach has turned out to be a good and stable method to select a unique stencil in the class of methods which formulate a quadratic minimization problem. However, as we will show in this chapter, it does not guarantee positive stencils.

Also, the methods presented in the previous chapter used a circular neighborhood condition to select points which are incorporated into a local derivative approximation. Formally, all points are used, but due to compactly supported (or negligibly small in value) distance weight functions, only points inside a ball of radius r around the point of consideration were considered. In Section 3.5 we have presented alternative neighborhood criteria, generated by geometric criteria. These can be used in local generalized finite difference approaches. The presented neighborhood criteria all had problems in special point cloud configurations. For instance, the Delaunay neighborhood may yield too few neighbors, whereas the circular neighborhood may yield too many or not nicely distributed neighbors.

We present a new approach, which both deviates from the quadratic minimization dogma, as well as yields a nouvelle concept of defining neighboring points to a point in a cloud. A concept which is not defined by geometric criteria, but by a requirement of approximating the Laplace operator by positive stencils. We provide criteria on the point cloud geometry, such that the new approach yields positive stencils for any point. This is shown to lead to an M-matrix structure of the system matrix. The new approach requires methods from linear optimization to obtain the local approximation. We compare various methods for the linear programs which arise in this context. The treatment at and close to the boundaries requires special attention.

6.1 M-Matrices

The numerical solution of the Poisson equation, as presented in Chapter 4, typically consists of two steps: Firstly discretizing the partial differential equation to a finite dimensional linear system, and secondly computing (or at least approximating sufficiently well) the solution of the linear system. As reasoned before, there are various ways how to discretize the equation, and similarly, there is a wide arsenal of linear solvers available. Since efficient linear solvers typically can be guaranteed to converge only for specific classes of matrices, it is of interest to consider which properties of the system matrix can be achieved by the discretization approach.

In the context of finite element methods, and also for finite difference methods for regular grids, the arising linear system matrices are typically symmetric. This is a favorable situation with respect to many solvers, in particular since symmetric matrices arising as discretizations of partial differential equations are often times positive definite under moderate conditions. The latter is one of the most favorable classes with respect to linear solvers.

In a meshless finite difference context, however, the arising system matrices are in general non-symmetric, and due to the locality of the approximation it would be fairly difficult to enforce symmetry. Hence, we need to consider favorable subclasses of regular non-symmetric matrices. One desirable structural property is reciprocity, i.e. the matrix is symmetrically structured. Also of interest are weak diagonal dominance properties. Further considered need to be connectivity aspects of the matrices. Our main interest lies in the class of M-matrices.

Consider a square matrix $A \in \mathbb{R}^{n\times x}$. Let the matrix elements be denoted a_{ij}. We define

Definition 6.1. A square matrix A is called *Z-matrix*, if $a_{ij} \leq 0 \ \forall i \neq j$.

Definition 6.2. A square matrix A is called *L-matrix*, if it is a Z-matrix and additionally $a_{ii} > 0 \ \forall i$.

We furthermore consider inverse positive matrices, which is a componentwise statement. Hence we introduce the notation

Definition 6.3. Let $A \in \mathbb{R}^{m\times n}$ be a matrix. We write $A \geq 0$ if $a_{ij} \geq 0 \ \forall i, j$, and $A > 0$ if $a_{ij} > 0 \ \forall i, j$. Analogously we write $A \leq 0$ if $-A \geq 0$, and $A < 0$ if $-A > 0$. The same notation applies to vectors.

In this context, we do not deal with any notation for positive definiteness of matrices, thus inequality relations are always meant componentwise.

Definition 6.4. A regular square matrix A is called *inverse positive*, if $A^{-1} \geq 0$.

Definition 6.5. A square matrix A is called *M-matrix*, if it is a Z-matrix and inverse positive.

The class of M-matrices is a true subset of the class of inverse positive matrices.

Example 6.1. An example of an inverse positive matrix which is not an M-matrix is

$$A = \begin{pmatrix} 1 & -\frac{3}{2} & \frac{1}{4} \\ -1 & 2 & -1 \\ 0 & 0 & 1 \end{pmatrix} , \tag{6.1}$$

which is a consistent discretization matrix of a Poisson problem on the grid $X = (0, 1, 2)$ with one Neumann and one Dirichlet boundary point. Obviously, it is not an L-matrix. The inverse matrix is

$$A = \begin{pmatrix} 4 & 3 & 2 \\ 2 & 2 & \frac{3}{2} \\ 0 & 0 & 1 \end{pmatrix} . \tag{6.2}$$

In the following, we consider the class of M-matrices only, since these yield a sufficient condition for inverse positivity. If the latter is the only desired property, one can use other characterizations of inverse positive matrices, such as given by Fujimoto and Ranade [FR04], in order to find conditions on the discretization approach.

While the Z- and the L-matrix property are easy to verify, the M-matrix property is difficult to check by definition, since our interest lies in such large matrices, that its inverse is not directly available by reasonable effort. Required are easy to verify or enforce criteria, which are sufficient for inverse positivity. On the other hand it is of interest which other properties the M-matrix property gives rise to, i.e. in which sense the M-matrix property is favorable.

6.1.1 Sufficient Conditions for M-Matrix Structure

Sufficient conditions for an M-matrix structure can be obtained from diagonal dominance and connection properties. We define

Definition 6.6. A matrix A is called *strictly diagonally dominant*, if

$$\forall i : \quad |a_{ii}| > \sum_{j \neq i} |a_{ij}| , \tag{6.3}$$

and *weakly diagonally dominant*, if

$$\forall i : \quad |a_{ii}| \geq \sum_{j \neq i} |a_{ij}| . \tag{6.4}$$

Sometimes one calls a single row i strictly respectively weakly diagonally dominant, if the appropriate condition (6.3) respectively (6.4) is satisfied for this row. An important criterion for M-matrix structure is the following

Theorem 6.1. *A strictly diagonally dominant L-matrix is an M-matrix.*

Proof. The proof is given in [Hac93, p. 153]. □

In our context, all rows corresponding to interior points do not have the diagonal dominance property, since the sum of all entries is zero. If the stencil is positive, then the weak diagonal dominance property is satisfied. Unfortunately, weak diagonal dominance is not sufficient for L-matrices to be M-matrices, hence we need to employ information about the connectivity of the matrix. The connectivity of a matrix is given by the graph, as defined in Section 4.1.1. We consider each row and column to be associated with a value from an index set I.

Definition 6.7. Let $A \in \mathbb{R}^{I \times I}$ be a square matrix with respect to the index set I. It is called *essentially diagonally dominant*, if it is weakly diagonally dominant, and every point $i \in I$ is connected to a point $j \in I$ which satisfies the strict diagonal dominance relation.

In other words, every point whose row is only weakly diagonally dominant is connected to a point with a strictly diagonally dominant row.

Theorem 6.2. *An L-matrix arising as a finite difference discretization of problem (4.1) is essentially diagonally dominant, if it is essentially irreducible.*

Proof. The discretization of problem (4.1) has Dirichlet boundary points (since the matrix is essentially irreducible). Each row corresponding to such a point satisfies the strict diagonal dominance relation. Every interior and Neumann boundary point satisfies the weak diagonal dominance criterion, due the L-matrix structure and relation (4.9). The matrix is essentially irreducible, thus every such point is connected to a Dirichlet boundary point. Hence the matrix is essentially diagonally dominant. □

In our context, the central sufficient criterion for a M-matrix structure is the following

Theorem 6.3. *An essentially diagonally dominant L-matrix is an M-matrix.*

Proof. The proof is given in [Hac93, p. 153]. □

Essential irreducibility is actually the minimum requirement on a Poisson discretization matrix, as the following theorem shows.

Theorem 6.4. *A matrix arising as a finite difference discretization of problem (4.1) is singular, if it is not essentially irreducible.*

Proof. Let I_D denote the index set of the Dirichlet boundary points. Define the set $I_0 = \{i \in I : \exists j \in I_D, \text{ s.t. } i \text{ is connected to } j\}$. The complementary set $I_1 = I \setminus I_0$ is nonvoid by assumption. Consider the index set be ordered $I = (I_0, I_1)$. With respect to this ordering, the system matrix has the representation

$$A = \left(\begin{array}{c|c} A_{I_0 I_0} & A_{I_0 I_1} \\ \hline 0 & A_{I_1 I_1} \end{array}\right) \tag{6.5}$$

The set I_0 contains no Dirichlet boundary point, thus every row of $\det(A_{I_0 I_0})$ sums up to zero. Consequently $\det(A) = \det(A_{I_0 I_0}) \det(A_{I_1 I_1}) = 0$. □

This chapter is devoted to the two tasks:

- **How can the finite difference matrix be guaranteed to have an L-matrix structure?**
  We prove conditions on the point cloud geometry, such that positive stencils can be guaranteed for interior points (Section 6.4.3) and for Neumann boundary points (Section 6.5.1). The approach presented in Section 6.3 yields these positive stencils computationally.
- **How can the finite difference matrix be guaranteed to be essentially irreducible?**
  In Section 6.6 we investigate how the given approximations can be forced to yield appropriately connected system matrices.

If both properties are satisfied, the system matrix is guaranteed to be an M-matrix, and one can enjoy all the nice properties described in the following.

6.1.2 Consequences of M-Matrix Structure

The class of (not necessarily symmetric) M-matrices is particularly nice both theoretically and with respect to linear solvers. The most important properties of M-matrices are the discrete maximum principle and the guaranteed convergence of various linear solvers.

Theorem 6.5. *Let A be an M-matrix. Then $A\mathbf{x} \leq 0$ implies $\mathbf{x} \leq 0$.*

Proof. A is an M-matrix, thus $A^{-1} \geq 0$ by definition. Let $\mathbf{y} = A\mathbf{x}$. Then $\mathbf{x} = A^{-1}\mathbf{y}$. The componentwise inequalities $A^{-1} \geq 0$ and $\mathbf{y} \leq 0$ imply $\mathbf{x} \leq 0$. □

Remark 6.1. The above statement can be reversed, as presented in [QSS00, p. 29]: A Z-matrix satisfying $A\mathbf{x} \leq 0 \Rightarrow \mathbf{x} \leq 0$ is an M-matrix.

Theorem 6.5 shows that M-matrices mimic the maximum principle of the Poisson equation (4.1) in a discrete maximum principle.

Various maximum principles for elliptic problems are given in the book of Evans [Eva98]. In order to see the transfer to the discrete case, we state the following version. Consider the Poisson equation

$$\begin{cases} -\Delta u = f & \text{in } \Omega \\ u = g & \text{on } \Gamma \end{cases} \tag{6.6}$$

Let $f \leq 0$ and $g \leq 0$. Then the solution satisfies $u \leq 0$.

Let now $A\hat{\mathbf{u}} = \hat{\mathbf{f}}$ be a discretization of problem (6.6). The right hand side vector contains values of the functions f and g, thus $\hat{\mathbf{f}} \leq 0$. Assume now the discretization matrix A is a Z-matrix. Then it is an M-matrix if and only if $\hat{\mathbf{u}} \leq 0$. In other words, if the discretization matrix is an M-matrix, then the discrete solution satisfies the maximum principle. Conversely, if the system matrix is a Z-matrix, but does not have an M-matrix structure, then the maximum principle is not satisfied.

Furthermore, the M-matrix property is sufficient for the convergence of various iterative linear solvers. We will present the important theorems in Section 7.1.1. More information about M-matrices in relation to iterative linear solvers can be found in the books of Axelsson [Axe94] and Varga [Var00].

6.2 Approaches to Obtain Positive Stencils

Positivity is a desired property of Laplace stencils. It is well known that many approaches do not always yield positive stencils (see Example 6.2). Various authors have approached the question under which conditions classical least squares approaches yield positive stencils. Demkowicz, Karafiat and Liska [DKL84] analyze the positivity of stencils for specific configurations in two space dimensions. Duarte, Liszka and Tworzyako [DLT96] consider an "optimal star selection" for positive stencils. The arguments, however, are heuristic. Fürst and Sonar [FS01] derive topological conditions on point clouds for positive least squares stencils in a one dimensional setup, which is of course not the main interest of meshless methods. Similarly, Iliev and Tiwari [IT02] analyze specific 1d cases and provide conditions for the positivity of stencils when points are added and removed. Similar problems with positive Laplace stencils arise in the immersed interface method, as described by Li [Li03]. The method deals with a regular grid, however, in the presence of interfaces one has to incorporate specific jump conditions.

To our knowledge, the question under which conditions least squares approaches yield positive stencils, has not been answered yet in a general meshless d dimensional setup. In this chapter, we pose and answer a slightly different question, namely under which conditions on the point geometry does a positive stencil exist, and how can it be obtained?

Note that there are various approaches on how least squares methods can be augmented to make them more likely to yield positive stencils. One idea is the following: The central stencil entry should be negative. Since it is minus the sum of all other entries, one can force the central entry to be "more negative", i.e. prescribe a significant negative value and hope that all other entries increase, possibly shifting values from negative to positive. The central value can prescribed as an additional constraints in system (5.13) for the ALLS method. It has to be pointed out that this approach leads to a contradiction in the case of all neighboring points have the same distance h to the central points. In this case the central point must have the value $-\frac{2d}{h^2}$, there is no freedom to prescribe it. To our knowledge, there is no augmentation to a least squares approach, which guarantees positive stencils in a general meshless d dimensional setup.

6.3 Linear Minimization Approach

In Section 4.2 we have derived that a finite difference approximation is consistent if the stencil vector satisfies $V \cdot \mathfrak{s} = \Delta\mathbf{b}$, where V is the Vandermonde matrix depending on the local point cloud geometry. As soon as more neighboring points than constraints are present (which typically is the case), there are infinitely many solutions. Least squares methods, as presented in Chapter 5, select a unique stencil by formulating a quadratic minimization (QM) problem. For instance, the ILLS method, as presented in Section 5.1.2 can be formulated as a minimization problem

$$\min \sum_{i=1}^{n} \frac{\mathfrak{s}_i^2}{w_i}, \quad \text{s.t. } V \cdot \mathfrak{s} = \Delta\mathbf{b} \ , \tag{6.7}$$

as derived in Section 5.1.3. Let in the following $\mathfrak{s} = (\mathfrak{s}_1, \ldots, \mathfrak{s}_m)^T$, i.e. the stencil entry corresponding to the central point is not included in the vector $\mathfrak{s}$. Consequently, the matrix V does not contain the constant constraint (4.9).

In this chapter, we would like to guarantee the system matrices to be M-matrices, which requires the stencils to be positive. However, least squares approaches in general fail to always yield positive stencils, as the following example indicates.

Example 6.2. Consider a central point $\mathbf{x}_0 = (0, 0)$ and 6 neighboring points on the unit circle $\mathbf{x}_i = (\cos(\frac{\pi}{2}t_i), \sin(\frac{\pi}{2}t_i))$, where $(t_1, \ldots, t_6) = (0, 1, 2, 3, 0.1, 0.2)$ (see Figure 6.1). As the neighboring points all lie on the unit circle, the distance-weight function does not play a role in any local approximation. In moving approximations, derivatives of the distance

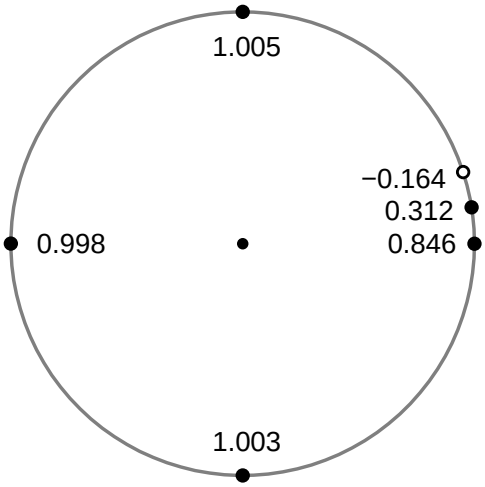

Figure 6.1: Quadratic minimization can yield non-positive stencils

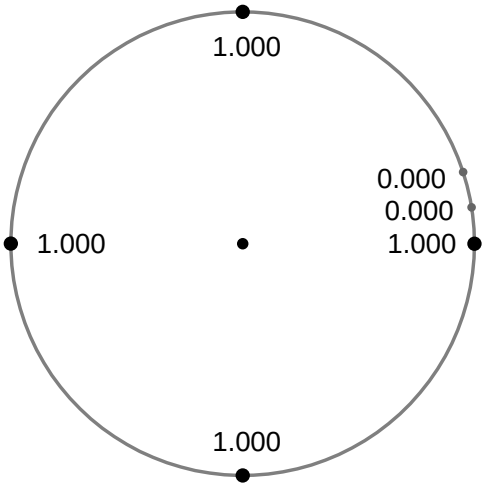

Figure 6.2: Positive stencil by linear minimization

weight function at distance 1 can enter the result. However, the additional terms arising in a moving approach are unlikely to yield a positive stencil, due to the following fact. One quadratic constraint for the stencil to satisfy is

$$\sum_{i=1}^{6} x_i y_i \mathfrak{s}_i = 0 \tag{6.8}$$

For the first four neighboring points we have $x_i y_i = 0$, leaving us with the constraint $\sin(0.1\pi)\mathfrak{s}_5 + \sin(0.2\pi)\mathfrak{s}_6 = 0$. Both coefficients are positive, hence a positive stencil must satisfy $\mathfrak{s}_5 = \mathfrak{s}_6 = 0$. The condition defines the positive stencil uniquely, namely $\mathfrak{s} = (1, 1, 1, 1, 0, 0)$, as shown in Figure 6.2. This stencil, however, is not minimal in a weighted least squares sense. Figure 6.1 shows the stencil resulting from the ILLS method, $\mathfrak{s} = (0.846, 1.005, 0.998, 1.003, 0.312, -0.164)$, which would destroy an M-matrix structure. By slightly moving the points on the circle, one can extend this example to a case, for which the positivity constraint does not determine the stencil uniquely. Since for least squares approaches the obtained stencils depend continuously on the geometry, one will still obtain non-positive solutions.

In general QM approaches do not guarantee positive stencils. Thus we enforce them, i.e. we search for solutions contained in the polyhedron

$$P = \{\mathfrak{s} \in \mathbb{R}^m : V \cdot \mathfrak{s} = \Delta\mathbf{b},\ \mathfrak{s} \geq 0\}\ . \tag{6.9}$$

Since we impose sign constraints, we obtain the feasibility problem from the field of linear optimization [Chv83]. It is not clear a priori whether solutions exist, even if the matrix V has full rank. In Section 6.4 we show under which conditions solutions do exist.

If P is nonvoid and not degenerate, there are infinitely many feasible stencils. Again, one has to single out a unique one. This can again be done by formulating a minimization problem. Now we formulate a linear minimization (LM) problem

$$\min \sum_{i=1}^{m} \frac{\mathfrak{s}_i}{w_i}, \text{ s.t. } V \cdot \mathfrak{s} = \Delta\mathbf{b}, \ \mathfrak{s} \geq 0 \ . \tag{6.10}$$

As for the least squares approaches, we assume the weights to depend on the distance to the central points only, i.e. $w_i = w(\|\mathbf{x}_i - \mathbf{x}_0\|)$, where w is an appropriately decaying distance weight function (see Section 6.3.1 on the role of the distance weight functions).

Note first of all that we indeed *can* afford to formulate a linear minimization problem, since we have imposed sign constraints. Without sign constraints, a LM formulation would be unbounded and thus have no solution. With the sign constraints, the problem is bounded, since the weights w_i are all non-negative.

The following facts support our approach:

- **M-matrix property**
  If a positive stencil $\mathfrak{s} \geq 0$ can be obtained for every point, the resulting system matrix is an L-matrix. As outlined in Section 6.1.1, this is one of the requirements for obtaining an M-matrix. The desirability of positive stencils has also been pointed out by Liszka et. al. in [DKL84, DLT96].

- **Error in Taylor expansion**
  The finite difference approximation is based on a Taylor expansion. As derived in Section 4.2, the error in the approximation, given by formula (4.7), is

  $$e = \sum_{i=1}^{m} \mathfrak{s}_i e_i \tag{6.11}$$

  where e_i is the local error of the Taylor expansion around $\mathbf{x}_0$, evaluated at the point $\mathbf{x}_i$. The total error e becomes particularly large if the local errors e_i accumulate, i.e. have the same sign. Due to the Taylor expansion, we expect $e_i = c\|\mathbf{x}_i\|^3$. The approximation is the better, the smaller the error e is. Minimizing the error yields the above LM formulation (6.10) with distance weight function $w(d) = |d|^{-3}$. If the point geometry is nearly symmetric, local errors e_i arising from points on opposing sides of the central point will cancel, in which case the distance weight function $w(d) = |d|^{-4}$ would be more appropriate to minimize the error e. In any case, least squares approaches will in general yield larger error terms in the Taylor expansion (however, not significantly larger).

- **Fundamental theorem of linear programming**
  Problem (6.10) is a linear program in standard form. Assumed the constraints admit a

solution (see Section 6.4), then due to the *fundamental theorem of linear programming* [Chv83] there is a basic solution, in which only k of the m stencil entries $\mathfrak{s}_i$ are different from zero, where k is the number of constraints. This yields significantly fewer matrix entries as in QM. Hence, applying the sparse system matrix will be significantly faster.

In practice, the LM approach will be implemented in the following manner. For every interior point take a set of neighbors as candidates. In principle any geometric criterion presented in Section 3.5 would be possible, but here one should choose a circular neighborhood, since for this we can show that a positive stencil always exists (see Section 6.4). The LM problem can be solved by any method from linear programming. We compare various methods in Section 6.7. Due to the fundamental theorem of linear programming, a stencil with only k nonzero entries is obtained. Since every stencil entry corresponds to one point, the LM approach selects a minimal stencil out of the candidate points. Due to the construction, this stencil is positive. Hence, we refer to this approach as the *minimal positive stencil method* (MPS).

Due to this selection of a stencil, the LM approach gives rise to a new concept of neighborhood. To any interior point, the points selected by LM are defined as its neighbors. In Section 6.9 we relate this new concept to the geometric neighborhood concepts presented in Section 3.5.1.

6.3.1 Comparison of Quadratic and Linear Minimization

In its formulation, the LM approach looks similar to the QM approach. However, the two approaches are fundamentally different. The major differences are:

- **Number of stencil entries**
  The QM approach considers neighboring points (defined by a geometric concept), and computes a stencil. Unless the points form a special configuration, all neighbors will have a nonzero stencil entry. The MPS method, on the other hand, considers a set of neighbors, and chooses out of them a minimal number of points. In general optimization routines yield the stencil values as an output. However, one can also think of the MPS method as a selection approach. Once k stencil entries are selected, the specific stencil entries are uniquely defined by the linear system $V \cdot \mathfrak{s} = \Delta\mathbf{b}$, where V consists only of the points selected by the MPS method.

- **Continuous dependence on geometry**
  The QM stencil entries depend continuously on the point positions. If the points are slightly moved, the obtained stencil entries also change only slightly, assumed the weight function is compactly supported and the radius of the neighborhood equals

the compact support. The LM stencil entries do not have this property. The set P of feasible solutions is a hyper polyhedron in $\mathbb{R}^m$. A basic solution of problem (6.10) is a vertex of P. The set P depends on the point positions, i.e. there is a relation between the problem space $\mathbb{R}^d$ and the optimization space $\mathbb{R}^m$. If the points are slightly moved, the solution depends continuously on the positions, as long as the same set of points is selected (i.e. the same basis is chosen). However, at some stage of moving the points, another vertex of P might minimize the linear function $\mathbf{c}^T\mathfrak{s}$, where $c_i = \frac{1}{w_i}$. The solution would "jump" from one vertex to another. A different set of k points is selected. A stencil entry which has been in the basis now drops to zero. Another stencil entry which now is in the basis jumps from zero to a positive value. Least squares methods are stencil computation approaches, the MPS method is a *stencil selection* approach. In Section 6.3.4 we briefly investigate to which extent the non-continuous dependence may cause problems.

- **Continuous dependence on locality parameter**
  Similarly, the QM stencil entries depend continuously on the locality parameter of the distance weight function. The LM stencil entries might jump in a similar manner as for changing the point positions.

- **Dependence on neighborhood radius**
  Enlarging the set of candidate neighbors (e.g. by increasing the circular radius) changes every stencil entry in QM. In the MPS method, considering more and more candidate points will at some point not yield a better selection, since with a reasonably decaying distance weight function far away points are unlikely to decrease the linear function $\mathbf{c}^T\mathfrak{s}$. Note, however, that this effect *can* happen in extreme geometries, since the MPS method approach forms a comparably elegant compromise between choosing points close by and choosing points distributed nicely around the central point (see Section 6.3.2).

- **Role of distance weight function**
  The distance weight function in the MPS method plays a different role as in least squares approaches. Consider for instance the class of distance weight functions $w(d) = |d|^{-\alpha}$. As shown in Section 5.4.2 for the ILS approaches, the stencil entries depend continuously on the locality parameter α (see Figure 5.4). For large values of α closer points are preferred, for small values of α points further away receive larger values. In least squares approaches, one generally chooses α large enough to give closer points more influence, but not too large, in order to have not only considered the closest points (which will in general not be well distributed).

  In the MPS method the distance weight function also prefers far away points for small and close by points for large values of α. However, the behavior for changing

α is different. Also, there is a clear separation of behavior at $\alpha = 2$, as shown in Section 6.3.2.

- **Interpretation as interpolation approach**
  Least squares approaches are typically derived from interpolation approaches, as presented in Section 5.2. The MPS method, on the other hand, is derived directly to approximate derivatives. It forms a neighbor selection approach, and its nice geometric properties are directly related to the approximation of the Laplace operator. In this sense, there is no straightforward interpretation of the approach as a method for function reconstruction.

Remark 6.2. One could ask why not remain with a QM problem (choose for instance any of the least squares methods), and additionally impose sign constraints. If the set of feasible solutions P is nonvoid, the problem does have a solution. Also, effective optimization routines exist to solve the arising Karush-Kuhn-Tucker system [Van01]. However, such a formulation would have the worst of both worlds. There would be no continuous dependence on the point cloud geometry, since stencil entries may be forced to be positive. On the other hand, if a solution of an unconstrained QM problem is positive anyhow, it is also the solution of the QM problem with sign constraints. Hence, the number of nonzero stencil entries is not limited as for the MPS method. In addition, the error in the Taylor expansion does not second quadratic minimization. We conclude: If sign constraints are imposed, linear minimization is the preferable minimization formulation.

6.3.2 Geometric Interpretation of Minimal Positive Stencils

The MPS method selects a minimal positive stencil out of the candidate points. Which points are chosen depends on the locality parameter α in the distance weight function. Summing over the diagonal in the quadratic constraints (4.11) yields the following relation, which is a property of every consistent Laplace stencil.

$$\sum_{i=1}^{m} \|\mathbf{x}_i\|_2^2 \mathfrak{s}_i = 2d \ , \tag{6.12}$$

i.e. the sum over the stencil entries, weighted with the squares of the point distances is a constant. When considering distance weight functions $w(d) = |d|^{-\alpha}$, this relation yields a clear separation in LM approaches with respect to the parameter α:

- If $\alpha < 2$, then points further away are preferred.
- If $\alpha = 2$, then the LP (6.10) is degenerate.

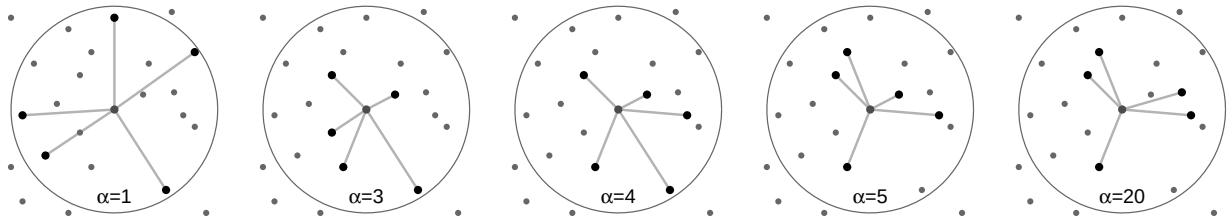

Figure 6.3: Minimal positive stencil for various values of α

- If $\alpha > 2$, then points close to the central point are preferred. Obviously, this case is of our interest.

The choice of stencil points for different values of $\alpha > 2$ is more complicated to analyze. Figure 6.3 shows an example point neighborhood and which positive stencil is selected for different values of α. As predicted, for $\alpha = 1$ a set of far away points is selected. The case $\alpha = 2$ does not yield a unique stencil, since there is no feasible function to minimize. In this specific example for $\alpha = 3, 4, 5, 20$ different stencils are selected (in practice, such a strong dependence on α does typically not take place). Observe that large values of α force the maximum distance to be as small as possible, but do not necessarily contribute in the closest points being selected. Still one can see a pattern, that smaller values of α tend more towards minimizing the largest angle, while larger values tend more towards minimizing the largest distance.

Important to notice is that for any choice of α, the selected points are distributed nicely around the central point. In particular, there are no angles larger than π, a necessary criterion which we prove in Theorem 6.7. The MPS method prefers points close by. However it does in general not select the closest positive stencil. It rather takes a compromise between the selected points being close by and nicely distributed.

Remark 6.3. This and the following examples will be two dimensional solely due to simplicity of presentation. The MPS applies directly to the three dimensional case. Even more, it only shows its real strength in three space dimensions. In Section 8.4 we apply the method to three dimensional problems and investigate the performance.

6.3.3 Minimal Positive Stencils for Specific Geometries

An interesting aspect about meshless methods is which results are obtained when they are applied to regular grid situations. We consider two examples, a regular hexagonal grid and a orthogonal grid with an additional point on an axis.

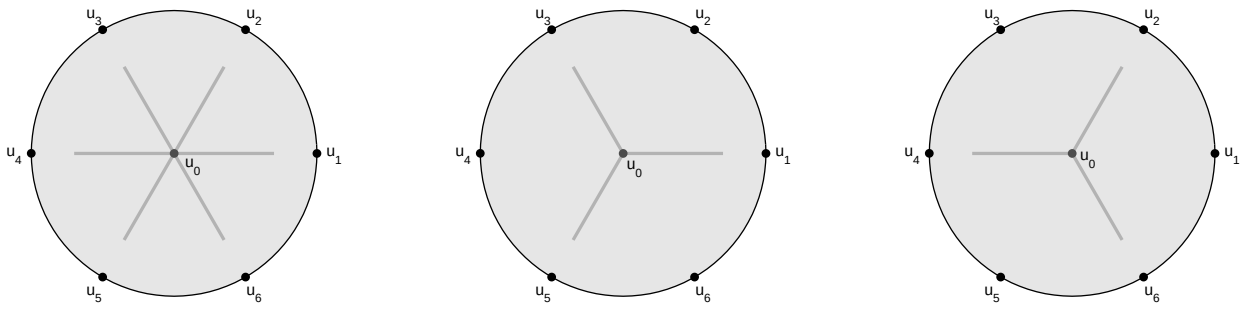

Figure 6.4: Hexagonal grid configuration with least squares and two possible MPS stencils

Regular Hexagonal Grid

For the example of a regular two dimensional hexagonal grid consider the central point $\mathbf{x}_0 = (0,0)$ and the neighboring points $\mathbf{x}_i = h\left(\cos(\frac{\pi}{3}i), \sin(\frac{\pi}{3}i)\right)$. The resulting Vandermonde system has the general solution

$$\mathfrak{s} = \mathfrak{s}_0 + \beta \mathfrak{s}_N \ , \quad \beta \in \mathbb{R} \ , \tag{6.13}$$

where $\mathfrak{s}_0 = \frac{4}{3h^2}(1,0,1,0,1,0)^T$ and $\mathfrak{s}_N = \frac{4}{3h^2}(-1,1,-1,1,-1,1)^T$. Since all neighboring points have the same distance, the distance weight function does not influence the result. Due to relation (6.12) the central stencil entry is $\mathfrak{s}_0 = -\frac{4}{h^2}$. Consequently, least squares methods yield the stencil $\mathfrak{s}_{LSQ} = \frac{2}{3h^2}(1,1,1,1,1,1)^T$. This result can also be concluded from the fact that least squares methods are independent of the direction, hence all stencil entries must be identical. The resulting stencils are sketched in Figure 6.4.

The linear minimization formulation (6.10) for this problem is degenerate in the sense that the minimization function is independent of the parameter β. Due to the sign constraints, any $\mathfrak{s}$ with $0 \leq \beta \leq 1$ solves the minimization problem. The MPS method additionally requires a solution to be a basis solution, which leaves us with the two solutions $\mathfrak{s}_{MPS} = \frac{4}{3h^2}(1,0,1,0,1,0)^T$ or $\mathfrak{s}_{MPS} = \frac{4}{3h^2}(0,1,0,1,0,1)^T$. Which of the two solutions is selected, depends on the particular implementation of the solver of the linear program. In any case, a basis solution will be degenerate. Two unknowns will formally appear in the basis, but with value zero.

On the polyhedron of feasible solutions, the given hexagonal setup has stencil solutions along an edge. The linear minimization function is parallel to this edge. Least squares methods choose the point in the middle of this edge ($\beta = \frac{1}{2}$). The MPS selects one of the two vertices ($\beta = 0$ or $\beta = 1$). This setup is also an example of a situation in which the MPS stencil can "jump", as described in Section 6.3.1.

Regular Orthogonal Grid with Additional Point

We consider the situation described in Example 4.4 and shown in Figure 4.3. The stencil solutions to the problem are

$$\mathfrak{s} = (1 - 3\mathfrak{s}_5, 1 - \mathfrak{s}_5, 1, 1, \mathfrak{s}_5)^T \ . \tag{6.14}$$

The ILLS method yields the solution satisfying $\mathfrak{s}_5 = \frac{4}{10+\lambda}$, where $\lambda = \frac{w(1)}{w(2)}$. Only in the limit of a very localized distance weight function ($\lambda \to \infty$) the stencil approaches the regular five point stencil $\mathfrak{s} = (-4, 1, 1, 1, 1, 0)^T$. On the other hand, the MPS approach yields exactly this regular stencil as a unique solution (assumed $\alpha > 2$, as derived in Section 6.3.2).

6.3.4 Discontinuous Dependence on Geometry

The MPS stencils do not depend continuously on the point geometry. As outlined before, if the points move, the stencil will at some point jump from one to another selection of points. The question is whether the numerical solution of the Poisson problem also shows this discontinuous behavior with respect to the point positions. As an example consider the unit circle as computational domain and a hexagonal setup of 6 boundary points. Additionally, one interior point in the origin in considered. This is exactly the setup described in Section 6.3.3, which has two possible MPS stencils. The geometry is shown in Figure 6.4.

Consider the Poisson problem (4.1). We have the right hand side $f_0 = \Delta u(\mathbf{x}_0)$ and the Dirichlet data $g_i = u(\mathbf{x}_i)$. The numerical value obtained for the interior point is

$$u_0 = \frac{1}{\mathfrak{s}_0} \left(f_0 - \sum_{i=1}^{6} \mathfrak{s}_i g_i \right) \tag{6.15}$$

Let us now consider the central point to be slightly moved. For simplicity consider $f = 0$ and $g \neq 0$. The following example indicates that the MPS solution of a Poisson problem does in general not depend continuously on the point positions.

Example 6.3. Let the domain boundary be parameterized by the angle φ. Let the function $g(\varphi) = \cos(\varphi)^3$ be prescribed, and $f = 0$. Due to the mean value principle for harmonic functions [Eva98] the analytic solution at the origin is $u(0) = 0$. The two MPS stencils, however, yield the two possible numerical approximations $u_0 = \frac{3}{4}$ and $u_0 = -\frac{3}{4}$.

If the central point is slightly moved, one of the two stencils is uniquely selected and one of the two solutions is (approximately) adopted. Thus the numerical solution obtained by the MPS method does in general not depend continuously on the point geometry. Depending on the context this might be a drawback. Note that this fact does not contradict the convergence

of the MPS solutions for the number of points increasing. We have merely observed that a discretization with a single interior point can yield an inaccurate approximation to the correct solution.

Remark 6.4. In the considered setup, the least squares method would yield the correct solution, or at least be closer than the MPS solutions. This is due to the underlying symmetry of the geometry. The setup is favorable for the least squares method, while the situation is degenerate for the MPS method.

In the case $f \neq 0$, the same type of discontinuity can happen. Note that in the specific setup of all neighbors on a circle the central stencil entry $\mathfrak{s}_0$ does actually not jump. It will, however, for different geometries.

6.4 Conditions for the Existence of Positive Stencils

In this section we investigate under which conditions the polyhedron P is nonvoid, i.e. when do positive solutions exist. W.l.o.g. we consider the point of approximation to be in the origin $\mathbf{x}_0 = 0$. We assume a set of neighboring points $(\mathbf{x}_1, \ldots, \mathbf{x}_m) \subset \mathbb{R}^d$ to be selected, for instance by the circular neighborhood relation. We assume that at least as many neighbors are present as constraints are to be satisfied, i.e. $m \geq k$.

We consider an interior point at which the Laplace operator should be approximated. Let the Vandermonde matrix V and the vector $\Delta\mathbf{b}$ be defined as before. The polyhedron being nonvoid means that the underdetermined system $V \cdot \mathfrak{s} = \Delta\mathbf{b}$ has a solution $\mathfrak{s} \geq 0$. The point geometry lives in $\mathbb{R}^d$, where $d \in \{2, 3\}$. The polyhedron exists in the space $\mathbb{R}^m$, where typically $m \approx 40$. The latter space is difficult to relate to the point geometry space. Hence, we consider the dual problem, which exists in $\mathbb{R}^k$, where $k \in \{5, 9\}$. We use

Theorem 6.6 (Farkas' Lemma)**.** *If A is a real matrix and $\mathbf{b}$ a real vector, then exactly one of the following two systems has a solution:*

- $A \cdot \mathbf{x} = \mathbf{b}$ *for some* $\mathbf{x} \geq 0$*, or*
- $A^T \cdot \mathbf{w} \geq 0$ *for some* $\mathbf{w}$ *satisfying* $\mathbf{b}^T \cdot \mathbf{w} < 0$*.*

Proof. The proof is given in the book of Chvátal [Chv83]. □

Applying *Farkas' Lemma* to our problem yields that system $V \cdot \mathfrak{s} = \Delta\mathbf{b}$ has no solution $\mathfrak{s} \geq 0$, if and only if system $V^T \cdot \mathbf{w} \geq 0$ has a solution satisfying $\Delta\mathbf{b}^T \cdot \mathbf{w} < 0$. The i^{th} component of $V^T \cdot \mathbf{w}$ can be written as

$$\left(V^T \cdot \mathbf{w}\right)_i = \mathbf{a}^T \cdot \mathbf{x}_i + \mathbf{x}_i^T \cdot A \cdot \mathbf{x}_i, \tag{6.16}$$

where $\mathbf{a} = (w_1, \ldots, w_d)^T$ and A is the symmetric matrix

$$A = \begin{pmatrix} w_4 & w_3 \\ w_3 & w_5 \end{pmatrix} \text{ (2d)} \quad \text{resp.} \quad A = \begin{pmatrix} w_7 & w_4 & w_5 \\ w_4 & w_8 & w_6 \\ w_5 & w_6 & w_9 \end{pmatrix} \text{ (3d)} \tag{6.17}$$

Given $\mathbf{w}$ (respectively $\mathbf{a}$ and A), we consider the quadratic form

$$f(\mathbf{x}) = \mathbf{a}^T \cdot \mathbf{x} + \mathbf{x}^T \cdot A \cdot \mathbf{x} \tag{6.18}$$

Since A is symmetric, an orthogonal matrix $S \in O(d)$ exists, such that $S^T A S = D$, where $D = \operatorname{diag}(\lambda_1, \ldots, \lambda_d)$. In the new coordinates we define

$$g(\mathbf{x}) = f(S\mathbf{x}) = \mathbf{d}^T \cdot \mathbf{x} + \mathbf{x}^T \cdot D \cdot \mathbf{x} \tag{6.19}$$

where $\mathbf{d} = S^T \mathbf{a}$. If all eigenvalues $\lambda_i \neq 0$, the matrix D is regular, and we can write

$$g(\mathbf{x}) = (\mathbf{x} - \mathbf{c})^T \cdot D \cdot (\mathbf{x} - \mathbf{c}) - \mathbf{c}^T \cdot D \cdot \mathbf{c} \tag{6.20}$$

where $\mathbf{c} = -\frac{1}{2} D^{-1} \mathbf{d}$. If, on the other hand, one or two $\lambda_i = 0$, we are in a degenerate case, and stick to the representation with $\mathbf{d}$ as parameter. In the following derivation we consider $\mathbf{c}$ as a parameter. However, the statements do hold for the degenerate case, too. Only the geometry is described with $\mathbf{d}$ instead of $\mathbf{c}$.

Choosing $\mathbf{w} \in \mathbb{R}^k$ arbitrarily is equivalent to choosing $S \in O(d)$, $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_d) \in \mathbb{R}^d$ and $\mathbf{c} \in \mathbb{R}^d$ (respectively $\mathbf{d} \in \mathbb{R}^d$) arbitrarily. For any $\boldsymbol{\lambda}, \mathbf{c} \in \mathbb{R}^d$ define the domain

$$H_{\boldsymbol{\lambda},\mathbf{c}} = \{\mathbf{x} \in \mathbb{R}^d : g(\mathbf{x}) \geq 0\} \tag{6.21}$$

For a set of points $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ define $SX = \{S\mathbf{x}_1, \ldots, S\mathbf{x}_m\}$. Due to Farkas' Lemma, system $V \cdot \mathfrak{s} = \Delta\mathbf{b}$ has no solution $\mathfrak{s} \geq 0$, if and only if $S \in O(d)$, $\mathbf{c}, \boldsymbol{\lambda} \in \mathbb{R}^d$ with $\sum_{i=1}^d \lambda_i < 0$ exist, such that

$$SX \subset H_{\boldsymbol{\lambda},\mathbf{c}}. \tag{6.22}$$

Criterion (6.22) means that the set of points X can be transformed (via $S \in O(d)$), such that it is contained in the set $H_{\boldsymbol{\lambda},\mathbf{c}}$ for some $\mathbf{c}, \boldsymbol{\lambda} \in \mathbb{R}^d$ with $\sum_{i=1}^d \lambda_i < 0$. It is exactly equivalent to the non-existence of a positive Laplace stencil.

Example 6.4. The setup in Figure 6.5 shows that for a special domain $H_{\boldsymbol{\lambda},\mathbf{c}}$ a cloud of neighboring points can be rotated, such that it is completely contained in the domain. Due to Farkas' Lemma one knows that for the given set points no positive stencil exists, i.e. any solution of $V \cdot \mathfrak{s} = \Delta\mathbf{b}$ has at least one negative component.

Example 6.5. The setup in Figure 6.6 shows a point setup which obviously has a positive stencil solution. Due to Farkas' Lemma it is impossible to find a domain $H_{\boldsymbol{\lambda},\mathbf{c}}$ and to rotate the set of points, such that all points are contained the domain.

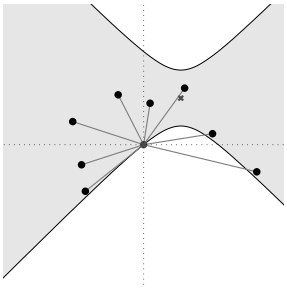

Figure 6.5: No positive stencil

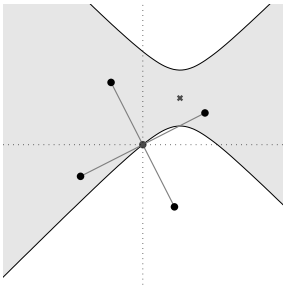

Figure 6.6: Positive stencil

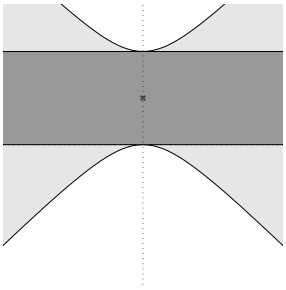

Figure 6.7: Necessary criterion

Remark 6.5. It is a curious aspect that Farkas' Lemma flips the method of checking positivity. In order to show that a set of points has no positive solution it suffices to find a single example of $\boldsymbol{\lambda}$, $\mathbf{c}$ and S which places the whole set in the domain. On the other hand, showing the existence of a positive stencil requires to show for all possible combinations of parameters that the set of points can not be contained in the domain.

We have derived a geometric condition, which is equivalent to the existence of positive stencils. However, due to the nonlinearity in g, it is difficult to translate condition (6.22) into geometric means. Instead, we derive a necessary (but not sufficient) as well as a sufficient (but not necessary) criterion on the point geometry for the existence of a positive Laplace stencil. To our knowledge the latter has not been given yet.

6.4.1 A Necessary Criterion for Positive Stencils

If $V \cdot \mathfrak{s} = \Delta \mathbf{b}$ has a solution $\mathfrak{s} \geq 0$, then for any $S \in O(d)$, $\mathbf{c}, \boldsymbol{\lambda} \in \mathbb{R}^d$ with $\sum_{i=1}^{d} \lambda_i < 0$, there is a point $\mathbf{x}_i$ with $S\mathbf{x}_i \notin H_{\boldsymbol{\lambda},\mathbf{c}}$. For the particular choice $\lambda_1 = -1$, $\lambda_i = 0 \ \forall i > 1$ and $c_1 \gg \max_i \|\mathbf{x}_i\|$ it follows that for any $S \in O(d)$ at least one point must satisfy $x_1 < 0$. This yields the following

Theorem 6.7. *If a set of points $X \subset \mathbb{R}^d$ around the origin admits a positive Laplace stencil, then the points must not lie in one and the same half space (with respect to an arbitrary hyperplane through the origin).*

This result is well known and has for instance been presented by Duarte, Liszka and Tworzyako [DLT96]. Obviously, due to the particular choice of $\boldsymbol{\lambda}$ above, this criterion is very crude, but very easy to formulate in geometric means. A more careful estimate of (6.22) may yield stricter necessary criteria.

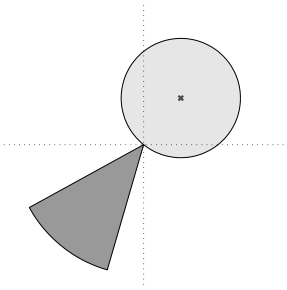

Figure 6.8: Case (−−)

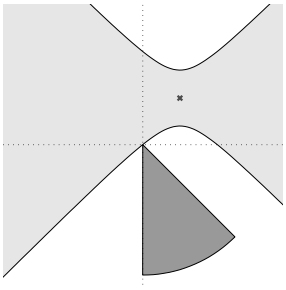

Figure 6.9: Case (+−)

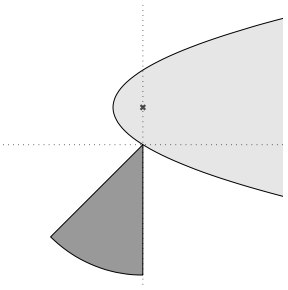

Figure 6.10: Case (0−) type 1

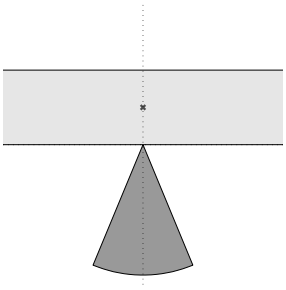

Figure 6.11: Case (0−) type 2

6.4.2 A Sufficient Criterion for Positive Stencils

For any $\mathbf{c}, \boldsymbol{\lambda} \in \mathbb{R}^d$ with $\sum_{i=1}^d \lambda_i < 0$ we construct a domain $G_{\boldsymbol{\lambda},\mathbf{c}} \supset H_{\boldsymbol{\lambda},\mathbf{c}}$. The domain $G_{\boldsymbol{\lambda},\mathbf{c}}$ will be the whole $\mathbb{R}^d$ aside from a cone centered at the origin. If we can ensure that for any $\mathbf{c}, \boldsymbol{\lambda} \in \mathbb{R}^d$, $S \in O(d)$ there is at least one point $S\mathbf{x}_i \notin G_{\boldsymbol{\lambda},\mathbf{c}}$, then $S\mathbf{x}_i \notin H_{\boldsymbol{\lambda},\mathbf{c}}$. Hence, a positive Laplace stencil exists. We call this criterion *cone criterion*.

Theorem 6.8 (Cone criterion in 2d)**.** *Let $\mathbf{c}, \boldsymbol{\lambda} \in \mathbb{R}^2$ with $\lambda_1 + \lambda_2 < 0$. There exists always a cone $C_{\mathbf{v}}$ defined by*

$$\mathbf{v} \cdot \mathbf{x} > \frac{1}{\sqrt{1+\beta^2}} \|\mathbf{x}\| \ , \tag{6.23}$$

where $\beta = \sqrt{2} - 1$ (a cone with total opening angle 45°, the direction vector $\mathbf{v}$ depends on $\boldsymbol{\lambda}$ and $\mathbf{c}$), such that $G_{\boldsymbol{\lambda},\mathbf{c}} = \mathbb{R}^d \setminus C_{\mathbf{v}}$ satisfies $H_{\boldsymbol{\lambda},\mathbf{c}} \subset G_{\boldsymbol{\lambda},\mathbf{c}}$.

Proof. We show that $H_{\boldsymbol{\lambda},\mathbf{c}}$ and $C_{\mathbf{v}}$ do not intersect. Since the problem is invariant under interchanging coordinates, we can w.l.o.g. assume that $\lambda_2 < 0$. Including the degenerate case, three cases need to be considered:

- **Case (−−):** $\lambda_1 < 0$, $\lambda_2 < 0$
  The set $H_{\boldsymbol{\lambda},\mathbf{c}}$ is the interior of an ellipse centered at $\mathbf{c}$ with $0 \in \partial H_{\boldsymbol{\lambda},\mathbf{c}}$. The vector $\mathbf{v} = -(\frac{\lambda_2}{\lambda_1}c_1, \frac{\lambda_1}{\lambda_2}c_2)$ is the outer normal vector to the ellipse. Obviously the cone $C_{\mathbf{v}}$ touches the ellipse only at the origin, as shown in Figure 6.8.

- **Case (+−):** $\lambda_1 > 0$, $\lambda_2 < 0$
  The geometry is shown in Figure 6.9. Define $\mu_1 = \frac{|\lambda_1|}{|\lambda_2|} < 1$. The domain $H_{\boldsymbol{\lambda},\mathbf{c}}$ is defined by

$$\tilde{g}(x_1, x_2) = \mu_1(x_1^2 - 2c_1x_1) - (x_2^2 - 2c_2x_2) \geq 0 \ . \tag{6.24}$$

Due to symmetry we can assume $c_1, c_2 \geq 0$. For all $\mathbf{x} \in B$, where $B = \{(x_1, x_2) | x_1 > 0, x_2 < 0, |x_1| < |x_2|\}$, the function $\tilde{g}$ satisfies

$$\tilde{g}(\mathbf{x}) = \mu_1(|x_1|^2 - 2c_1|x_1|) - (|x_2|^2 + 2c_2|x_2|) \tag{6.25}$$

$$< (\mu_1 - 1)|x_2|^2 - 2(\mu_1 c_1|x_1| + c_2|x_2|) < 0, \tag{6.26}$$

hence $H_{\boldsymbol{\lambda},\mathbf{c}} \cap C_{\mathbf{v}} = \emptyset$. The domain B is a 2d cone with opening angle 45°, where $\mathbf{v} = (\frac{1}{2}\sqrt{2-\sqrt{2}}, \frac{1}{2}\sqrt{2+\sqrt{2}})$, which proves the claim.

- **Case** $(0-)$**:** $\lambda_1 = 0$, $\lambda_2 < 0$
  Here we describe the domain by representation (6.19). Define $\mu_2 = |\lambda_2|$. The domain $H_{\boldsymbol{\lambda},\mathbf{d}}$ is defined by the relation

$$g(x_1, x_2) = d_1 x_1 + d_2 x_2 - \mu_2 x_2^2 \geq 0 \ . \tag{6.27}$$

  Due to symmetry we can w.l.o.g. assume that $d_1, d_2 \geq 0$. Two subcases have to be distinguished:

  - **Case** $d_1 \neq 0$**:**
    The setup is shown in Figure 6.10. For all $\mathbf{x} \in B$, where $B = \{(x_1, x_2) | x_1 < 0, x_2 < 0, |x_1| < |x_2|\}$, one has

$$g(x_1, x_2) = -d_1|x_1| - d_2|x_2| - \mu_2 x_2^2 < 0 \ , \tag{6.28}$$

    hence $H_{\boldsymbol{\lambda},\mathbf{d}} \cap C_{\mathbf{v}} = \emptyset$. As above, the domain B is a 2d cone with opening angle 45°.

  - **Case** $d_1 = 0$**:**
    The setup is shown in Figure 6.11. The domain $g(x_1, x_2) \geq 0$ is the set which satisfies $0 \leq x_2 \leq \frac{d_2}{\mu_2}$. Any cone contained in the domain $x_2 < 0$ proves the claim.

□

Remark 6.6. The cone criterion can in 2d be expressed in terms of angles: If any angle between two neighboring points (looked at from the central point) is no more than 45°, then a positive stencil always exists.

Theorem 6.9. *The cone criterion given in Theorem 6.8 is sharp in the following sense: For any $\varepsilon > 0$ a point setup can be constructed, such that the maximum angle is less than $\frac{\pi}{4} + \varepsilon$ and a positive stencil does not exist.*

Proof. In order to show that no positive stencil exists, it suffices to consider one specific domain $H_{\boldsymbol{\lambda},\mathbf{c}}$. We consider the domain defined by $\lambda_1 = \lambda_2 = 1$, $c_1 = \frac{1}{2}$, $c_2 = 0$, We construct a family of point clouds with maximum opening angle arbitrarily close to $\frac{\pi}{4}$, which are all

contained in the domain $H_{\boldsymbol{\lambda},\mathbf{c}}$. The setup is shown in Figure 6.12. Consider two parameters: L and l. The point cloud of neighbors is then defined as $\mathbf{x}_1 = (2,0)$, $\mathbf{x}_2 = (L+1, L)$, $\mathbf{x}_3 = (-l^2, l)$, $\mathbf{x}_4 = (-\sqrt{2}, \sqrt{2})$, $\mathbf{x}_5 = (-2, 0)$, $\mathbf{x}_6 = (-\sqrt{2}, -\sqrt{2})$, $\mathbf{x}_7 = (-l^2, -l)$, $\mathbf{x}_8 = (L+1, -L)$. The points $\mathbf{x}_1$, $\mathbf{x}_4$, $\mathbf{x}_5$ and $\mathbf{x}_6$ obviously contained in the domain $H_{\boldsymbol{\lambda},\mathbf{c}}$. For the other points we check

$$g(\mathbf{x}_2) = (L+1)^2 - (L+1) - L^2 = L > 0 \tag{6.29}$$

$$g(\mathbf{x}_3) = (l^2)^2 + l^2 - l^2 = l^4 > 0 \tag{6.30}$$

Due to symmetry, the same holds for $\mathbf{x}_7$ and $\mathbf{x}_8$.

The angles between neighboring points are less than or equal $\frac{\pi}{4}$, aside from the angles between the points $\mathbf{x}_2$ and $\mathbf{x}_3$, respectively $\mathbf{x}_6$ and $\mathbf{x}_7$, which is equal due to symmetry. The angle of point $\mathbf{x}_2$ with respect to the positive x_2-axis is given by $\tan(\varphi_2) = 1 + \frac{1}{L}$. The angle of point $\mathbf{x}_3$ with respect to the positive x_2-axis is given by $\tan(\varphi_3) = l$. The angle φ_2 can be chosen arbitrarily close to $\frac{\pi}{4}$ by letting L take sufficiently large values. Similarly, the angle φ_3 can be made arbitrarily small by choosing l sufficiently small. By choosing l and L such that $\varphi_2 < \frac{\pi}{4} + \frac{\varepsilon}{2}$ and $\varphi_3 < \frac{\varepsilon}{2}$ the maximum angle is less than $\frac{\pi}{4} + \varepsilon$. □

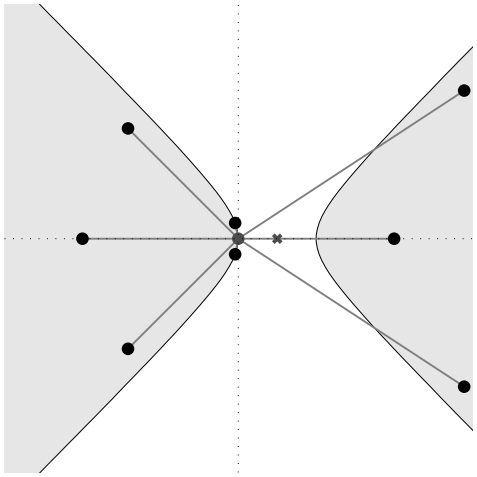

Figure 6.12: The 2d cone estimate is sharp

Remark 6.7. The construction in Theorem 6.9 yields extremely unbalanced neighbors, i.e. the ratio between distances of points is tremendous for angles close to 45°. For instance, a deviation of 1° to each side requires distances of $\|\mathbf{x}_2\| < 1.8 \cdot 10^{-2}$ and $\|\mathbf{x}_3\| > 40$. Such extreme cases will in practice be avoided by particle management, yielding positive stencils also for angles significantly larger than 45°.

Theorem 6.10 (Cone criterion in 3d)**.** *Let* $\mathbf{c}, \boldsymbol{\lambda} \in \mathbb{R}^3$ *with* $\lambda_1 + \lambda_2 + \lambda_3 < 0$*. There exists always a cone* $C_{\mathbf{v}}$ *defined by*

$$\mathbf{v} \cdot \mathbf{x} > \frac{1}{\sqrt{1+\beta^2}} \|\mathbf{x}\|, \tag{6.31}$$

where $\beta = \sqrt{\frac{1}{6}(3-\sqrt{6})}$ *(a cone with total opening angle 33.7°), such that* $G_{\boldsymbol{\lambda},\mathbf{c}} = \mathbb{R}^d \setminus C_{\mathbf{v}}$ *satisfies* $H_{\boldsymbol{\lambda},\mathbf{c}} \subset G_{\boldsymbol{\lambda},\mathbf{c}}$*.*

Proof. The following cases need to be considered:

- **Cases** $(---)$**,** $(0--)$**,** $(00-)$**:** $\lambda_1 \leq 0$, $\lambda_2 \leq 0$, $\lambda_3 < 0$
  As is the 2d degenerate case, we describe the domain by representation (6.19). Define $\mu_i = |\lambda_i|$ $\forall i = 1,2,3$. The domain $H_{\boldsymbol{\lambda},\mathbf{d}}$ is defined by the relation

  $$g(x_1,x_2,x_3) = d_1x_1 + d_2x_2 + d_3x_3 - \mu_1x_1^2 - \mu_2x_2^2 - \mu_3x_3^2 \geq 0\ , \tag{6.32}$$

  where μ_1 and μ_2 can be zero. Let us first assume that if a $\mu_i = 0$, then the corresponding $d_i \neq 0$. Due to symmetry we can w.l.o.g. assume that $d_1, d_2, d_3 \geq 0$. For all $\mathbf{x} \in B$, where $B = \{(x_1,x_2,x_3) | x_1,x_2,x_3 < 0\}$, the function g satisfies

  $$g(x_1,x_2,x_3) = -d_1|x_1| - d_2|x_2| - d_3|x_3| - \mu_2|x_2|^2 - \mu_3|x_3|^2 < 0 \tag{6.33}$$

  The domain B is not a cone, but the corresponding domain from the case $(++-)$ can be contained in it. Hence, the cone constructed in this case can be used here.

  In the case that for any $\mu_i = 0$, the corresponding $d_i = 0$, the geometry reduces to the 2d, respectively trivial 1d, case. Since in 2d the desired estimates have been shown for a cone with a larger opening angle as required in 3d, the constructions transfer to the 3d case.

- **Case** $(++-)$**:** $\lambda_1 > 0$, $\lambda_2 > 0$, $\lambda_3 < 0$
  Define $\mu_1 = \frac{|\lambda_1|}{|\lambda_3|} < 1$, and $\mu_2 = \frac{|\lambda_2|}{|\lambda_3|} < 1$. The domain $H_{\boldsymbol{\lambda},\mathbf{c}}$ is defined by

  $$\tilde{g}(x_1,x_2,x_3) = \mu_1(x_1^2 - 2c_1x_1) + \mu_2(x_2^2 - 2c_2x_2) - (x_3^2 - 2c_3x_3) \geq 0 \tag{6.34}$$

  Due to symmetry we can assume $c_1, c_2, c_3 \geq 0$. For all $\mathbf{x} \in B$, where $B = \{(x_1,x_2,x_3) | x_1,x_2 > 0, x_3 < 0, |x_1|, |x_2| < \sqrt{\frac{1}{2}}|x_3|\}$, the function $\tilde{g}$ satisfies

  $$\begin{aligned} \tilde{g}(\mathbf{x}) &= \mu_1(|x_1|^2 - 2c_1|x_1|) + \mu_2(|x_2|^2 - 2c_2|x_2|) - (|x_3|^2 + 2c_3|x_3|) && (6.35)\\ &< (\tfrac{1}{2}(\mu_1+\mu_2) - 1)|x_3|^2 - 2(\mu_1c_1|x_1| + \mu_2c_2|x_2| + c_3|x_3|) < 0. && (6.36) \end{aligned}$$

  Note that B is not a cone. However, a 3d cone can always be contained inside B. Some geometric considerations yield that the cone with maximum opening angle contained

inside B is given by

$$\beta = \sqrt{\frac{1}{6}(3-\sqrt{6})} \tag{6.37}$$

$$\mathbf{v} = \frac{1}{\sqrt{41-16\sqrt{6}}}\left(2(\sqrt{3}-\sqrt{2}), 2(\sqrt{3}-\sqrt{2}), 1\right) . \tag{6.38}$$

- **Case** $(+0-)$**:** $\lambda_1 > 0$, $\lambda_2 = 0$, $\lambda_3 < 0$
  As in the preceding degenerate cases, we describe the domain by representation (6.19). Define $\mu_1 = |\lambda_1|$ and $\mu_3 = |\lambda_3|$. As previously, the case $d_2 = 0$ reduces to the 2d case $(+-)$. Hence, w.l.o.g. we consider $d_1 \geq 0$, $d_2 > 0$, $d_3 \geq 0$. For all $\mathbf{x} \in B$, where $B = \{(x_1, x_2, x_3) | x_1, x_2, x_3 < 0, |x_1| < |x_3|\}$, the function g satisfies

$$g(x_1, x_2, x_3) = -d_1|x_1| - d_2|x_2| - d_3|x_3| + \mu_1|x_1|^2 - \mu_3|x_3|^2 < 0 . \tag{6.39}$$

  The estimate holds, since $\mu_1 < \mu_3$ and $|x_1| < |x_3|$. Note that B is not a cone. However, a 3d cone with desired opening angle can always be contained inside B. In the case $(++-)$ we have already considered a smaller domain B. The same cone can be used here.

- **Case** $(+--)$**:** $\lambda_1 > 0$, $\lambda_2 < 0$, $\lambda_3 < 0$
  Define $\mu_2 = \frac{|\lambda_2|}{|\lambda_1|}$ and $\mu_3 = \frac{|\lambda_3|}{|\lambda_1|}$. Since $\mu_2 + \mu_3 > 1$, we assume w.l.o.g. $\mu_3 \geq \frac{1}{2}$. The domain $H_{\boldsymbol{\lambda},\mathbf{c}}$ is defined by

$$\tilde{g}(x_1, x_2, x_3) = (x_1^2 - 2c_1x_1) - \mu_2(x_2^2 - 2c_2x_2) - \mu_3(x_3^2 - 2c_3x_3) \geq 0 \tag{6.40}$$

  Due to symmetry we can assume $c_1, c_2, c_3 \geq 0$. For all $\mathbf{x} \in B$, where $B = \{(x_1, x_2, x_3) | x_1 > 0, x_2, x_3 < 0, |x_1|, |x_2| < \sqrt{\frac{1}{2}}|x_3|\}$ the function $\tilde{g}$ satisfies

$$\tilde{g}(\mathbf{x}) = (|x_1|^2 - 2c_1|x_1|) - \mu_2(|x_2|^2 + 2c_2|x_2|) - \mu_3(|x_3|^2 + 2c_3|x_3|) \tag{6.41}$$

$$= \underbrace{(|x_1|^2 - \mu_2|x_2|^2 - \mu_3|x_3|^2)}_{<(\frac{1}{2}-\mu_3)|x_3|^2 \leq 0} - 2(\mu_1c_1|x_1| + \mu_2c_2|x_2| + c_3|x_3|) < 0. \tag{6.42}$$

  Here the domain B is the same as in case $(++-)$, merely reflected at the x_1, x_3 plane. Hence, a 3d cone can be placed in the same way.

□

Remark 6.8. Unlike the 2d case, the given cone opening angle is not the best possible. In 2d, we constructed the cones directly. In 3d, we constructed intermediate domains B, which do not intersect with the domain $H_{\boldsymbol{\lambda},\mathbf{c}}$, and as a second step placed a cone into these domains. The domains B being contained in one sector of the coordinate system admitted easy estimates. It is possible to place a slightly larger cone directly to the domain $H_{\boldsymbol{\lambda},\mathbf{c}}$, which

then lies in multiple sectors. However, the algebra becomes significantly more cumbersome, and the gained opening angle is only by a couple of degrees larger. In practice, a precise value for the opening angle is not important anyhow, since the point clouds will be much nicer than the worst case estimates here.

Remark 6.9. It should be pointed out that the existence of a positive stencil implies the existence of a stencil, i.e. the Vandermonde system $V \cdot \mathfrak{s} = \Delta\mathbf{b}$ has a solution. In Section 4.2.3 we have presented an example of a point configuration which made the Vandermonde system unsolvable. All vague geometric criteria like "points must not lie on one line", "points must exist in all directions" are contained in the cone criterion.

We have shown that in 2d and in 3d a positive stencil exists, if in a cone with a given opening angle always points are contained, no matter which direction the cone points to. This criterion is a strict formulation for the requirement of points being *distributed nicely around* the central point. Note that the cone criterion is not necessary, i.e. positive stencils can very well exist if the cone condition is not satisfied.

6.4.3 Condition on Point Cloud Geometry

The cone criterion derived in the previous section guarantees positive stencils. We now give conditions on the point cloud geometry and the choice of candidate points, such that the cone criterion is guaranteed to be satisfied.

Theorem 6.11. *Let the point cloud have a mesh size h, as in Definition 3.1. Let γ be the opening angle of the cone derived in Theorem 6.8 and Theorem 6.10. If the smoothing length satisfies*

$$r > \frac{1}{\sin(\gamma/2)} \frac{h}{2} \ , \tag{6.43}$$

then for every interior point which is sufficiently far from the domain boundary, a positive stencil exists.

Proof. The point cloud having a mesh size h implies that there are no holes larger in diameter than h, i.e. $\forall \mathbf{x} \in \Omega \ \exists \mathbf{x}_i \in X : \|\mathbf{x}_i - \mathbf{x}\| < \frac{h}{2}$. Figure 6.13 shows a ball with radius r around the central point and a cone with opening angle γ. Assumed, the cone contains no point, then there must be a ball of radius $\frac{h}{2}$ which contains no points. The claim follows by considering the triangle $(0, \mathbf{x}, \mathbf{x}_i)$. □

In the FPM the condition on the mesh size given in Theorem 6.11 is guaranteed by particle management. Large holes are filled by inserting new points. If a maximum hole size h is prescribed, in the interior of the domain (and sufficiently far away from the boundary), one

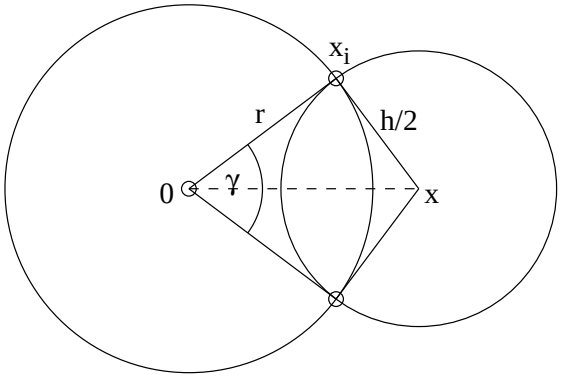

Figure 6.13: Relation between circular neighborhood radius and mesh size

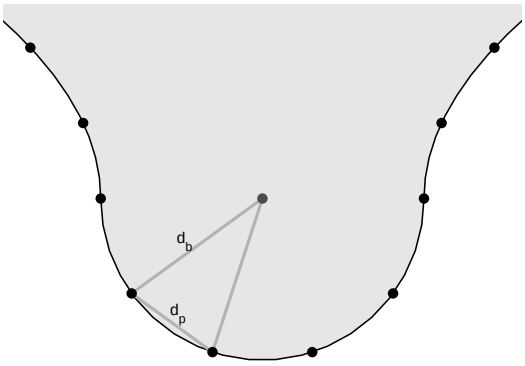

Figure 6.14: Guaranteeing the cone criterion close to the boundary

has to consider all points in a circular neighborhood of radius large enough. Then a positive stencil exists for any such point. The ratios given by Theorem 6.11 are

$$\frac{r}{h/2} > \sqrt{1+\tfrac{1}{\beta^2}} = \begin{cases} \sqrt{4+2\sqrt{2}} = 2.61 & \text{in 2d} \\ \sqrt{7+2\sqrt{6}} = 3.45 & \text{in 3d} \end{cases} \tag{6.44}$$

Again note that for the 3d case, the given ratio is not the sharpest possible. With significantly more algebra one can lower the value to $\sqrt{6+2\sqrt{6}} = 3.30$. Both values are valid for the worst case scenario. In practice the given point clouds are typically much nicer, such that already significantly smaller ratios lead to positive stencils.

Theorem 6.11 is valid for any interior point which is far enough from the boundary, such that the mesh size criterion guarantees points to exist between the point in consideration and the boundary. For a layer of interior points close to the boundary, the cone criterion has to be enforced by other means. A straightforward approach, which in shown in Figure 6.14, is the following.

1. The boundary is staffed with boundary points which are sufficiently dense. For instance, in 2d, the boundary points must have a distance of at most d_p.
2. Any interior point must have a minimum distance from the boundary. For instance, in 2d, any interior point must have a distance $d_\mathrm{b} > \frac{4}{\pi} d_\mathrm{p}$ from the boundary.

6.5 Neumann Boundary Points

So far we have only considered interior points and Dirichlet boundary points. For both cases positive stencils can be obtained. In this section, we present how Neumann boundary points

can be incorporated into the MPS method.

Assume that in the neighborhood of Neumann boundary data the boundary $\partial\Omega$ is C^1, such that the normal $\mathbf{n}$ can be uniquely defined. The conditions for a consistent approximation of a normal derivative are derived in Section 4.2 as

$$\sum_{i=1}^{m} \mathbf{x}_i \mathfrak{s}_i = \mathbf{n} \ . \tag{6.45}$$

The central stencil entry is obtained according to (4.17) as $\mathfrak{s}_0 = -\sum_{i=1}^{m} \mathfrak{s}_i$. As derived in Section 4.2.2, higher order approximations require in particular the quadratic constraints (4.23), which are

$$\sum_{i=1}^{m} \left(\mathbf{x}_i \cdot \mathbf{x}_i^T \right) \mathfrak{s}_i = 0 \ . \tag{6.46}$$

A negative aspect about higher order approximations at Neumann boundary points is the following

Theorem 6.12. *A higher order approximation at a Neumann boundary point can never yield a positive stencil.*

Proof. Summing over the diagonal in (6.46) yields the relation

$$\sum_{i=1}^{m} \|\mathbf{x}_i\|_2^2 \mathfrak{s}_i = 0 \ , \tag{6.47}$$

which a positive stencil can only satisfy if $\mathfrak{s} = 0$. This stencil, however, is inconsistent. □

In the following, we consider for every boundary point in consideration a local coordinate system, in which we set w.l.o.g. the normal vector $\mathbf{n} = (-1, 0)$ in 2d, respectively $\mathbf{n} = (-1, 0, 0)$ in 3d. The domain boundary $\partial\Omega$ is then in the origin tangent to the normal plane, which equals the y-z-plane. Figure 6.15 shows the setup. A desirable normal derivative stencil has a positive central value and all other values are negative. Such a stencil can be incorporated into the desired L-matrix. For the following analysis, we would like to keep the similarity to the sign constraints for Laplace stencils. Hence, we multiply the stencil entries and the normal vector by -1, i.e. we seek for solutions to the problem

$$V \cdot \mathfrak{s} = \mathbf{n} \ , \ \mathfrak{s} \geq 0 \ . \tag{6.48}$$

In the particular setup, the linear system reads as, in 2d

$$\begin{pmatrix} x_1 & \dots & x_m \\ y_1 & \dots & y_m \end{pmatrix} \cdot \begin{pmatrix} \mathfrak{s}_1 \\ \vdots \\ \mathfrak{s}_m \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{6.49}$$

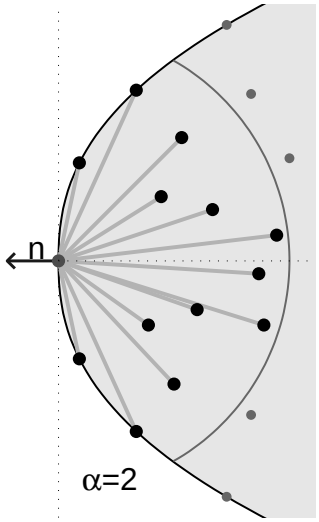

Figure 6.15: Neumann point with circular neighborhood

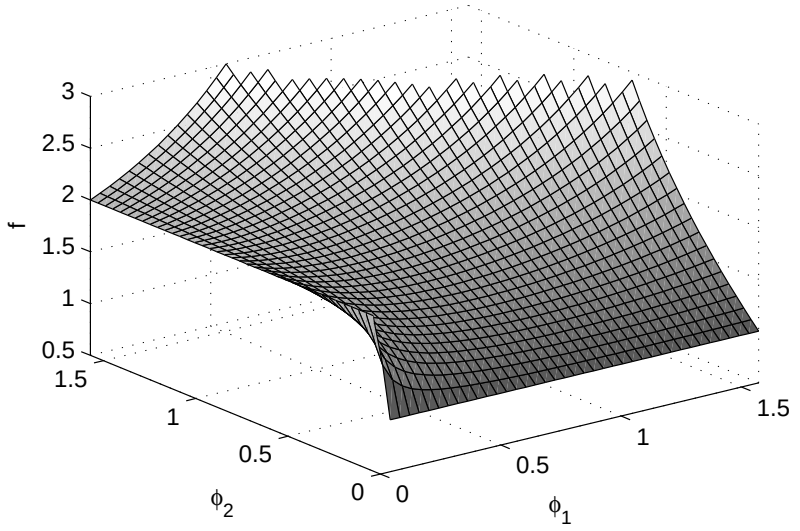

Figure 6.16: Function $f(\phi_1, \phi_2)$

and in 3d

$$\begin{pmatrix} x_1 & \dots & x_m \\ y_1 & \dots & y_m \\ z_1 & \dots & z_m \end{pmatrix} \cdot \begin{pmatrix} \mathfrak{s}_1 \\ \vdots \\ \mathfrak{s}_m \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{6.50}$$

In the following, we consider the 3d case. The 2d case works either analogously or is a special case of the 3d geometry.

For an easier analysis, we further restrict to the case of a locally convex domain, i.e. all neighboring points satisfy $x_i \geq 0$. A proper treatment of positive stencils in the case of points with $x_i < 0$ are present requires more care. On the other hand, including such points into an approximation of a normal derivative is often times not be desirable anyhow. Thus, one can a priori exclude such points from the set of considered neighbors.

6.5.1 Conditions for Positive Neumann Stencils

An equivalent geometric criterion for the existence of positive Neumann stencils yields the following

Theorem 6.13. *A first order approximation at a Neumann boundary point has a positive stencil solution, iff the points' projections onto the normal plane do not lie all in one and the same half space.*

Proof. As in Section 6.4 we employ Farkas' lemma. System (6.50) has no solution $\mathfrak{s} \geq 0$, iff the system $V^T \cdot \mathbf{w} \geq 0$ has a solution satisfying $w_x < 0$, where $\mathbf{w} = (w_x, w_y, w_z)^T$.

Let no positive stencil exist. Then $\mathbf{w} \in \mathbb{R}^d$ with $w_x < 0$ exists, such that $V^T \cdot \mathbf{w} \geq 0$, i.e. $w_x x_i + w_y y_i + w_z z_i \geq 0 \; \forall i$. This is equivalent to

$$\mathbf{k} \cdot \begin{pmatrix} y_i \\ z_i \end{pmatrix} \geq x_i \quad \forall i \; , \tag{6.51}$$

where $\mathbf{k} = (\frac{w_y}{|w_x|}, \frac{w_z}{|w_x|})$. This means that the (y, z) projection of all points lies in one and the same half space (in the direction of vector $\mathbf{k}$). More precisely, every point with $x_i > 0$ lies strictly in the half space, while points with $x_i = 0$ can lie on the line perpendicular to $\mathbf{k}$.

Conversely, assume that the y-z projection of all points lies in one and the same half space. Let I be the indices of all points in consideration. Define $I_p = \{i \in I : \; x_i > 0\}$. Consider w.l.o.g. the case $z_i \geq 0 \; \forall i$, where additionally $z_i > 0 \; \forall i \in I_p$. Choose

$$\mathbf{w} = \left(-1, 0, \frac{\max_{i \in I} x_i}{\min_{i \in I_p} z_i} \right) \; . \tag{6.52}$$

Then for all $i \in I_p$ it holds

$$\mathbf{w}^T \cdot \mathbf{x}_i = -x_i + \frac{\max_{i \in I} x_i}{\min_{i \in I_p} z_i} z_i \geq -x_i + \max_{i \in I} x_i \geq 0 \; , \tag{6.53}$$

and for all $i \in I \setminus I_p$ one has $\mathbf{w}^T \cdot \mathbf{x}_i \geq 0$, since $x_i = 0$. Thus, no positive stencil exists. □

Remark 6.10. In 2d, the analogous criterion can be derived, which can be interpreted as: There must be points both below and above the local x-axis. Alternatively: The cross product $\mathbf{n} \wedge \mathbf{x}_i$ must not have the same sign for all points.

6.5.2 Linear Minimization for Neumann Points

As done in Section 6.3 for Laplace stencils (6.10), we formulate a linear minimization problem also for Neumann stencils

$$\min \sum_{i=1}^{m} \frac{\mathfrak{s}_i}{w_i}, \; \text{s.t. } V \cdot \mathfrak{s} = \mathbf{n}, \; \mathfrak{s} \geq 0 \; . \tag{6.54}$$

We consider a distance weight function $w(d) = |d|^{-\alpha}$, i.e. $w_i = r_i^{-\alpha}$. As we will motivate in Section 6.5.3, choices of $\alpha < 1$ will result in points far away being selected. For a value of $\alpha = 1$ the stencil selection will only depend on the angles. Desired in our context are values $\alpha > 1$, such that close by points will be preferred.

The linear program (6.54) can be solved in the same manner as the program (6.10) for interior points, by a method as described in Section 6.7.

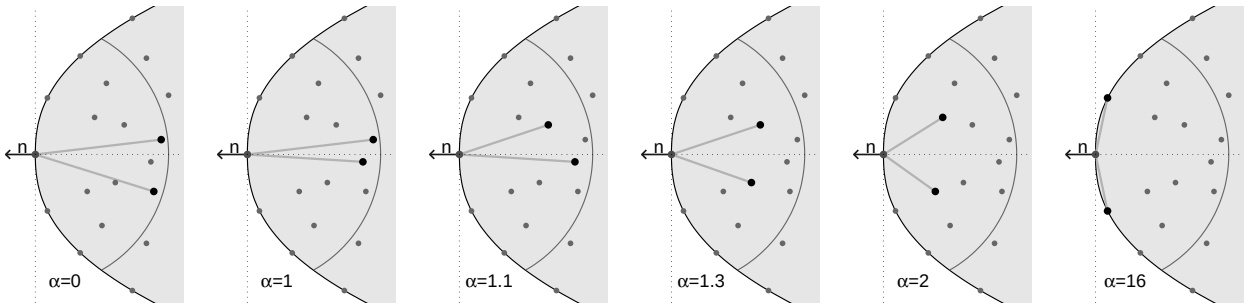

Figure 6.17: Minimal positive Neumann stencil for various values of α

6.5.3 Geometric Interpretation of Neumann Stencils

Consider a 2d geometry, as before in local coordinates, i.e. $\mathbf{n} = (1, 0)^T$. The MPS approach selects a basis solution, i.e. two points $\mathbf{x}_1$ and $\mathbf{x}_2$, as shown in Figure 6.17. Due to Theorem 6.13, the positivity constraint implies the second components to have opposing signs. We write w.l.o.g. $\mathbf{x}_1 = (x_1, -y_1)$ and $\mathbf{x}_2 = (x_2, y_2)$, where $x_1, x_2, y_1, y_2 \geq 0$. Or in polar coordinates $\mathbf{x}_1 = r_1(\cos(\phi_1), -\sin(\phi_1))$, $\mathbf{x}_2 = r_2(\cos(\phi_2), \sin(\phi_2))$.

Using relation (6.49) the stencil values can be obtained as

$$\begin{pmatrix} \mathfrak{s}_1 \\ \mathfrak{s}_2 \end{pmatrix} = \frac{1}{x_1 y_2 + x_2 y_1} \begin{pmatrix} y_2 \\ y_1 \end{pmatrix} = \frac{1}{\sin(\phi_1 + \phi_2)} \begin{pmatrix} r_1^{-1} \sin(\phi_2) \\ r_2^{-1} \sin(\phi_1) \end{pmatrix} . \tag{6.55}$$

Here the weights are $w_i = r_i^{-\alpha}$. The MPS approach selects two points, such that their positive stencil minimizes

$$f = \sum_{i=1}^{n} \frac{\mathfrak{s}_i}{w_i} = \frac{r_2^{\alpha-1} \sin(\phi_1) + r_1^{\alpha-1} \sin(\phi_2)}{\sin(\phi_1 + \phi_2)} . \tag{6.56}$$

Obviously values of $\alpha > 1$ are required to prefer close by points. Interpreting the distances as parameters $k_1 = r_1^{\alpha-1}$ and $k_2 = r_2^{\alpha-1}$, one obtains a function, which should be small in value

$$f(\phi_1, \phi_2) = \frac{k_2 \sin(\phi_1) + k_1 \sin(\phi_2)}{\sin(\phi_1 + \phi_2)} . \tag{6.57}$$

The function is shown in Figure 6.16 for the parameters $k_1 = 2$ and $k_2 = 1$, i.e. point $\mathbf{x}_1$ is further away from the boundary point than point $\mathbf{x}_2$.

For one angle zero, the resulting function value is constant: $f(\phi_1, 0) = k_2$ and $f(0, \phi_2) = k_1$. On can easily check that the smallest parameter (here k_2) is the minimum function value of f. The function is small, if the angle of a close by point is as small as possible. If a point is positioned exactly on the x-axis, then this point can together with the central point yield a consistent approximation to the normal derivative. Of course, in general, this will not be the

case. If, however, a point close to the x-axis is found, then there is no requirement for the angle of the second point to the also small. If one is not in the case of one angle being small, an angle of about π between the two points yields small values of f. The MPS approach often times yields such a setup.

Figure 6.17 shows the choice of Neumann boundary points for various values of α. Small values of α focus on choosing points pointing into the normal direction. With increasing α, the distance to the boundary point becomes more important in the minimization, and points closer by are selected. Until, finally for $\alpha \geq 16$, the two neighboring boundary points are incorporated into the normal approximation. Obviously, this is not a good thing to happen. We conclude that on the one hand α must be large enough to have points close by, on the other hand α must not be too large in order to have a stable approximation. A good choice is $\alpha = 2$.

6.6 Minimal Stencils and Matrix Connectivity

The MPS method, as presented in Section 6.3, yields positive stencils for any interior and Neumann boundary point, hence the system matrix is an L-matrix.

Due to Theorem 6.3, the L-matrix is an M-matrix, if it is essentially irreducible, i.e. every interior and Neumann boundary point is connected to a Dirichlet boundary point. We start showing

Theorem 6.14. *Consider the Poisson problem (4.1) on a bounded domain Ω which has no holes (i.e. only an outer boundary). With a MPS discretization every interior point is connected to a boundary point.*

Proof. Let I be the index set of the points. Assume there is a point with index $i \in I$ which is not connected to a boundary point. Define $I_i = \{j \in I : i \text{ is connected to } j\}$. Every point whose index is in I_i is not connected to a boundary point. Hence, the set of points given by the index set $I_i \subset I$ forms an island inside the domain Ω which does not reach the boundary. Consider the convex hull of this set of points, and further consider an island point which spans the convex hull. This spanning point only uses points in its stencil which lie inside the island. Since the convex hull is convex, these neighboring points lie in one and the same half space, which violates the necessary condition on positive stencils given by Theorem 6.7. □

Remark 6.11. Assumed that the MPS method takes care that every Neumann boundary point is connected to an interior point (i.e. the setup shown in Figure 6.17 for $\alpha = 16$ does not happen), then due to Theorem 6.14 also every Neumann boundary point is connected to the boundary.

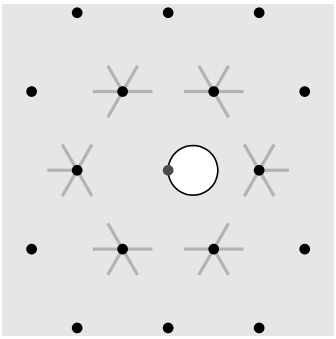

Figure 6.18: An interior point is not reached

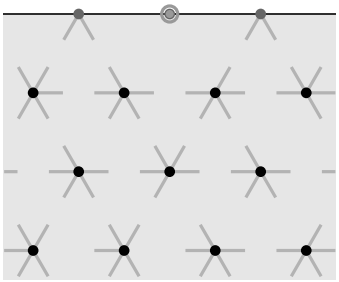

Figure 6.19: Failure to reach a Dirichlet point

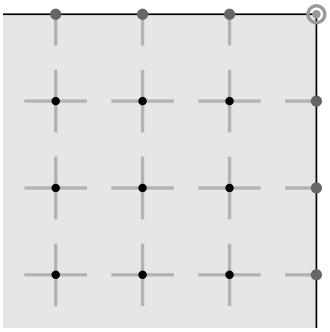

Figure 6.20: Connection failure with a regular grid

Remark 6.12. The proof of Theorem 6.14 does not extend to interior boundaries, since the convex hull argument cannot be applied to an island of points around a hole. Indeed, it is possible that one point or a small set of points is not used in a stencil approximation of any point. Figure 6.18 shows such a setup. The point in the middle is not used in any stencil. This is no problem if it is an interior point itself (since it uses the other points in its stencil). However, if there is an interior hole whose boundary is discretized by this one single point, the boundary data is not considered in the computation. However, in the MPS method, such a case can only happen if the hole is as small as the point resolution. If it is larger, then its boundary appears locally flat, and boundary points will be incorporated in stencils of neighboring interior points.

Theorem 6.14 only states that every interior point is connected to the boundary. We need more, namely every point be connected to a Dirichlet boundary point. Unfortunately, this cannot be concluded from the MPS directly, as the following example indicates.

Example 6.6. Consider the setup shown in Figure 6.19. At the boundary, there is one single Dirichlet boundary point, the rest are Neumann boundary points. The setup shows that it is in principle possible that no other point is connected to this Dirichlet point. The matrix would not be essentially irreducible and thus singular. Note that such a setup can also happen in the case of regular rectangular grids. Figure 6.20 shows a geometry with one single Dirichlet boundary point. The interior points are discretized by regular five point stencils. Obviously, the Dirichlet boundary data is not considered, yielding exactly the same problem as shown in Figure 6.19. Further note that also with least squares methods it is possible that a stencil value equals exactly zero due to the specific geometric setup. Also in such a case one would obtain a singular system matrix.

As soon as there is a whole line or area of Dirichlet boundary points, the neighboring interior points must use the Dirichlet data, due to Theorem 6.7. However, the problem with sparse

Dirichlet data remains. While it can easily controlled in the case of a regular grid, in the meshless MPS the situation is more complicated. It might be required to force certain points into the stencil. In Section 6.7.3 we will remark how this aspect can be included into the LM formulation.

Concluding, the mentioned problems require care. The MPS yields non-symmetric stencils with very few entries. This can cause problems in complicated geometries. Extreme geometry cases might require careful coding. However, if these problem are taken care of, Theorem 6.14 guarantees the matrix to be essentially irreducible. **Matrices discretized by the MPS method are M-matrices.**

6.7 Optimization Routines

The MPS method requires for every interior and every Neumann boundary point a linear program to be solved. For an interior point, the program (6.10) has 5 constraints in 2d and 9 constraints in 3d. For a Neumann boundary point, the program (6.54) has 2 constraints in 2d and 3 constraints in 3d. The number of variables is the number of neighboring candidate points, which is between 10 and 30 in 2d and between 20 and 50 in 3d. In Section 6.4.3 we have shown that for interior points, considering all points in a circular neighborhood with a sufficiently large radius guarantees the existence of a positive stencil. Similarly, we have provided criteria for Neumann boundary points in Section 6.5.1.

Generally, solving the systems for Neumann boundary points is negligible in effort, since firstly they are smaller and secondly there are in general more interior points. The systems for the interior points, on the other hand, require fast solution methods. In the FPM, in every time step of a flow computation, one or more Poisson problems have to be solved. Each Poisson problem requires to set up the linear system (unless information from previous computations can be used). Setting up the linear system requires to solve as many linear programs as there are points in the point cloud. Altogether, this can be a tremendous amount of small linear programs to be solved. Hence, efficient methods for small linear programs are of interest. Unfortunately, in the field of linear optimization, there is not much interest in solving small linear programs fast. Typically, if small systems are considered, computation time is not an issue. Additionally, the linear programs in the MPS context have a special structure, arising from the geometric background. In particular, the Vandermonde matrix is full, while in many optimization applications, the matrices are sparse.

Due to the structure of our systems, we presume that the *simplex method* is the best approach for the arising linear programs. The basic algorithm is described in the book of Chvátal [Chv83]. The simplex algorithm starts with a feasible basic solution (i.e. a minimal positive stencil) and successively improves the function to be minimized by stepping from vertex

to vertex in the polyhedron of feasible solutions. The stepping is done along edges, i.e. in each step one variable leaves the basis and another one enters. In the geometric background this means that in each step one point leaves the positive stencil and is replaced by another one, such that a new positive stencil is obtained, however, with a smaller function value. Theoretically, it is possible that the simplex algorithms visits every vertex of the polyhedron of feasible solutions, thus the worst case performance is not polynomial. In practice, however, the simplex method often finds the optimal solution after just a few steps. In the context of the MPS method, the geometric interpretation gives rise to the assumption that the same good performance is to expect here.

If no information about feasible solutions is available, one can obtain the starting positive stencil by the *two phase simplex algorithm*. In the first phase, the Vandermonde matrix is extended by an identity matrix, and as many variables are added, as there are constraints. The simplex algorithm is run on this augmented system until every new variable has left the basis, i.e. a positive starting stencil is found.

We have implemented the two phase simplex algorithm into the FPM Fortran code. Since the routine has to be run many times, one can save a large amount of effort by not performing initial checks, which many external solvers do. In order to gain an impression about the performance, we consider a test geometry (as in Section 8.2), generate linear programs for 42000 points (each is 9×50 in size), and apply various optimization methods to solve these problems. One the one hand we run the implemented simplex method. On the other hand, we apply the software CPLEX [CPL]. Firstly, we consider the simple primal simplex solver in CPLEX, then run other solvers in CPLEX. Of particular interest is the application of an interior point method. These are theoretically preferable, since the worst case performance is polynomial. In practice, however, there are many examples in which simplex based approach beat interior point methods.

The outcome of the comparison is shown in Figure 6.21. The implemented simplex method performs well in comparison with the CPLEX routines. We assume that CPLEX has an overhead for testing stability, which is normally negligible, but not for these small problems to be solved many times. CPLEX offers to use optimization heuristics (denoted *presolve* in Figure 6.21), which for many problem can accelerate the solution procedure significantly. However, this is not the case here. An important observation, which seconds our presumption that simplex is the preferable method here, is the fact that interior point methods perform significantly worse for the problems in our context. Finally note that the listed times are merely the solution times. If external optimization routines are called from the FPM code, an additional communication overhead will have to put up with. We conclude that a direct implementation of a basic version of the simplex method performs best for the given problems. Of course, there is still a large potential to optimize the solver. In Section 6.7.2 we outline various ideas of improvement.

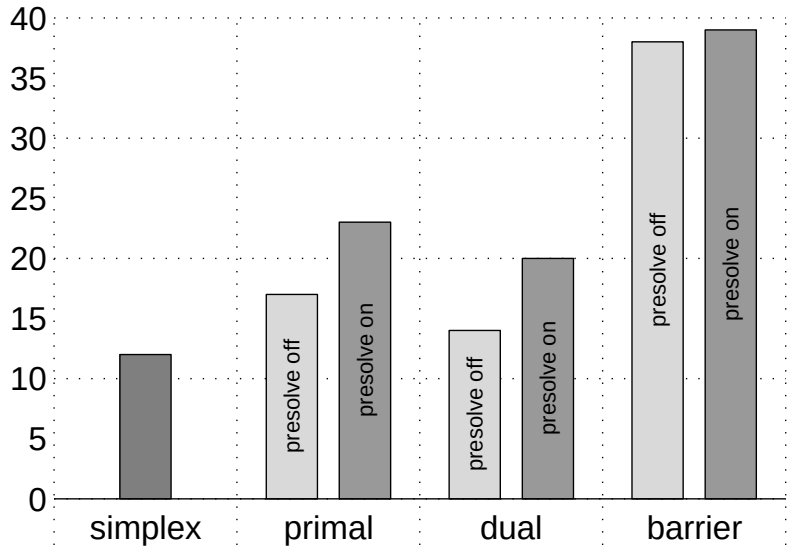

Figure 6.21: Run times for various optimization methods

6.7.1 Effort of Simplex in Comparison to Least Squares

The MPS provides an alternative to least squares approaches. These require setting up and solving a least squares problem for each interior and Neumann boundary point. The MPS yields smaller stencils, thus the resulting system matrices are sparser than with least squares approaches. However, the MPS can only compete with least squares if the computation of the stencils themselves is not significantly more expensive. In this section we compare the expected complexities of the LM and the QM approach in stencil computations.

Comparing the methods directly is not easily possible, since the complexity of the simplex algorithm depends on the number of basis changes done till an optimal solution is found. In practice, for nice point clouds, we observe the number of simplex steps to be in the order of the number of constraints.

Of the various least squares approaches, the ILLS method requires the lowest number of operations to compute a Laplace stencil. The numerical effort of the ILLS method has been estimated in Section 5.5.3. For m points included in the neighborhood, it is slightly more than $k(k+1)m$, where k is the number of constraints. The values are $30m$ in 2d, and $90m$ in 3d.

The simplex method applied to a linear program which a feasible starting solution is known to, is a generalization of the Gaußian elimination method. In each step of the simplex method one pivot step has to be performed. In the basic simplex method, this requires adding a multiple of one row to all other rows. These are km operations in each step. If the number of steps is about k, the overall number of operations required is k^2m, which equals

the complexity of the LLS method.

Note that the number of operations includes only the solution phase. If a feasible solution has to be found in a first phase, the effort roughly doubles. On the other hand, the number of operations in a pivot step are mostly additions. In modern computer architectures a pure operation count does not yield a strict comparison anyhow. The messages are:

- The simplex method's complexity is in the same order as least squares approaches.
- Finding a feasible starting solution without a first phase simplex run would accelerate the method significantly.

In Section 8.2 we compare the run times of the ILLS and the MPS method for the generation of stencils in a numerical experiment.

6.7.2 Improvement of the Simplex Method

We have implemented the simple two phase simplex method to compute minimal positive stencils. Obviously, there is a large potential of improvement. The following ideas can be considered and investigated:

- **Simplex variants**
  The revised simplex, the dual simplex, or the primal dual simplex method [Chv83] can be implemented. The run time comparison shown in Figure 6.21 indicates that the dual simplex is a good candidate for a fast method.
- **Pivot rule**
  It can happen that the choice of a new basis change is not unique. A *pivot rule* has to be provided, which new point to select. Simple pivot rules can yield a cycling of the method, i.e. the function value is not improved and the method dead locks. Various rules exist on how to prevent the simplex method from cycling (e.g. Bland's rule). We have experienced that for our class of linear programs a steepest edge rule works best. Only if the rule fails, one can switch to Bland's rule to prevent cycling. However, there is a large selection of rules and criteria [Van01], which can significantly improve the performance.
- **Simplex Improvements**
  There is wide variety of improvements and tricks on how to squeeze out more performance of the simplex method, some of which are presented in the book of Vanderbei [Van01].

- **Guessing a solution**
  The optimization would be significantly accelerated if a feasible starting solution was found by geometric criteria. Motwani and Raghavan [MR95] present randomized approaches which can guess a feasible solution. These perform well if the number of variables significantly exceeds the number of constraints. In our case, however, we do not have enough neighboring points for these approaches to improve the performance significantly.

  There is a wide selection of heuristics on how to guess a feasible solution and, even more, the optimal solution, as presented by Lustig, Marsten and Shanno [LMS94]. If such an approach works, the whole optimization procedure can be saved. It is an open question whether such approaches succeed for the linear programs arising in our context.

  Another idea, which goes into the direction of guessing the optimal solution, is to employ information about the point geometry. Knowing the background, we can perform such a task, while an optimization method applied to the Vandermonde matrix does not necessarily "see" this background. Examples can be

  – If one is in a hybrid concept, in which Delaunay neighborhood information is easily available, the Delaunay neighborhood is a good candidate for a positive stencil. As shown in Section 6.9, this guess need not yield a positive stencil, but it often times does. If the Delaunay neighborhood has too few points, one can add the closest other points. If Delaunay yields more points than constraints, points can be erased, or various combinations of selections can be tested.

  – The 5 closest points can be used as a first guess. This is only a good guess for comparably regularly structured point clouds.

  – The inverse convex hull criterion can be used, given it can be implemented efficiently.

  Note that it is typically not expensive to test a small number of first guesses. If one of these is feasible or even optimal, a great speedup in the simplex method is gained.

6.7.3 Forcing a Specific Point into the Stencil

In Section 6.6 we have shown that a specific point might not be used in a stencil of neighboring points (see Figure 6.19). If this point is of importance (for instance because it is the only Dirichlet boundary point), it might be required to force this specific point into the stencil, even if the linear function is then not minimized anymore. Our new problem formulation is: Among all positive stencils which involve a given point we search the one which minimizes

the given linear function. Or in the language of linear programming: Among all feasible basic solutions which have a given variable in their basis we search the optimal one.

At first glance, this additional restriction does not seem to significantly complicate the problem. It does. The restriction of a specific variable being in the basis cannot be rewritten as an additional condition, such as $\mathfrak{s}_i > 0$ or $\mathfrak{s}_i \geq \varepsilon$. Also the simplex method cannot be augmented to take care of this requirement. Fukuda, Liebling and Margot prove the following

Theorem 6.15. *Including the requirement of a specific variable being in a basis solution into a linear program makes the problem as complex as listing all vertices of the convex polyhedron.*

Proof. The proof is given in [FLM97]. □

While linear programs can be solved in polynomial time, for instance by the ellipsoid method by Khachiyan [Kha79] or by Karmarkar's algorithm [Kar84], the constraint of a specific point being in the basis makes the problem NP-hard. This implies in particular that it is not possible to obtain the sought stencil with a few manipulations from the optimal positive stencil, which is obtained by the unconstrained MPS method (believing that P≠NP).

In practice one can do the following approaches:

- Test the k possible basis changes, in which the point of interest enters and another point leaves the stencil (i.e. check the k neighboring points in the polyhedron). From any such stencil which is positive take the one with the minimal function value. Of course this does not yield the solution to the above problem, since the optimal constrained stencil need not be a neighboring vertex to the MPS. Also it is possible that none of the such constructed stencil is positive.
- Add the point of interest into the stencil and compute a $k+1$ point stencil. This can be done by a least squares approach (which might not yield a positive stencil), or by walking along an edge of the feasibility polyhedron, but not the whole way to a neighboring vertex.

6.8 Implementation of the MPS Method

We have provided criteria for the MLS method to work and outlined that it should be able to compete in generating a discretization of a Poisson problem. We propose the following implementation of the MPS method.

1. Assume a feasible point cloud is provided or constructed, which has a mesh size h.

2. For each interior point:
   (a) Consider neighbors in a circular neighborhood with radius $r = \beta_{\text{mean}} h$.
   (b) Set up the Vandermonde matrix. Compute weights.
   (c) (Possibly guess a positive stencil.)
   (d) Apply the simplex method (or variant). If a solution is obtained, take it. If not...
   (e) Consider neighbors in a circular neighborhood with radius $r = \beta_{\text{max}} h$.
   (f) Apply the simplex method (or variant). A solution is obtained.
3. For each Neumann boundary point:
   (a) Consider neighbors in a circular neighborhood with radius large enough such that criterion given by Theorem 6.13 is satisfied.
   (b) Set up the Vandermonde matrix. Compute weights.
   (c) Apply the simplex method (or variant). A solution is obtained.
4. For each Dirichlet boundary point:
   (a) Incorporate identity row into matrix.

For interior points it pays to start with a smaller than necessary radius, since this way the resulting programs have significantly fewer variables. Good choices are $\beta_{\text{mean}} \in [1, 1.5]$. In practice this approach yields a positive stencil in almost all cases. In the rare case there is no stencil found, the radius is extended to the value given by Theorem 6.11, i.e. $\beta_{\text{max}} = 2.61$ in 2d, respectively $\beta_{\text{max}} = 3.45$ in 3d. Note that this approach does not necessarily yield the optimal positive stencil, which can theoretically use a point further away than β_{max}. In practice, however, the resulting stencils turn out to be optimal or at least very close to optimal.

6.9 Neighborhood Concepts Revisited

In Section 3.5 we have presented various approaches on how to define the neighborhood of a point in a point cloud. The presented criteria were geometric. The MPS method can also be interpreted as a neighborhood criterion. It yields a preferred neighborhood with respect to the Laplace operator (or a normal derivative at Neumann boundary points, but these we neglect in this consideration).

Also of interest is the aspect of positive stencils. For a given point cloud, we can consider various neighborhood concepts, and compute the Laplace stencil by the ILLS method. Of interest is which of the neighborhood concepts yield positive stencils. The MPS method does so by construction. For each other method we provide an example in which the ILLS method does not yield a positive stencil.

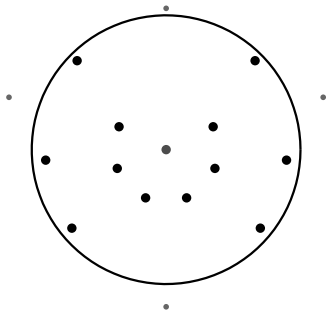

Figure 6.22: Large circular neighborhood

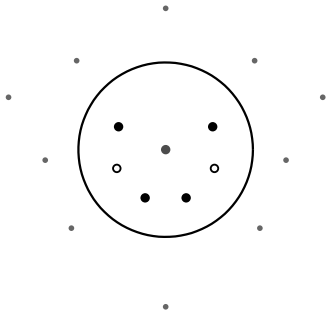

Figure 6.23: Small circular neighborhood

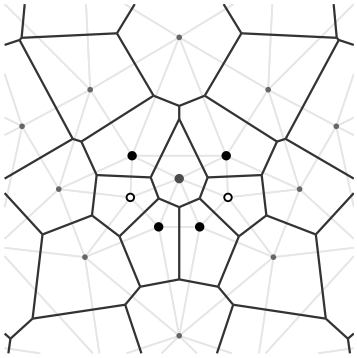

Figure 6.24: Delaunay neighborhood

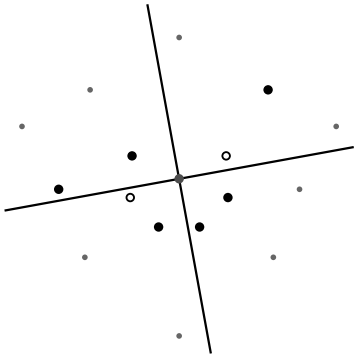

Figure 6.25: Four quadrant neighborhood

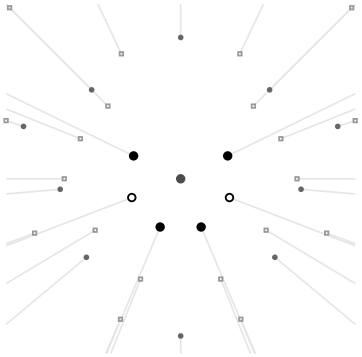

Figure 6.26: Inverse convex hull neighborhood

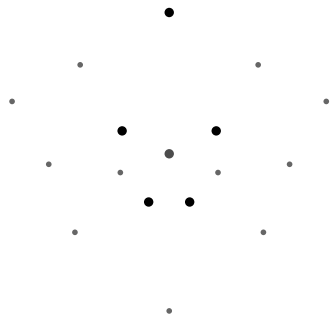

Figure 6.27: Positive stencil neighborhood

Figure 6.22 shows the circular neighborhood with a large radius. The obtained stencil is positive. Figure 6.23 shows the circular neighborhood with a smaller radius. The obtained stencil has negative entries (denoted by the marker with a hole). Note that a non-positive stencil can very well occur with a large circular neighborhood radius. Figure 6.24 shows the Delaunay neighborhood and Figure 6.26 shows the inverse convex hull neighborhood. Both yield the same six neighbors as the small circular neighborhood. Consequently, both do not result in positive stencils. Figure 6.25 shows the four quadrant neighborhood, which selects eight neighboring points. Also for this selection, negative stencil entries occur. In comparison, the MPS neighborhood selects five points which have a positive stencil, as shown in Figure 6.27.

6.10 Generalizations of the MPS Approach

The MPS method can be interpreted as a neighborhood selection approach. This gives rise to various generalizations of other methods. We outline the idea of using the MPS neighborhood for improving the performance of least squares methods, as well as the concept of MPS cells.

6.10.1 Hybrid Approximation Methods

The MPS method yields minimal stencils. This results in a very sparse system matrix. In terms of solving the arising systems, this is an advantage, since the matrix can be applied very fast. On the other hand, in some applications a larger number of stencil entries might be desired. In these cases, one can combine the MPS method with other methods to obtain hybrid approximation methods, i.e. the MPS method can be used to improve the performance of classical least squares methods. Among many others, the following ideas are promising:

- Use the MPS method as a neighbor selection method, then add points to the neighborhood, for instance all Delaunay neighbors and/or a certain number of the closest points. Apply the favorite least squares method to compute a stencil for the given set of points.
- Use the MPS method as a definition for the local smoothing length, i.e. consider all points closer or at equal distance than the farthest MPS points. Use a least squares approximation on this set of points.
- Use the MPS stencil as a basis. Add further points, e.g. by one of the above criteria, then obtain a new stencil, however now with a method which guarantees the positivity of a stencil. Here, a sign constrained quadratic minimization, solved by a Karush-Kuhn-Tucker solver, can be a good option.

6.10.2 MPS Cells

Given a point cloud X, the Voronoi cells [Vor07] yield a partition of the whole $\mathbb{R}^d$ into cells. In each cell all points have a common property. In the i^{th} cell every point is closer to the point $\mathbf{x}_i$ than to any other point $\mathbf{x}_j \in X$.

The MPS neighborhood can also be used to partition a part of the $\mathbb{R}^d$ into cells. For any point $\mathbf{x} \in \mathbb{R}^d$ consider $\mathbf{x}$ as a central point and the point cloud X as other points. Now compute the MPS stencil, i.e. select 5 (in 2d) respectively 9 (in 3d) points out of the point cloud. Of course, this might not be possible, for instance outside of the convex hull of X, no positive stencils exist due to Criterion 6.7.

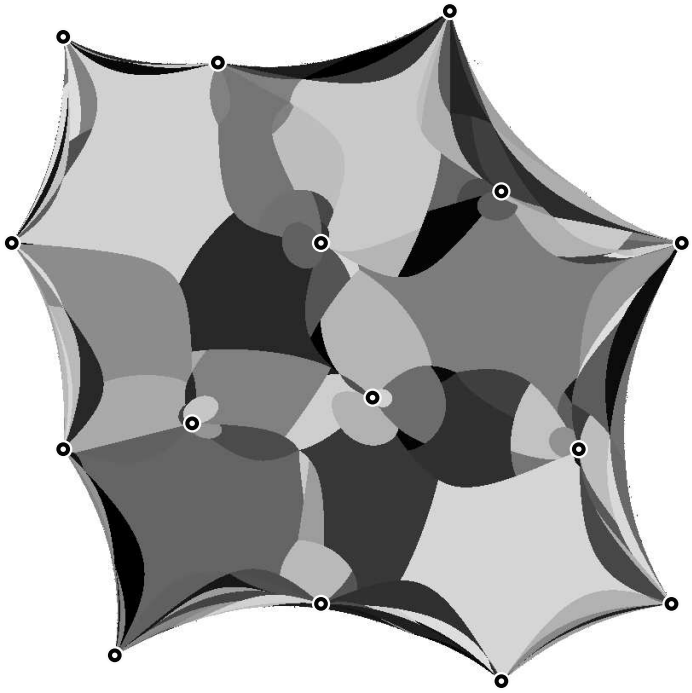

Figure 6.28: MPS mosaic for a given set of points

A MPS cell is defined as the set of all points which use the same neighboring points in their MPS stencil. Cells can have common edges, but will in general not overlap. Also, there might be points even inside the convex hull of X which are in no cell, since no positive stencil exists.

Figure 6.28 shows a mosaic of MPS cells for a given point cloud. Note that the MPS mosaic depends on the distance weight function (here $w(d) = |d|^{-4}$). While Voronoi cells have the points of the cloud in their middle, the MPS cell boundaries often meet in cloud points. A Voronoi cell is associated to a single point. A MPS cell is associated to k points. This makes a geometric interpretation more difficult.

A possible application of the MPS mosaic could be particle management. If a new point should be inserted into a hole in the cloud, the standard management, as described in Section 3.2.4, places it in a point where Voronoi cells meet, which is a solely geometric criterion. Alternatively, new points can be inserted into a point safely inside a large MPS cell. Large MPS cells can be used as an indicator for the need of a new point, and in the middle of a cell the stencil selection procedure is stable. This criterion would not be solely geometric, but be with respect to a MPS-approximation of the Laplace operator. In this sense the particle management could be focused on the approximation of the Laplace operator.

6.11 General Second Order Elliptic Problems

In Section 4.3 we have outlined how the presented finite difference approaches can be applied to general second order elliptic problems (4.28) with the elliptic operator $Lu = A : \nabla^2 u + \mathbf{b} \cdot \nabla u + cu$. The constraints to be satisfied are

$$\sum_{i=1}^{m} \bar{\mathbf{x}}_i \mathfrak{s}_i = \mathbf{b} \tag{6.58}$$

$$\sum_{i=1}^{m} \left(\bar{\mathbf{x}}_i \cdot \bar{\mathbf{x}}_i^T \right) \mathfrak{s}_i = 2A \tag{6.59}$$

i.e. we obtain the same linear system as for the Laplace operator, but a different right hand side $((b_x, b_y, 2a_{11}, 2a_{12}, 2a_{22})^T$ in 2d). The central entry is then obtained as

$$\mathfrak{s}_0 = c - \sum_{i=1}^{m} \mathfrak{s}_i \ . \tag{6.60}$$

One motivation for the M-matrix structure of an approximation of the Poisson equation is that the maximum principle of the problem is reflected in the discretization. For a general second order elliptic problem it is not clear to which extent an M-matrix has the appropriate type of maximum principle. Certainly of interest remain the advantages of the M-matrix structure with respect to linear solvers. Hence we briefly outline for which types of elliptic operators L the derived results on positive stencils can be generalized.

For general constants c, $\mathbf{b}$, A a complete analysis could not be performed. One can say about the different parts:

- If positive stencils can be guaranteed for $c = 0$, then positive stencils are always obtained for $c < 0$.
- The role of the advection $\mathbf{b}$ for positive stencils is not completely clear. Presumably, if $\mathbf{b}$ is uniformly bounded by a sufficiently small constant, then the derived results can be preserved.
- If A is diagonal, and $\mathbf{b} = 0$, the estimates in Theorem 6.8 respectively Theorem 6.10 can be preserved in a modified version. The cone opening angles become the worse the larger the ration between smallest and largest eigenvalue of A. Unclear is the role of mixed derivatives, i.e. the case when A is not diagonal.

Chapter 7

Linear Solvers

Solving a linear system $A \cdot \mathbf{x} = \mathbf{b}$ is a fundamental problem in numerical mathematics. Mostly complex problems often times boil down to the solution of linear systems, which, of course, are typically very large and, in addition, a large amount of them has to be solved in a full computation. The FPM is such an example. In a full computation, many linear problems have to be solved. The system matrices arise from a meshless finite difference discretization, thus they are firstly sparse and secondly non-symmetric. Efficient solvers for linear systems with non-symmetric sparse matrices are required for the FPM.

A large amount of literature is devoted to the numerical solution of linear systems, and a huge variety of solvers exist. Often times one is facing the problem that the more sophisticated and efficient solvers do not apply to any regular matrix, but only to specific subclasses. While Gaußian elimination with pivoting in principle works for any regular matrix (though not necessarily precisely and efficiently), iterative and semi iterative methods can only be guaranteed to converge if the system matrix satisfies additional assumptions, such as an appropriate type of diagonal dominance.

Multigrid solvers additionally require multiple levels of resolution and appropriate transfer from one to another level. Since iterative solvers are one basic ingredient of multigrid solvers, one faces often times even larger formal restrictions on the system matrices.

On the other hand, the conditions on the system matrices are typically only sufficient for the convergence of the solver, i.e. a good numerical solution method can in practice very well converge for many matrices which formally violate the imposed conditions. Such a solver is referred to as *robust*.

Another interesting aspect is preconditioning. Any linear solver can be improved, if it is not applied directly to the system matrix, but to the matrix times an approximation to its inverse. The approximation to the inverse of the system matrix can be obtained by another linear solver, yielding a wide class of possible combinations of linear solvers.

In this section we briefly outline the basis ideas of iterative and Krylov subspace methods for non-symmetric system matrices. Our main interest lies in applying multigrid solvers, since only these are known to be able to achieve an optimal effort $O(n)$, where n is the number of points. We outline the fundamental ideas of multigrid methods and then focus on algebraic multigrid. We present which versions of multigrid qualify for the system arising in the FPM.

7.1 Iterative Solvers

Consider the linear system

$$A \cdot \mathbf{x} = \mathbf{b} \ . \tag{7.1}$$

with $A \in \mathbb{R}^{n \times n}$ regular to be solved. An iterative scheme is defined by a function $\Phi : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$, such that for a starting vector $\mathbf{x}^{(0)}$ a sequence of approximate solutions is generated by $\mathbf{x}^{(m+1)} = \Phi\left(\mathbf{x}^{(m)}, \mathbf{b}\right)$.

Definition 7.1. An iterative scheme is called *consistent* with (7.1) if $\Phi(\mathbf{x}, \mathbf{b}) = \mathbf{x}$ for the solution $\mathbf{x} = A^{-1} \cdot \mathbf{b}$. A scheme is called *convergent* if for all $\mathbf{b} \in \mathbb{R}^n$ one has $\lim_{m \to \infty} \mathbf{x}^{(m)} = \mathbf{x}$.

Linear iteration schemes can we written as

$$\mathbf{x}^{(m+1)} = M \cdot \mathbf{x}^{(m)} + N \cdot \mathbf{b} \ . \tag{7.2}$$

Theorem 7.1. *Iteration (7.2) is consistent, iff $M + NA = I$, where I is the identity matrix. A consistent linear iteration scheme is convergent, iff $\rho(M) < 1$.*

Proof. The proofs are given in [Hac93]. □

One can rewrite (7.2) as

$$\mathbf{x}^{(m+1)} = \mathbf{x}^{(m)} - N\left(A \cdot \mathbf{x}^{(m)} - \mathbf{b}\right) \ . \tag{7.3}$$

Define $P = N^{-1}$. The iteration scheme is the faster, the better P approximates A. One obtains the scheme

$$P \cdot \left(\mathbf{x}^{(m+1)} - \mathbf{x}^{(m)}\right) = \mathbf{b} - A \cdot \mathbf{x}^{(m)} \ . \tag{7.4}$$

The right hand side $\mathbf{r}^{(m)} = \mathbf{b} - A \cdot \mathbf{x}^{(m)}$ is called *residual*. The matrix P which approximates A can be interpreted as a preconditioning matrix, as outlined in Section 7.3. Introducing the update vector $\mathbf{z}(m) = \mathbf{x}^{(m+1)} - \mathbf{x}^{(m)}$, the iteration scheme can be written as

$$P \cdot \mathbf{z}^{(m)} = \mathbf{r}^{(m)} \ . \tag{7.5}$$

If the right hand side is multiplied by ω, one obtains a relaxed scheme.

7.1.1 Examples and Convergence for M-Matrices

Let the system matrix be split

$$A = D - L - U \ , \tag{7.6}$$

where D is a diagonal matrix, L is a strictly lower triangular and U is a strictly upper triangular matrix. The matrices D, L and U are uniquely defined. Assume D is regular. Examples of basic iteration schemes are

- **Richardson**: $P = I$
- **Jacobi**: $P = D$
- **Gauß-Seidel**: $P = D - L$
- **SOR**: $P = \frac{1}{\omega}D - L$

The first three methods can additionally be relaxed by a parameter ω. Various approaches exist on how to split the system matrix in other ways. One example are ILU-schemes [Hac93], for which $P = LU$, where $L\ U$ is an incomplete decomposition.

The listed iteration schemes do not work for all regular matrices A. Additional conditions have to be imposed in order to guarantee convergence. One important sufficient criterion for the convergence of iterative methods is the M-matrix property, as described in Section 6.1. The following theorems hold:

Theorem 7.2. *If A is an M-matrix, then $\rho(I - D^{-1}A) < 1$, i.e. the Jacobi iteration converges.*

Proof. The proof can be found in [Hac93]. □

This statement extends to the Gauß-Seidel iteration by the

Theorem 7.3 (Stein-Rosenberg-Theorem)**.** *The Jacobi iteration matrix J and the Gauß-Seidel iteration matrix G satisfy exactly one of the following relations*

1. $\rho(G) = \rho(J) = 0$
2. $0 < \rho(G) < \rho(J) < 1$
3. $\rho(G) = \rho(J) = 1$
4. $\rho(G) > \rho(J) > 1$

Proof. The proof is given in Varga [Var00]. □

In particular, if the Jacobi iteration converges, then the Gauß-Seidel iteration also converges, and its converge rate will be better. Hence, for matrices obtained by the MPS method, the Gauß-Seidel iteration always converges. Further results about M-matrices in relation to iterative linear solvers can be found in the books of Axelsson [Axe94] and Varga [Var00].

Note that the presented iterative methods in general are no efficient stand alone solvers for discretizations of elliptic partial differential equations. However, if used in a multigrid method, they can be very efficient, as outlined in Section 7.4.

7.2 Krylov Subspace Methods

Krylov subspace methods are a general approach which is tailored for the case of a black box linear operator A, which can be applied (at some given cost) to a vector, but its particular entries cannot (or should not) directly be accessed. We outline some of the approaches. More information is provided in the book of Quarteroni, Sacco and Saleri [QSS00].

Consider a linear system $A \cdot \mathbf{x} = \mathbf{b}$ to be solved. To a vector $\mathbf{w}$ one defines the *Krylov subspace* of order m as

$$K_m(A, \mathbf{w}) = \text{span}\{\mathbf{w}, A \cdot \mathbf{w}, \ldots, A^{m-1} \cdot \mathbf{w}\} \ . \tag{7.7}$$

Any vector $\mathbf{v} \in K_m(A, \mathbf{w})$ can be written as $p(A) \cdot \mathbf{w}$, where p is a polynomial of degree $\leq m - 1$.

Let $\mathbf{x}^{(0)}$ be a starting vector for an iteration, and $\mathbf{r}^{(0)} = \mathbf{b} - A \cdot \mathbf{x}^{(0)}$ the corresponding initial residual. The iterates of the Richardson method satisfy $\mathbf{x}^{(m)} \in W_m$, where

$$W_m = \mathbf{x}^{(0)} + K_m(A, \mathbf{r}^{(0)}) \ . \tag{7.8}$$

Krylov methods generalize this approach by considering the class of iterations whose iterates satisfy $\mathbf{x}^{(m)} \in W_m$, i.e. $\mathbf{x}^{(m)} = \mathbf{x}^{(0)} + p_{m-1}(A)\mathbf{r}^{(0)}$. The polynomial p_{m-1} is chosen in such a way that the iterates are close to the correct solution $\mathbf{x}$. Since the correct solution is not known, one provides criteria which make it likely to well approach the correct solution with increasing m. Two possible approaches are:

- **GMRES** (*generalized minimum residual*): Choose $\mathbf{x}^{(m)}$, such that the Euclidian norm of the residual $\|\mathbf{r}^{(k)}\|_2$ is minimized.
- **FOM** (*full orthogonalization method*): Choose $\mathbf{x}^{(m)}$, such that the residual is orthogonal to the Krylov subspace, i.e. $\mathbf{v}^T \cdot \mathbf{r}^{(k)} = 0 \ \forall \mathbf{v} \in K_m(A, \mathbf{r}^{(0)})$.

Using the Gram-Schmidt procedure one can construct an orthonormal basis for the Krylov subspaces. Using this approach, the FOM leads to the *Arnoldi method*. This method formally computes the correct solution after at most n iteration steps. For practical purposes, however, storing the whole Krylov subspace basis is only reasonable for small values of m. The same applies for the GMRES method and for most other Krylov methods: In their basic form as direct methods they are not useful, but with appropriate modification they become efficient iterative solvers, which yield good approximate solutions after less than n iteration steps.

For symmetric matrices, the Arnoldi method becomes the *Lanczos method*, which due to the symmetry is less elaborate in performing the orthonormalization procedure. The Lanczos method can again be extended to the case of non-symmetric matrices, but preserving the smaller effort. The orthogonalization is replaced by a bi-orthogonalization, i.e. two sequences of bases, spanning two different Krylov subspaces, which are bi-orthogonal, are constructed. This approach leads to the *BiCGstab* method, which can be seen as a stabilized version of a bi-conjugate gradient method.

In Chapter 8 the BiCGstab method is used in the FPM for solving the arising system. On the one hand, it is used as a solution method, augmented by a least squares preconditioner. On the other hand it is used as an outer iteration for an algebraic multigrid preconditioner.

7.3 Preconditioning

The linear system $A \cdot \mathbf{x} = \mathbf{b}$ can be difficult to solve, if its condition number $\kappa = \|A\| \cdot \|A^{-1}\|$ is large. This motivates the concept of preconditioning. Let P be an approximation of A, which is easy to construct and to solve for. Then one can equivalently solve the preconditioned system

$$P^{-1}A \cdot \mathbf{x} = P^{-1} \cdot \mathbf{b} \ . \tag{7.9}$$

Here P is denoted *left preconditioner*. Alternatively, one can consider to solve

$$AP^{-1} \cdot \mathbf{y} = \mathbf{b} \ , \quad P \cdot \mathbf{x} = \mathbf{y} \ . \tag{7.10}$$

Here P is denoted *right preconditioner*. Combinations of left and right preconditioners are possible.

One can easily check that the iteration scheme (7.3) with iteration matrix $N = P^{-1}$ is equivalent to solving the left preconditioned system (7.9) by a simple Richardson iteration

$$\mathbf{x}^{(m+1)} = \mathbf{x}^{(m)} - \left(P^{-1}A \cdot \mathbf{x}^{(m)} - P^{-1}\mathbf{b}\right) \ . \tag{7.11}$$

Preconditioners must not be understood merely as matrices, but as operational objects. Any black box function which approximately solves the linear system $A \cdot \mathbf{x} = \mathbf{b}$ can work

as a preconditioner, if applied to the residual, employing relation (7.5). Generally, two approximate solvers can be combined into a single method. One solver is used as an outer iteration, while the other solver is used as a preconditioner. If set up appropriately, one can benefit from the robustness of the outer iterative solver and the efficiency of the solver used as preconditioner. An example of such a construction is to use the BiCGstab method as an outer iteration and AMG as a preconditioner.

7.4 Multigrid

In discretizations of partial differential equations one is confronted with sparse linear systems to be solved. Typically, the number of nonzero entries is proportional to the number of unknowns, i.e. the number of matrix entries in each row is independent of the problem size n. Consequently, applying the sparse matrix to a vector is of complexity $O(n)$. Iterative solution methods, as presented in the previous sections, apply the matrix multiple times, until a satisfying approximation to the solution is found. If for an iterative method the number of required iteration steps was independent of n, the method would be of optimal complexity $O(n)$. Unfortunately, no such iterative method is known. Multigrid methods use iterative solvers on a whole collection of levels, and thus are able to achieve the desired optimal complexity. The first multigrid method was presented by Federenko [Fed61]. Brandt [Bra73] and Hackbusch [Hac76] extended Federenko's approach to efficient methods. Introductions to classical multigrid methods are given by Hackbusch [Hac85] and Wesseling [Wes92]. A detailed overview over the state of the art in multigrid is provided by Trottenberg, Oosterlee and Schüller [TOS01].

7.4.1 Basic Multigrid

The ingredients of a multigrid method are an iterative solver, a hierarchy of grids, operators which allow to go up and down in the grid hierarchy, and possibly a discretization of the governing equation on each level. Multigrid methods are based on the observation that iterative solvers, such as the Jacobi or the Gauß-Seidel method reduce high frequency error components very fast, while low frequency error components are reduced very slowly. When dealing with approximation of the Poisson equation, iterative solvers can be interpreted as discretizations of the corresponding time dependent problem, in this case the inhomogeneous heat equation. The heat equation lets a Fourier eigenfunction to the eigenvalue λ decay by the factor $\exp(-k\lambda^2)$, i.e. high frequencies are canceled after a few steps. Hence, the iterative solver is called *smoother* in the context of multigrid methods. Among the presented schemes, the Gauß-Seidel method generally works best.

Now another solver is desired which reduces the low frequency error components. Such a solver does not exist on the fine grid, but since the error is smooth, one can without problems restrict the approximate solution to a coarser level. Here the remaining error can be annihilated, and the result can be prolongated to the fine grid again.

Twogrid

Assume to have a fine level grid or point cloud, as well as a coarse level grid or point cloud. The variables are from $\mathbb{R}^{n_1}$ respectively $\mathbb{R}^{n_2}$. Let a prolongation operator $P : \mathbb{R}^{n_2} \to \mathbb{R}^{n_1}$ exist, which interpolates a function from the coarse level to the fine level. Accordingly, one has a restriction operator $R : \mathbb{R}^{n_1} \to \mathbb{R}^{n_2}$, mapping from the fine to the coarse level. The linear system on the fine level is $A_1 \cdot \mathbf{x}_1 = \mathbf{b}_1$. The system on the coarse level can either be constructed by discretizing the governing equation, or by a Galerkin approach

$$A_2 = RA_1P \tag{7.12}$$

$$\mathbf{b}_2 = R\mathbf{b}_1 \tag{7.13}$$

Often times it is desirable to have $R = P^T$.

Assume to have a starting guess $\mathbf{x}_1^{(0)}$. The *twogrid method* is given by the following procedure.

1. Iterate ν_1 steps on fine grid: $\mathbf{x}_1^{(m+1)} = \mathbf{x}_1^{(m)} + P^{-1} \cdot \mathbf{r}_1^{(m)}$
2. Restrict residual: $\mathbf{r}_2 = R\mathbf{r}_1^{(\nu_1)}$
3. Solve for error on coarse level: $\mathbf{e}_2 = A_2^{-1} \cdot \mathbf{r}_2$
4. Prolongate error to fine level: $\mathbf{e}_1 = P\mathbf{e}_2$
5. Correct fine grid solution: $\tilde{\mathbf{x}}_1^{(\nu_1)} = \mathbf{x}_1^{(\nu_1)} - \mathbf{e}_1$
6. Start with $\tilde{\mathbf{x}}_1^{(\nu_1)}$ and iterate ν_2 steps on fine grid: $\mathbf{x}_1^{(m+1)} = \mathbf{x}_1^{(m)} + P^{-1} \cdot \mathbf{r}_1^{(m)}$

The twogrid method assumes that on the coarse level the system A_2 is small enough to be solved directly and accurately. For large problems, this is in general not the case.

Multigrid

Multigrid uses the twogrid method recursively. The solution of the system on the coarse level is again done by a new twogrid method. This uses a third level, which is coarser than the second level. This procedure is done recursively until a coarsest level is reached at which the system is solved by a direct method. This approach starts at the finest level, goes through

the hierarchy of levels down to the coarsest level, solves exactly there, and goes again up the hierarchy to the finest level. Due to this behavior, this approach is called *V-cycle*.

The descent to the next lower level in the twogrid method can also be done twice, i.e. two coarse grid corrections can be computed. In this case the method would cycle up and down through the levels. In the case of three levels, the cycling would look like a W, hence this approach is called *W-cycle*.

The number of coarse grid corrections can also depend on the level and the cycling history. One example is the *F-cycle*. The V- and W-cycles are shown at [TOS01, p. 46], the F-cycle is sketched at [TOS01, p. 49].

In this basic version, each iteration on a new level has no information available about a good starting value. This can be provided, using information from a coarser level as an initial guess. This approach is referred to as *full multigrid* (FMG).

As presented at [TOS01, p. 59], the number of operations required for FMG is $O(n)$, where n is the number of unknowns. The reason is that the multigrid convergence factor is independent on the mesh size h.

7.4.2 Multigrid Versions

A multigrid method requires many ingredients. Multigrid versions differ in which of these ingredients are provided by the background of the method, i.e. by information which is not contained in the finest grid system $A \cdot \mathbf{x} = \mathbf{b}$, and ingredients which are constructed by the method. The most flexible approach is to require all levels, all prolongation and restriction operators and all level dependent systems to be provided by the "user". Obviously, this is also the most complicated approach, since by far not all of these ingredients can be arbitrarily combined. On the other hand, the most convenient approach would be a black box solver, i.e. the "user" enters nothing but the fine grid system, and everything else is constructed by the method. Of course, this approach is also the least flexible. The various multigrid approaches are distributed into two classes:

- **Geometric Multigrid** (GMG):
  The hierarchy of grids is constructed from the geometrical background of the problem. Also suitable prolongation operators are constructed using the geometric information. The corresponding restriction operators, as well as the level dependent systems can either be obtained by a Galerkin approach, or they can also be provided in a consistent manner. The GMG approach provides a large flexibility. However, in the case of meshless point clouds, it provides far more flexibility than desired. One can use the particle management methods, as described in Section 2.5, to construct a coarse point

cloud from a fine point cloud. However, by this approach multiple levels of point clouds can only be constructed for extremely friendly geometries. Also, the choice of correct prolongation operators is difficult. Additionally, it might be of interest to employ matrix dependent coarsening strategies. These are done automatically by algebraic multigrid approaches.

- **Algebraic Multigrid** (AMG):
  The method takes the fine scale linear system as input and constructs the corresponding matrix graph. This graph plays the role of a grid in geometric approaches. A hierarchy of graphs is constructed. Here, AMG methods use information from the system matrix entries for the coarsening. Linear systems on coarser levels are constructed by a Galerkin approach. Compared to geometric multigrid, the algebraic approach has an initial overhead in setting up the matrix graph and selecting the coarse level variables. We will comment on AMG methods in more detail in the following section.

For the meshless finite difference problems in our context, we decide to apply algebraic multigrid. Since in the meshless setup many of the information required for a geometric approach is not directly available, the AMG comes as a natural choice. Additionally, AMG is in some sense the meshless version of multigrid, i.e. it fits into the context. Of course, there is the drawback of the additional cost in the initialization process as well as less control and less flexibility.

7.4.3 Algebraic Multigrid

Algebraic Multigrid methods were introduced by Stüben [Stü83, RS87] as well as Brandt, McCormick and Ruge [BMR84]. A review [Stü01] and an overview [TOS01, App. A] over the state of the art of AMG is given by Stüben.

AMG methods take the system matrix A and the right hand side vector $\mathbf{b}$ as input. A smoother is chosen, for instance the Gauß-Seidel method. The graph of the system matrix is set up. Different levels are constructed by coarsening the graph (see below). Systems on coarser levels are constructed by the Galerkin approach. The smoother is applied to the systems on various levels. Note that for this general approach, the coarsening need not necessarily make sense geometrically. Consequently, also the smoothing need not be a geometric smoothing. In fact, in [TOS01, App. A] Stüben provides an example in which the AMG iteration actually makes the error geometrically non-smooth. One introduces the concept of algebraic smoothness, simply meaning that the error does not significantly decay with a given smoother. Once this stage is achieved, one goes to the next coarser level.

AMG methods fix a smoother and adjust the coarsening strategy, i.e. how to split the unknowns in fine-scale and coarse-scale variables. Various versions of coarsening exist. We

consider two examples:

- **Default coarsening**:
  Given the list of unknowns, the first variables are chosen as coarse scale variables, the other variables are chosen as fine scale variables. At first glance this appears to be a very inefficient coarsening strategy, but in the case of point clouds with a random numbering it actually yields good results, as presented in Section 8.3.
- **Ruge-Stüben coarsening** [RS87]:
  If the system matrix has M-matrix structure, the absolute value of off-diagonal entries is interpreted as a *coupling strength* between the corresponding two unknowns. Coarsening of the graph is done preferably in the directions of large couplings. This approach is particularly advantageous for strongly anisotropic problems, since the coarsening is done matrix dependent.

Besides these basic ingredients, many extensions exist on how to make the AMG method more efficient and robust. In Section 8.2 we consider the *SAMG* black box solver developed in the *Fraunhofer Institut für Algorithmen und Wissenschaftliches Rechnen*. Most of the features described in [Stü01] and [TOS01, App. A] are available in this solver. Of particular interest, especially for the MPS matrices, is the aspect of *aggressive coarsening*. If the stencils at the finest level are small (as it is the case with the MPS method), standard coarsening may yield matrices with many nonzero entries at the next coarser level (see [TOS01, pp. 476]). This effect can be prevented by enforcing aggressive coarsening, which considers long-range couplings.

7.4.4 Convergence Results for AMG

In [TOS01, App. A] Stüben provides convergence results of the AMG method for the case of symmetric M-matrices. Symmetric M-matrices are positive definite, and this property is fundamental to go from a twogrid to a multigrid estimate. Most proofs and estimates presented in the above reference are only valid for positive definite matrices. In our context, the M-matrices are non-symmetric, thus one cannot conclude positive definiteness from the M-matrix property. There is also no straightforward way to ensure positive definiteness by other means. Of course, positive definiteness is only a sufficient criterion, and generally AMG is very robust and often times turns out to work for many matrices which are not positive definite. Indeed, the SAMG method investigated in Section 8.2 performs well for all considered discretization approaches (MPS and least squares). However, there is no general guarantee of success.

An analysis of algebraic twogrid approaches for non-symmetric M-matrices is provided by Mense and Nabben [MN06]. Here the M-matrix property is used directly, without any

symmetry requirement. Unfortunately, extending the theoretical results to multigrid is not easily possible, as it is when positive definiteness is employed. The considered method is the *algebraic multilevel iteration* (AMLI) method, as presented by Axelsson and Vassilevski [AV89, AV90].

Consider the unknowns be split into coarse (C) and fine (F) nodes. The system matrix becomes in block form

$$A = \begin{bmatrix} A_{FF} & A_{FC} \\ A_{CF} & A_{CC} \end{bmatrix} . \tag{7.14}$$

If A_{FF} is regular then A can be written as

$$A = \begin{bmatrix} I & 0 \\ A_{CF}A_{FF}^{-1} & I \end{bmatrix} \cdot \begin{bmatrix} A_{FF} & 0 \\ 0 & S \end{bmatrix} \cdot \begin{bmatrix} I & A_{FF}A_{FC}^{-1} \\ 0 & I \end{bmatrix} , \tag{7.15}$$

with the Schur complement

$$S = A_{CC} - A_{CF}A_{FF}^{-1}A_{FC} . \tag{7.16}$$

Assume that A_{FF} is approximated by $\tilde{A}_{FF}$ and S by $\tilde{S}$. Then

$$\tilde{A} = \begin{bmatrix} I & 0 \\ A_{CF}\tilde{A}_{FF}^{-1} & I \end{bmatrix} \cdot \begin{bmatrix} \tilde{A}_{FF} & 0 \\ 0 & \tilde{S} \end{bmatrix} \cdot \begin{bmatrix} I & \tilde{A}_{FF}A_{FC}^{-1} \\ 0 & I \end{bmatrix} \tag{7.17}$$

is an approximation to A, and can thus be used as a preconditioner for an iteration, as described in Section 7.3. The iteration matrix is

$$M_{AMLI} = \tilde{A}^{-1} \cdot A . \tag{7.18}$$

Mense and Nabben show that the iteration matrix M_{AMLI} can be written as an additive Schwarz method

$$M_{AMLI} = I - P_1 - P_2 , \tag{7.19}$$

where

$$P_1 = \tilde{P}^T\tilde{S}^{-1}\tilde{R}A \tag{7.20}$$

$$P_2 = \hat{R}^T\tilde{A}_{FF}^{-1}\hat{R}A \tag{7.21}$$

Here the restriction and prolongation operators are

$$\tilde{R} = \begin{bmatrix} -A_{CF}\tilde{A}_{CF}^{-1} & I \end{bmatrix} \tag{7.22}$$

$$\tilde{P} = \begin{bmatrix} -\tilde{A}_{FF}^{-1}A_{FC} & I \end{bmatrix} \tag{7.23}$$

$$\hat{R} = \begin{bmatrix} I & 0 \end{bmatrix} \tag{7.24}$$

The additive AMLI version motivates to consider multiplicative variants. Altogether, four twogrid versions are considered:

- **AMLI**: $M = I - P_1 - P_2$
- **Multiplicative AMLI** (MAMLI): $M = (I - P_1)(I - P_2)$
- **Reverse MAMLI** (RMAMLI): $M = (I - P_2)(I - P_1)$
- **Symmetrized MAMLI** (SMAMLI): $M = (I - P_2)(I - P_1)(I - P_2)$

If A is an M-matrix, Mense and Nabben prove the convergence of these four methods, i.e. $\rho(M) < 1$, assumed the twogrid approximations $\tilde{A}_{FF}$ and $\tilde{S}$ satisfy some weak regularity. The approximations must satisfy

$$\tilde{A}_{FF}^{-1} \geq 0 \tag{7.25}$$
$$I - \tilde{A}_{FF}^{-1} A_{FF} \geq 0 \tag{7.26}$$
$$\tilde{S}^{-1} \geq 0 \tag{7.27}$$
$$I - \tilde{S}^{-1} \left(A_{CC} - A_{CF} \tilde{A}_{FF}^{-1} A_{FC} \right) \geq 0 \tag{7.28}$$

As in Section 6.1 the inequality is a componentwise statement. In [MN06] it is pointed out that these assumptions are satisfied for most popular splitting approaches.

In the context of meshless methods for the Poisson equation, the AMLI approach by Mense and Nabben guarantees convergence for matrices generated by the MPS method. The twogrid method might also converge for matrices obtained by least squares approaches, but there is no guarantee that it does. Indeed, the numerical experiments presented in Section 8.3 show that for a discretization approach with very few neighboring points considered none of the AMLI variants converges.

7.4.5 FPM and Multigrid for Small Time Steps

When considering the Poisson problems for the pressure correction in the solution of the incompressible Navier-Stokes equations, it can be advantageous to use information from the previous time step. For instance, the starting value of an iterative procedure for solving the linear system can be taken from the previous time step. As pointed out by Griebel, Dornseifer and Neunhoeffer [GDN98], multigrid methods might not be necessary in this case, since the error may be smooth already.

This is true if small time steps are taken. In the FPM, however, an implicit formulation for the viscosity admits to take comparably large time steps. This does not only change the values of the right hand side of the pressure correction Poisson problem. Also particle management will in general change the number and the ordering of points, such that a correspondence to the Poisson solution from the previous time step requires interpolation.

If one wants to use information from the previous time step, it is not guaranteed that the error will be algebraically smooth, when an AMG solver is used.

From our experience it has turned out easier to start with zero as initial guess (or as a crude approximation the solution from the previous step neglecting any particle management), for the Poisson solves and use a sophisticated AMG solver. Since the complexity is optimal, the effort for the AMG iteration is not too large. An approach which we have not investigated is to use information from the previous time step for the coarsening strategy. Here the removal and insertion of points due to particle management should not spoil the outcome too much. This way the initial setup phase of AMG could be significantly accelerated.

Chapter 8

Numerical Experiments

In this chapter we use numerical experiments to investigate the applicability, stability and computational performance of various finite difference discretization approaches. Considering a class of two dimensional Poisson test problems we compare the matrices generated by the MPS approach with matrices generated by least squares approaches, in dependence on the number of points. A Poisson problem in a more complex three dimensional geometry is considered in order to compare the MPS matrices with ALLS matrices with respect to two linear solvers, an iterative solver and a sophisticated multigrid solver. The latter can only be proven to converge for symmetric M-matrices matrices. In our applications, the matrices are non-symmetric, hence we consider various twogrid solvers which can be shown to converge for non-symmetric M-matrices. Finally, we consider the performance of the MPS method in time dependent simulations of problems arising in industrial applications.

8.1 Test Problem in 2d – Compare MPS and LSQ

We compare the MPS method, as motivated and derived in Chapter 6, to the ILLS method, as described in Section 5.1.2. The latter holds as a typical example of least squares methods. Most observations transfer to other least squares methods.

We consider a Poisson problem in a simple two dimensional geometry: On the unit sphere Dirichlet boundary points are placed. Inside the unit ball a point cloud is constructed. For various numbers of interior points n_i point clouds are set up with the following features

- The number of boundary points is $n_b = \pi\sqrt{n_i}$. These are set up equidistantly along the boundary, which yields a nicely dense distribution along the boundary.
- The mesh size must satisfy $h = \frac{2}{\sqrt{n_i}}$.

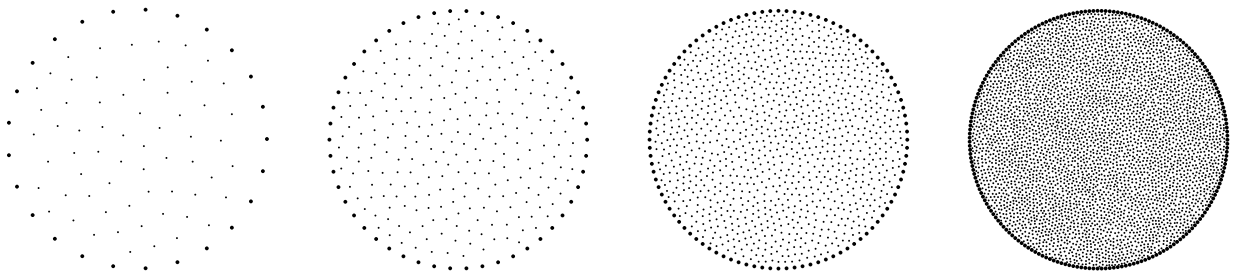

Figure 8.1: Point clouds with 63, 250, 1000 and 4000 interior points in circular domain

- The separation must satisfy $\delta < \frac{1}{8}$.

The parameters h and δ are as defined in Section 3.1. Particles are inserted until the point cloud satisfies the mesh size constraint. Particles are merged until the point cloud satisfies the separation constraint. Note that these values are rather restrictive and yield comparably uniform point clouds. Four example point clouds with 63, 250, 1000 and 4000 interior points are shown in Figure 8.1.

Of course, there are arbitrarily many point clouds with a given number of points which satisfy the above conditions. Therefore a strict analysis as in the case of regular grids is not easily possible here. A general statement for a whole class of matrices can be made theoretically by employing specific conditions. The cone criteria, as presented in Section 6.4, are such examples.

Obtaining rigorous results from numerical experiments is a more difficult task in this context. For a given number of points and conditions, it is certainly not sufficient to consider one single point cloud. A sample of clouds, which all satisfy the conditions, should be considered. Errors can be averaged over the sample. A sufficiently large sample can yield information about worst case situations. Note however, that the sample will in general be restricted by the strategy how the point clouds are constructed. In our case we start with an random point cloud and then apply particle management until the prescribed conditions are satisfied (the management is slightly modified here in order to end up with the desired number of points). Certainly, there are point clouds satisfying the conditions which cannot be generated by our construction strategy. Even more, a good and smart particle management will generally generate point clouds which are much nicer than one could conclude from the conditions alone. This is a good thing for an implementation of the FPM. For a numerical analysis one might not consider problematic cases in the sample. On the other hand, why should we bother with point clouds which do not arise in our particle method anyhow? We therefore change the definition of our sample of point clouds to the class of "point clouds with a prescribed number of points and conditions, which can arise in the FPM".

We consider various numbers of interior points ($n_i \in \{63, 125, 250, 500, 1000, 4000, 6000\}$).

For each such number we generate 20 point clouds, starting from a cloud of independently uniformly distributed points, then applying particle management. For each point cloud we generate discretizations of the Poisson equation with nontrivial nonlinear functions for f and g. We use the following four discretization approaches:

- **MPS**: Generate an M-matrix by the MPS method, as described in Section 6.8.
- **CS**: A Poisson discretization matrix for which for each interior point the closest 5 neighbors are incorporated into the stencil (hence *closest stencil* method). The Vandermonde system for each interior point is solved directly. This approach can be interpreted as a least squares approach with a minimum neighborhood radius.
- **ILLS 1**: A Poisson discretization matrix obtained by the ILLS method for a circular neighborhood with $r = \frac{4}{\sqrt{n_i}}$.
- **ILLS 2**: A Poisson discretization matrix obtained by the ILLS method for a circular neighborhood with $r = \frac{6}{\sqrt{n_i}}$.

For each such system we can consider various aspects (such as errors, number of nonzero entries, etc.). These values are averaged (or the minimum or maximum is considered) over the 20 sample experiments.

8.1.1 Matrix Structure

One aspect of interest is the sparse matrix structure. For the plots we consider one specific example point cloud only. Figure 8.2 shows the structure of the MPS matrices, Figure 8.3 shows the structure of the 5 closest point stencil discretization, Figure 8.4 and Figure 8.5 show the ILLS matrices with the above given circular radii. The matrix ILLS 1 has in average about 14 neighbors, the matrix ILLS 2 has in average about 35 neighbors. The latter choice yields unnecessarily many stencil entries. The matrix will be expensive to apply. In the comparison it holds a counterpart to the minimal stencil methods.

In all figures, a small dot denotes a nonzero entry in the system matrix. A fat dot denotes an off-diagonal entry of positive sign, i.e. the resulting row is not a positive stencil. If at least one fat dot is visible, the matrix is no L-matrix and thus no M-matrix (it can however, still be inverse positive, see below). As predicted by our theory, the plots shown in Figure 8.2 for the MPS method are all of the correct sign. The matrices are M-matrices. Note that the matrix is not reciprocal, the graph is not symmetric.

The same applies for the closest stencil matrix shown in Figure 8.3. The matrix is not reciprocal. Additionally, there is a significant amount of positive off-diagonal entries, which

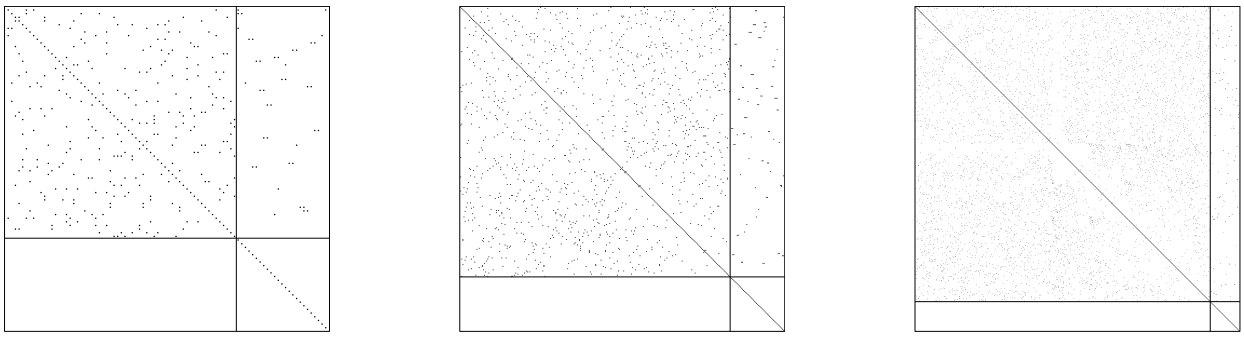

Figure 8.2: Sparse matrix structure of MPS matrix for 63, 250 and 1000 points

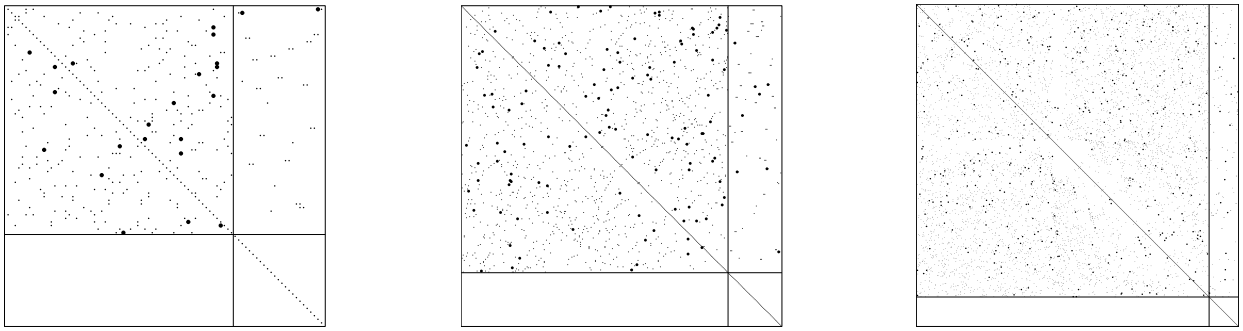

Figure 8.3: Sparse matrix structure of CS matrix for 63, 250 and 1000 points

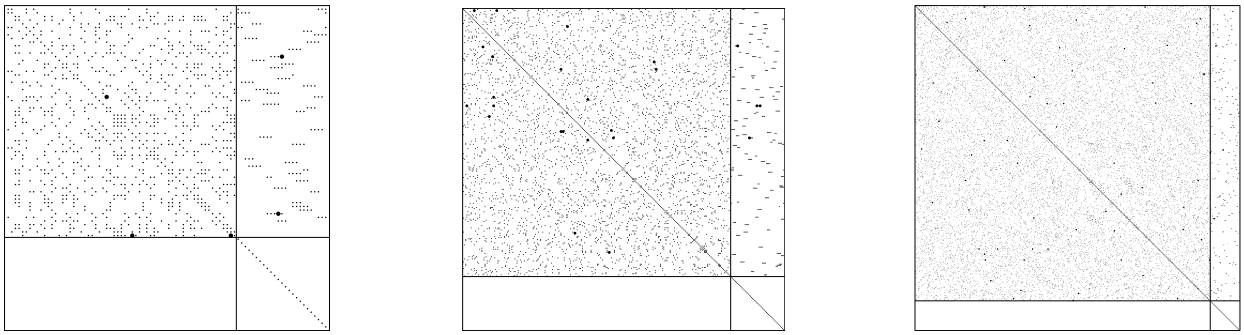

Figure 8.4: Sparse matrix structure of matrix ILLS 1 for 63, 250 and 1000 points

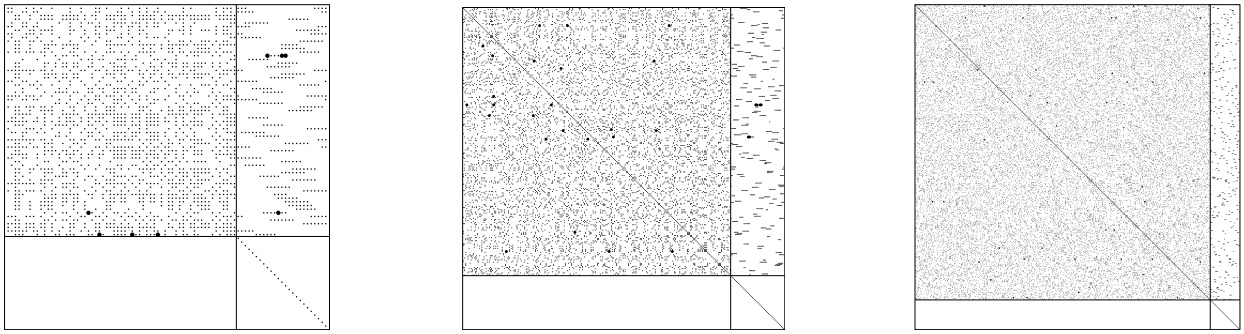

Figure 8.5: Sparse matrix structure of matrix ILLS 2 for 63, 250 and 1000 points

is a bad thing. We will see below that the corresponding approximation errors are larger than with the other approaches.

The ILLS approaches shown in Figure 8.4 and Figure 8.5 also show positive off-diagonal entries. Not as many as for the CS matrix, but in principle one single positive off-diagonal entry can spoil the inverse positivity of A. A good property of these two matrices is their reciprocity. The matrix graphs are symmetric. Note that the matrices themselves are not symmetric.

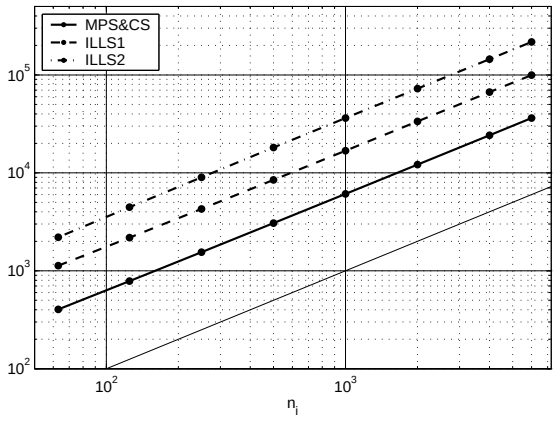

Figure 8.6: Number of nonzero entries in system matrix A

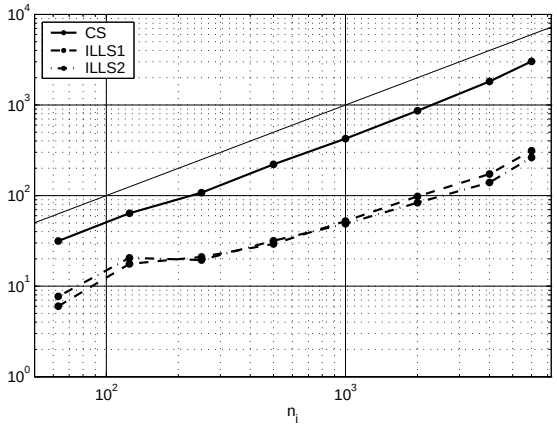

Figure 8.7: Number of positive off-diagonal entries in A

In Figure 8.6 the number of nonzero entries of the discretization matrices are plotted in dependence on the number of interior points. The solid line stands for the MPS and the CS matrices, which both have the minimum possible number of entries. The dashed line represents the ILLS 1, and the dash-dotted line the ILLS 2 matrix. These are less sparse, but the number of nonzero entries is linear in the number of points. Note here and in the following double log plots the control line with slope 1.

Figure 8.7 shows the number of positive off-diagonal entries. These are always zero for the MPS matrices. The CS matrix is shown by the solid line. The number of wrong signed matrix entries grows linearly, but it is significant. For the ILLS 1 and ILLS 2 matrices, denoted by the dashed and dash-dotted lines, the number of positive off-diagonal entries is almost equal. This is interesting, since the ILLS 2 matrix has significantly nonzero entries, which in principle could contribute to a wrong signed entry.

8.1.2 Inverse Matrix Structure

If the system matrix A is an M-matrix, we know that it is inverse positive, which yields the discrete maximum principle given in Theorem 6.5. The MPS matrices are M-matrices. The

CS matrix and the matrices arising from ILLS approximations are not. On the other hand, the matrices ILLS 1 and ILLS 2 have comparably few positive off-diagonal entries. These are relevant in absolute value. Still, the system matrices can be inverse positive, since the class of inverse positive matrices is much larger than the class of M-matrices (see Example 6.1).

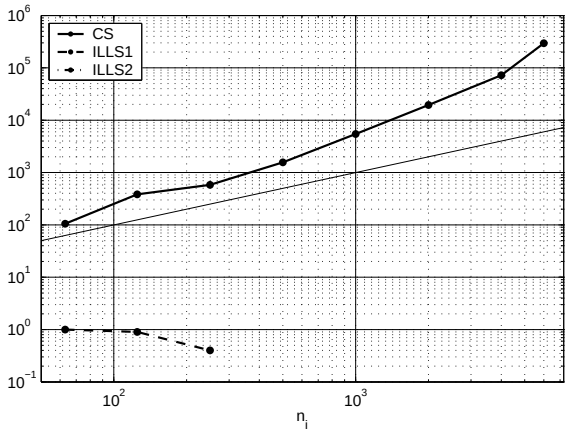

Figure 8.8: Number of negative entries in A^{-1}

n_i	MPS	CS	ILLS1	ILLS2
63	0	105.0	1.0	0.0
125	0	382.1	0.9	0.0
250	0	581.1	0.5	0.0
500	0	1563.1	0.5	0.0
1000	0	5443.6	0.0	0.0
2000	0	19439.3	0.0	0.0
4000	0	72364.9	0.0	0.0
6000	0	295538.7	0.0	0.0

Table 8.1: Number of negative entries in A^{-1}

Since the considered problems are comparably small, we can compute the inverse matrices explicitly. In Table 8.1 the average numbers of negative entries in the inverse matrix are listed. By construction, the MPS matrices have no such entries. The CS matrix has a tremendous amount of negative entries. Figure 8.8 shows the number of these entries in dependence on the number of interior points (solid line). The number of negative entries in A^{-1} grows superlinearly (compare with reference line), since the inverse matrices are full.

Mostly remarkable is the fact that the ILLS 1 approach yields inverse matrices which have almost no negative entries. Even better, all considered matrices obtained by the ILLS 2 approach are inverse positive. These observations give rise to the following

Assertion 8.1. *Under appropriate conditions on the geometry and for a sufficiently large circular neighborhood radius, the ILLS discretization approach yields inverse positive system matrices, even though they are no L-matrices.*

To our knowledge a statement into this direction has not been proven yet. Finding appropriate conditions to obtain such a statement is an interesting step to continue the work on this subject.

Assumed that conditions can be provided such that least squares methods can be guaranteed to yield inverse positive system matrices, the MPS method would lose one of its unique features, as long as only inverse positivity and no L-matrix structure is the property to be achieved. On the other hand, the other strong point of the MPS, the minimal stencil

property, remains. "Positive stencil least squares methods" would be mostly interesting in the context of hybrid methods, as described in Section 6.10.1.

A similar remarkable aspect regards Neumann boundary points. As shown in Section 6.5, Neumann boundary points can also be included to obtain an M-matrix structure. However, due to Theorem 6.12, a higher order approximation of Neumann boundary points is incompatible with an L-matrix structure. In various example computations we have included second order accurate Neumann approximations, and the resulting system matrices turned always out to be inverse positive (in the cases of practical relevance). Example 6.1 is such a case.

8.1.3 Approximation Error

For each point clouds we consider four different approximations of a given Poisson problem. We solve the arising linear systems up to machine accuracy. When considering the difference of the obtained numerical solution to the correct solution we can numerically analyze the global approximation error. Obviously, a good discretization method should not only be solvable fast, but it must yield accurate results. Of interest is the dependence of the error on the number of points. Since all our approximations are based on Taylor expansion up to quadratic order, we can expect the methods to be at least first order accurate, given the method is stable. Unless the point clouds possesses spacial symmetries, we cannot expect higher order accuracy with our methods. With two methods of the same order, the error constant is of importance. We can compare the methods with respect to their error constants. Note that for first order methods, a twice as large error constant means that twice as many points are required to recover the original accuracy.

The difference between true and numerical solution is

$$\mathbf{e} = \mathbf{u} - A^{-1} \cdot \mathbf{f} \tag{8.1}$$

For each discretization matrix we measure the error in the maximum norm as well as the scaled 2-norm

$$||\mathbf{e}||_\infty = \max_{i=1}^{n} |e_i| \tag{8.2}$$

$$||\mathbf{e}||_2 = \sqrt{\frac{1}{n} \sum_{i=1}^{n} |e_i|^2} \tag{8.3}$$

In the latter the normalization is necessary in order to be able to compare the errors for different numbers of points.

The numerical results are shown in Figures 8.9 to 8.12. In each figure, the solid line shows the maximum norm and the dashed line the scaled 2-norm, averaged over the sample of 20

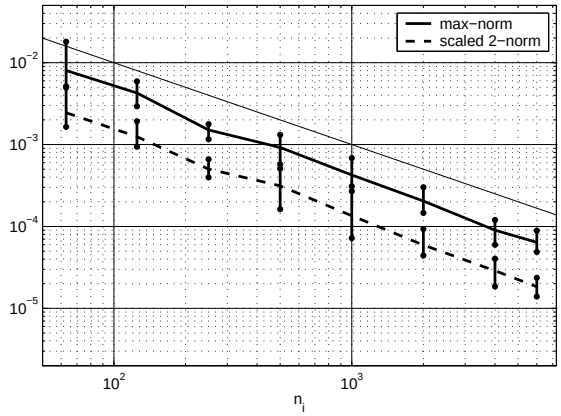

Figure 8.9: Error with MPS approach

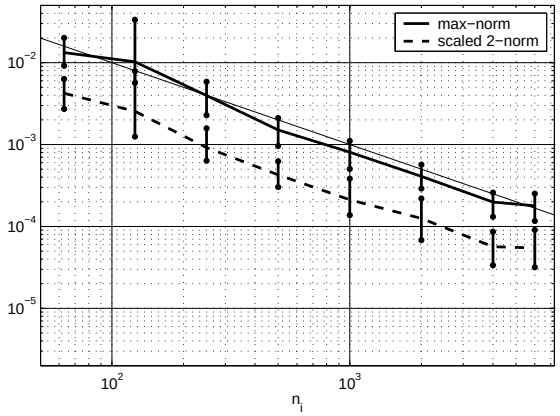

Figure 8.10: Error with CS approach

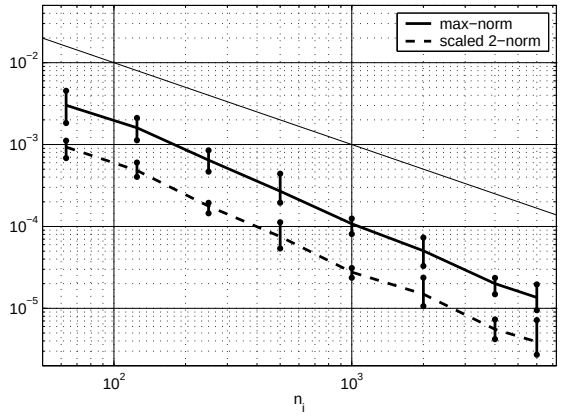

Figure 8.11: Error with ILLS 1 approach

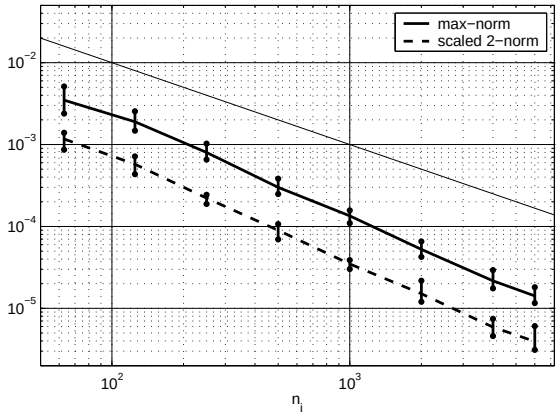

Figure 8.12: Error with ILLS 2 approach

experiments for each number of points. The vertical lines reach from the smallest result to the largest result in each sample. Also shown in the solid reference line of slope -1. Figure 8.9 shows the LM approach, Figure 8.10 shows the CS approach, and Figure 8.11 and 8.12 show the ILLS approaches with different circular radii.

One can observe that indeed all approximations are consistent and first order accurate. Precisely speaking, the CS approach shows multiple bad samples. The ILLS approaches show a slightly superlinear accuracy, presumably due to the fact that the generated point clouds become particularly nice (close to symmetric) for larger numbers of points.

Although all methods show (nearly) first order accuracy, the error constants do differ significantly. The ILLS approximations yield rather good approximations, the error constant is small. Note that the nearly identical errors are obtained for both choices for circular neighborhood radii. The approximation using more neighboring points yields actually slightly larger errors. In comparison, the CS approximation performs significantly worse. The obtained errors are by a factor 10 larger. Thus, in order to achieve the same accuracy as the

ILLS 1 approach, the CS approach would require 10 times as many points, which would more than outweigh any gain of computation time by smaller stencils.

The MPS approximation error is on the one hand significantly smaller than the error obtained by the CS, the method which yields the same number of nonzero entries in the system matrices. On the other hand, the MPS errors are by a factor of 3 larger than the ILLS errors. While this might in principle still satisfy the use of the MPS method, we assume that the considered situation actually slightly favors the ILLS methods. As the local geometry becomes nearly symmetric for large numbers of points, the ILLS methods naturally use this symmetry, while the MPS five point stencils do in general not show the same type of symmetry. The results presented in Section 8.2 second this interpretation.

8.2 Test Problem in 3d – Compare Solvers

We use the Fortran FPM code implemented in the *Fraunhofer Institut für Techno- und Wirtschaftsmathematik* to generate a point cloud on a 3d geometry, which is shown in Figure 8.13. The code admits to use point numbers up to $300,000$, which allows to compare efficient linear solvers for solving the arising linear system, in dependence of the number of points.

We generate Poisson discretization matrices similarly to the previous sections. On the one hand, we use an ALLS method with a radius chosen such that about 40 points are inside each circular neighborhood. On the other hand, we use the 40 points as candidates for the MPS method. In order to obtain information about performance of the linear solvers we save the MPS approximation is two ways: Firstly, as a new sparse matrix with only 10 nonzero entries per row. Secondly, as a sparse matrix with the structure of the ALLS method, i.e. for each candidate point which is not part of a basis solution we save a zero entry. This yields a sparse matrix which is structured exactly as the ALLS matrix and thus the effort of applying it is the same as for the ALLS matrix (assumed that a multiplication with zero takes the same time as a multiplication with a nonzero number).

Having obtained these approximation for various numbers of points, we apply two sparse matrix solvers to the linear systems:

1. A BiCGstab iteration with a least squares preconditioner
2. A BiCGstab iteration with SAMG as a preconditioner

The least squares preconditioner P is an approximation to the inverse of the system matrix A, in which only such entries are allowed to be nonzero, which are also nonzero in A. Among

all such matrices the one is chosen which minimizes $I - PA$ in a least squares sense. As the AMG preconditioner we use the SAMG multigrid solver [SAM], which is developed in the *Fraunhofer Institut für Algorithmen und Wissenschaftliches Rechnen*. The fundamental ideas are described by Stüben [Stü01].

Figure 8.14 shows the CPU times required to set up the linear systems. The solid line stands for the ALLS method, the dashed line for the MPS method, implemented with a two phase primal simplex method (i.e. there is potential for improvement). For both methods, the CPU times increase slightly superlinearly with the number of points, presumably due to data structure (circular neighbor identification, see Section 3.2.2) and memory effects. In our specific implementations, the MPS method is about 30% more expensive. For 280,000 points this makes a difference of about 18 seconds.

In Figure 8.15 the CPU times for a numerical solution with the BiCGstab solver are shown. The stopping criterion is the residual being reduced by 10^{-6}. All three approaches show a superlinear complexity in the number of points, as predicted by theory. The MPS matrices are significantly faster to solve than the ALLS matrices. However, the comparison with the MPS matrices with zeros indicates that the speed up is solely due to the sparsity of the MPS matrices. The pure convergence speed is about identical. This is interesting, since the MPS matrices are M-matrices and the ALLS matrices are not. On the other hand, in Section 8.1.2 we have observed that least squares discretization matrices with sufficiently many neighbors incorporated into the stencils can very well be inverse positive. Unfortunately, the interesting matrices are too large to admit a direct computation of their inverses. However, the fact that the converge performance is as good as for M-matrices, gives rise to the assumption that the ALLS matrices are indeed inverse positive, or at least very close to.

We apply the SAMG solver as a black box solver to the arising discretization matrices. Figure 8.16 shows the memory requirement. It is precisely proportional to the number of nonzero entries in the matrices. Consequently, it grows linearly, and it is by a factor of 3 to 4 smaller for the MPS method. Figure 8.17 shows the CPU times for solving the arising systems. We start with the same initial guess as for the BiCGstab method (zero vector). Also the stopping criterion is the same. Observe firstly that the SAMG solver is significantly faster than the BiCGstab solver. Solution times are reduced by a factor of 5. Observe secondly that the complexity grows (almost) linear with the number of points, as predicted by theory. AMG shows optimal complexity, both for the ALLS as for the MPS matrices. Thirdly notice that again solving the MPS system requires significantly less CPU time than solving the ALLS systems. However, as for the BiCGstab solver, the structural performance of the solver is not improved, as the test case with the MPS matrix with zeros indicates. Again, the speed up is significant, but only due to the sparsity of the MPS matrices.

It has to be pointed out that the SAMG solver is a very sophisticated black box solver. The implementation as a preconditioner in an outer iteration is extremely robust. The method

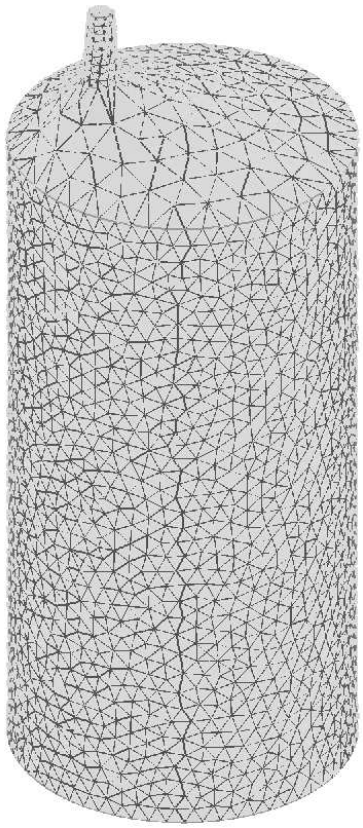

Figure 8.13: Bottle

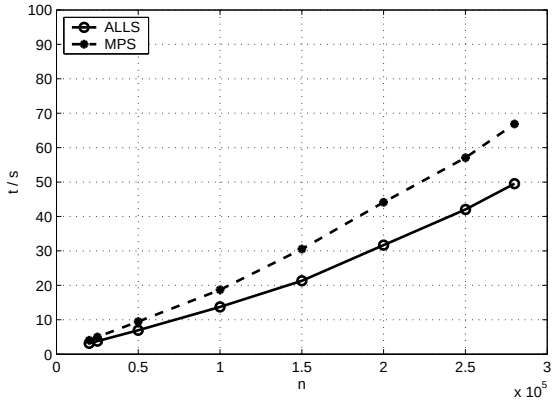

Figure 8.14: Setup Matrix

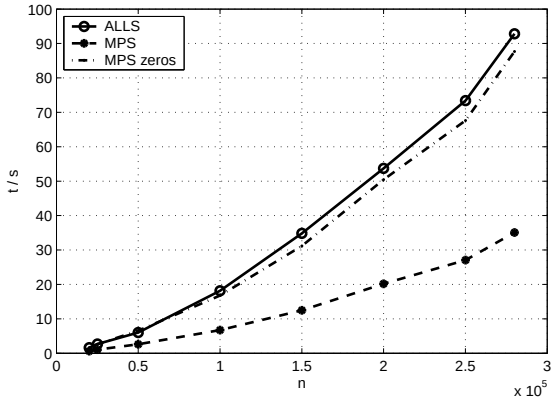

Figure 8.15: Solve with BiCGstab

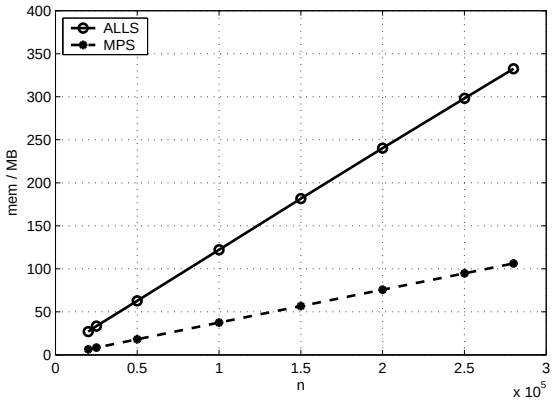

Figure 8.16: Memory SAMG

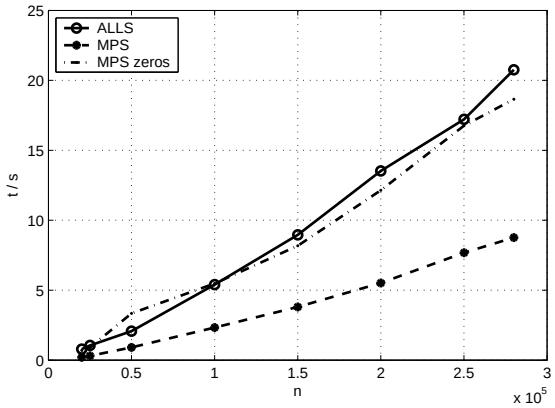

Figure 8.17: Solve with SAMG

can successfully be applied to all our discretization matrices. For our comparison of the approximation methods, the solver is in some sense "too good". A meshless setup causes problems, and a sophisticated linear solver can manage a "stupid" discretization. However, such a black box solver might not always be accessible. Also, there are no strict proofs that the SAMG solver always manages to solve any given discretization. In some situations one might wish to use a solver which can be guaranteed to converge always. With respect to these situations we consider in the following section a two grid AMLI solver, as presented in Section 7.4.4.

8.3 Investigate AMLI Solver

We consider a point cloud with 4000 interior points from the two dimensional Poisson test problem described in Section 8.1. The problem is discretized by the four approaches MPS, CS, ILLS 1, and ILLS 2, as described above. We consider the two-level AMG methods developed by Mense and Nabben, *Technical University of Berlin* [MN06], as presented in Section 7.4.4. The four versions of the method are applied to the resulting systems. As smoothers the Jacobi and the Gauß-Seidel method have been tested. Indeed, the Jacobi turned out to be slightly slower in computation times, but yielded completely analogous results as the Gauß-Seidel smoother, i.e. the relative performance of the different methods and matrices was the same for both smoothers. Consequently, we present only the results for the Gauß-Seidel case.

As coarsers the default coarser and the Ruge-Stüben coarsening strategy are applied, as described in Section 7.4.3. The considered splitting approaches are the AMLI, MAMLI, RMAMLI and SMAMLI method.

Figure 8.18 shows the run-times in comparison for the different matrices, splitting and coarsening strategies. Note that none of the methods converges for the matrices generated by the CS approach. On the other hand, the AMG method works well with the ILLS approaches, although the system matrices are both no L-matrices. The MPS matrices are M-matrices. Consequently, the method works also with these. Each run time consists of a setup phase, visualized by the lower dark bar, and the iteration phase, shown by the upper light bar.

Comparing the different splitting approaches, one can observe that the basic AMLI approach performs slightly worse, the other approaches show almost equal run times. More significant differences can be seen between the two coarsers. As expected, the Ruge-Stüben coarser has a larger setup time as the trivial default coarsening strategy. Surprising is the fact that the Ruge-Stüben coarsening does not result in a faster iteration phase. On the contrary, for the ILLS matrices, the Ruge-Stüben coarser actually yields longer iteration times.

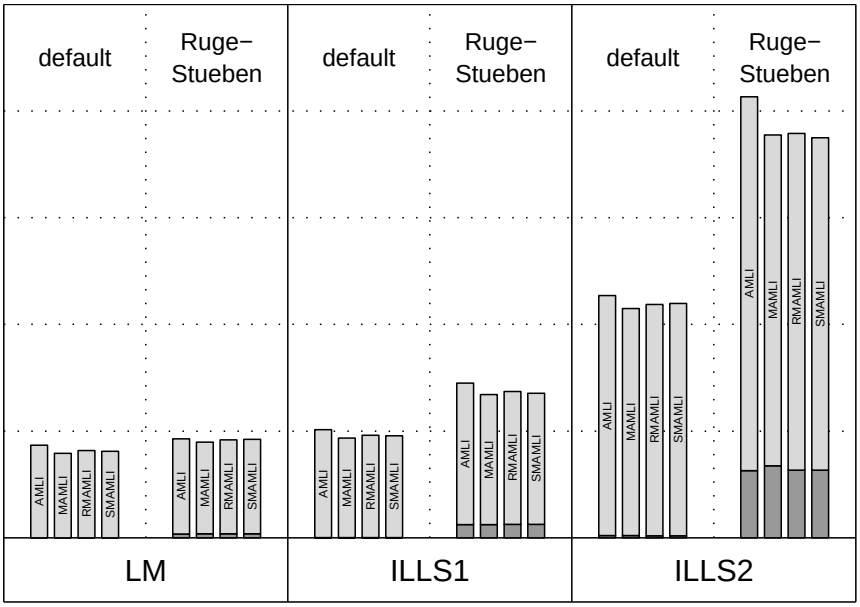

Figure 8.18: Run times for versions of AMLI solvers

When comparing the different matrices, the ILLS 2 matrix yields the longest computation times to solve. This is expected, since the number of nonzero entries is the largest. The MPS matrix is the fastest to solve. However, the speedup compared to the ILLS 1 matrix is by far not as significant as the number of nonzero matrix elements promises. The ILLS 1 matrix, although not an M-matrix, yields better convergence than the MPS matrix. Note that it is inverse positive, and it is very well possible that this property constitutes in a fast convergence. Still, the MPS approach yields the fastest to solve matrices, although the speed up is not as significant as with the SAMG approach, as shown in Section 8.2.

8.4 MPS Method in FPM Simulation

In order to investigate stability and robustness of the MPS method inside a time dependent particle simulation, we apply the method in a FPM simulation to discretize the arising Poisson problems. As test cases we consider the two applications described in Section 2.4.2, the melting of a glass cylinder and the filling process of a car tank. Both simulations have complex geometries in three space dimensions, involving various types of boundary conditions, in particular in both cases free boundaries are present. While the glass melting simulation takes place comparably slowly and is rather well behaved (the main challenge is the temperature dependent viscosity), the tank filling process is a fast, wild flow. Not turbulent, but unsteady and the precise form of the free boundary is very sensitive with respect to disturbances. We shall consider this aspect below.

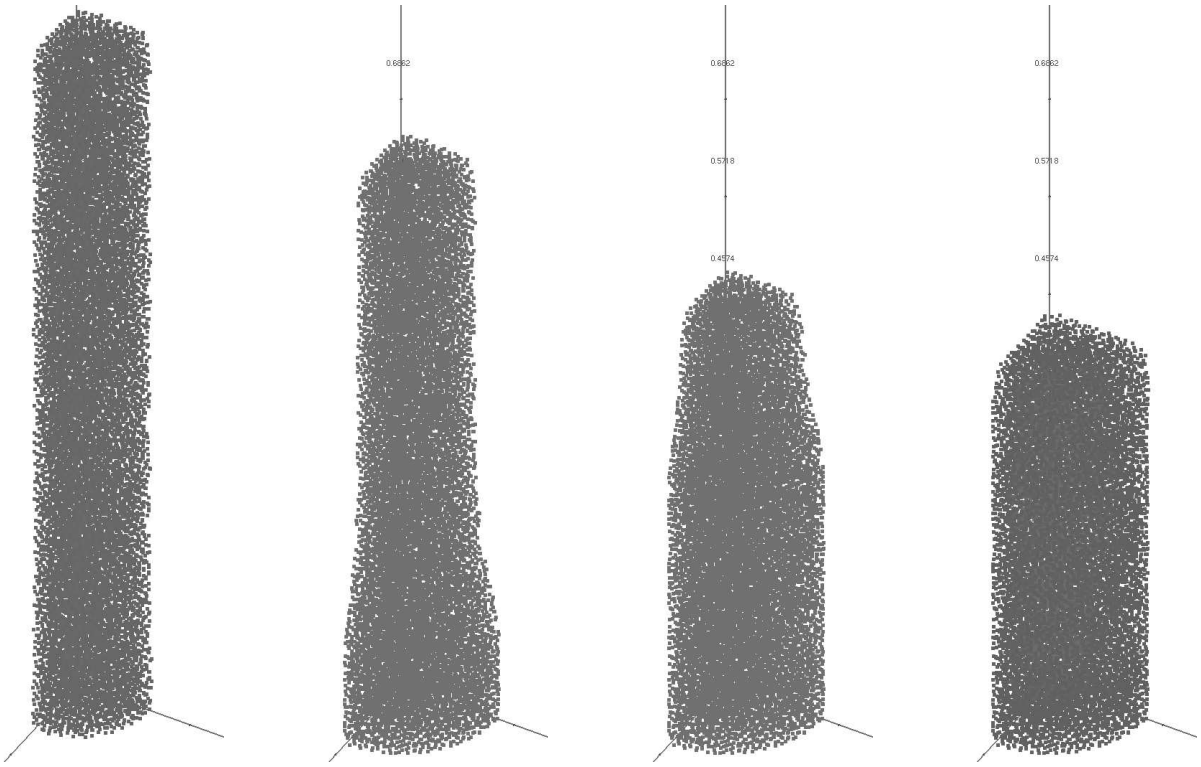

Figure 8.19: Glass melting simulation with LSQ approach

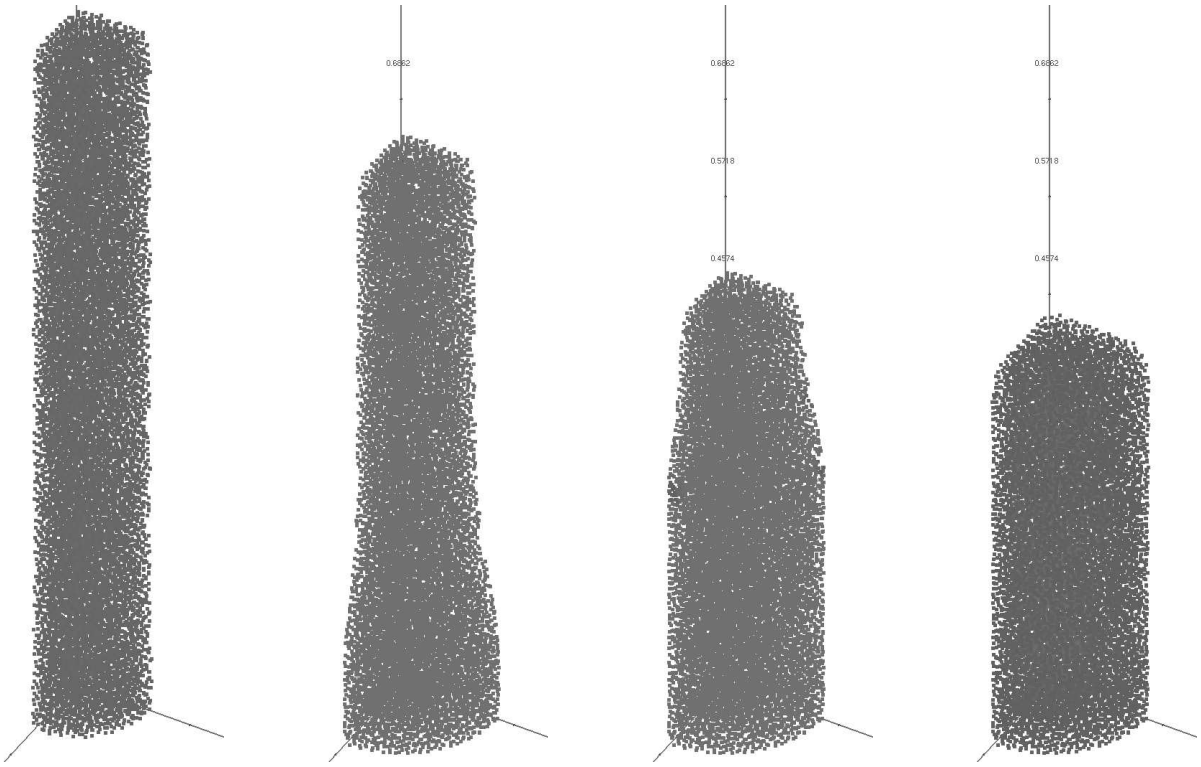

Figure 8.20: Glass melting simulation with MPS approach

For both simulations, geometric and physical data is available, and the FPM with a least squares approximation to solve the arising Poisson problems has been successfully applied. We replace the least squares approaches by the MPS method and run the simulations. It has to be pointed out that, although it is just a local change in the FPM code, the inclusion of the MPS is still tricky. In several instances, the FPM code is optimized for reciprocal matrices, which are obtained by least squares approaches. The MPS sacrifices reciprocity at the gain of significantly smaller stencils. Hence, the MPS can successfully included into the FPM code, however, due to these technical aspects, the structural advantages of the MPS are not used to full extent. Therefore we do not consider run times here. The investigations in the previous sections have given a good insight on the expected computation times anyhow. We instead compare the MPS-based FPM to the LSQ-based FPM in terms of stability and robustness, considering the obtained numerical solutions.

Glass Melting

The glass melting process, as described in Section 2.4.2, is considered. The same experiment is run twice, firstly with the least squares approach, secondly with the MPS approach for discretizing Laplace and normal derivative operators.

Figure 8.19 shows the evolution of the glass cylinder, computed with the LSQ-based FPM, at times $t_1 = 8200$s, $t_2 = 10600$s, $t_3 = 12000$s and $t_4 = 15000$s. Figure 8.20 shows the results of the MPS-based FPM computation. The times are approximately the same as for the LSQ-based simulation, although not exactly identical, since the time step is chosen adaptively. Additionally, the point clouds at later times are different. For small times, the point clouds are close, but as they slightly deviate, particle management will at some time include points in the one case and not in the other. Hence, a strict comparison in terms of errors would require to interpolate the numerical solutions both in space and in time. Since interpolation yields additional errors, the comparison itself would be complicated to perform.

As in this section we are interested in analyzing qualitatively, whether the MPS method performs stable in time dependent problems, we are content with comparing the results in the "eye-norm". Obviously, the results shown in Figure 8.20 do not differ noticeably from the results Figure 8.19. For the slowly moving melting process, the MPS performs well.

Tank Filling

The process of filling a car tank, as described in Section 2.4.2, is considered. In particular, the flow of petrol through the pipe connecting the nozzle inlet with the tank itself is computed from the point at which the liquid flow starts. The liquid falls and flows into the pipe. Due to initial velocity and gravity, the flow is rather fast and wild. As for the glass melting, we

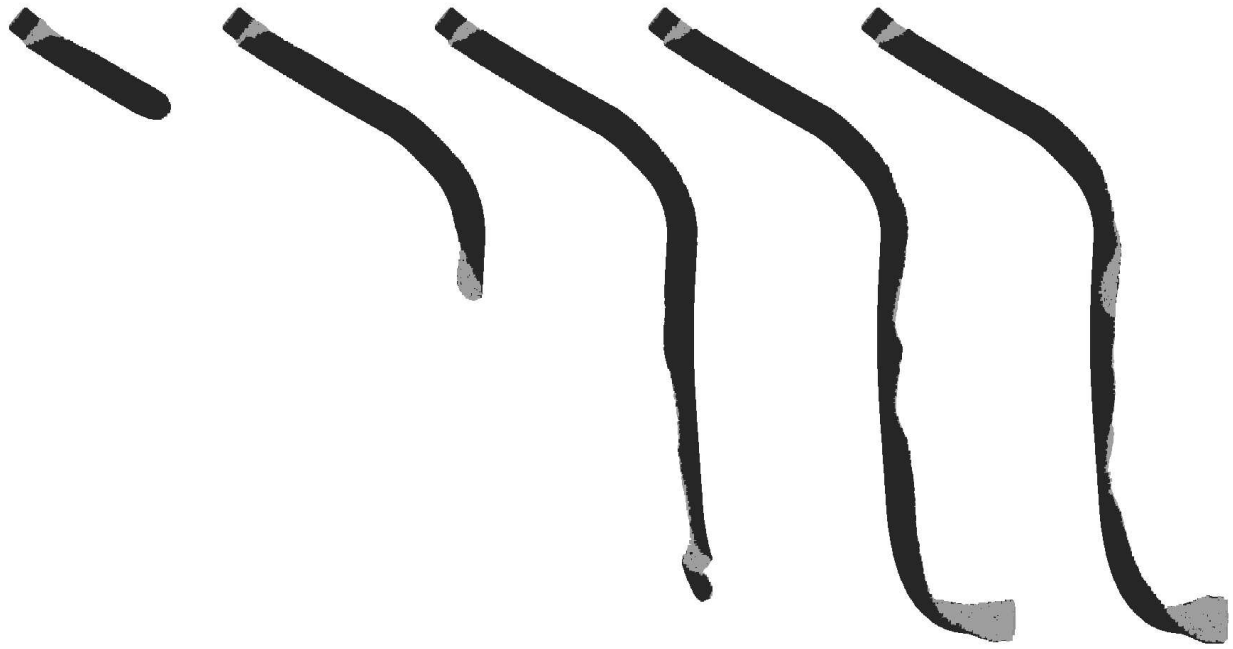

Figure 8.21: Tank filling simulation with LSQ approach (front view)

Figure 8.22: Tank filling simulation with LSQ approach (back view)

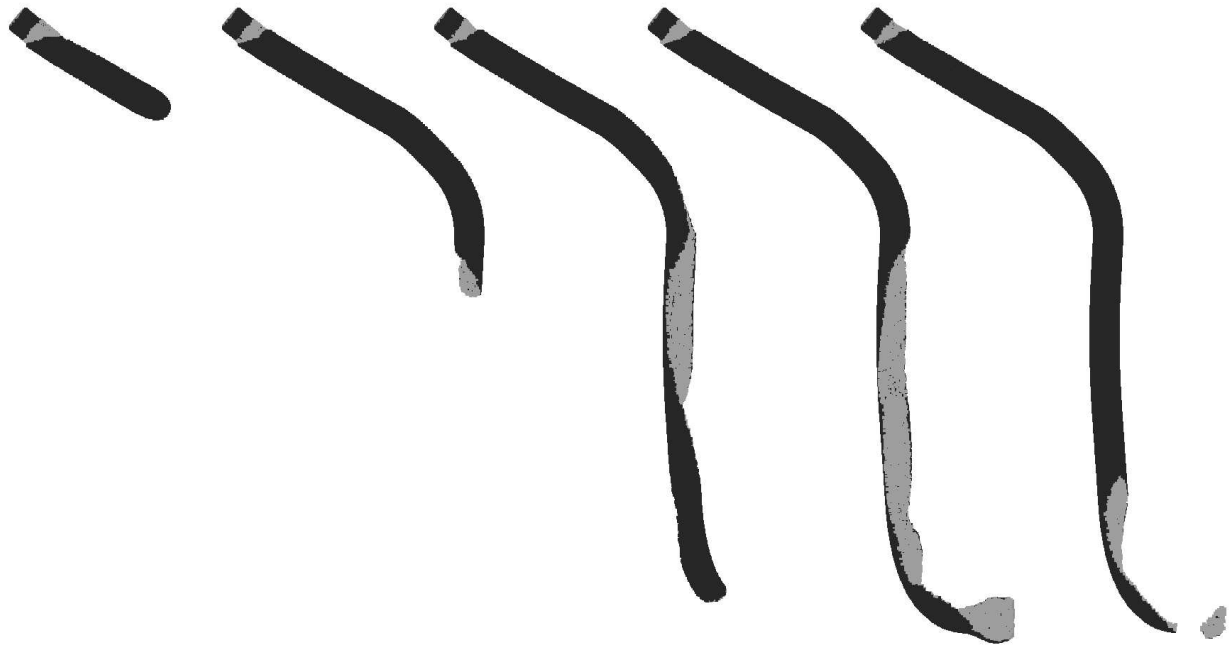

Figure 8.23: Tank filling simulation with MPS approach (front view)

Figure 8.24: Tank filling simulation with MPS approach (back view)

consider the same setup for two computation runs. Once with the LSQ-based FPM, and second with the MPS-based FPM.

The results of the computation with the LSQ-based FPM are shown in Figure 8.21 (looked at from the front) and Figure 8.22 (looked at from the back). Shown are five snapshots, taken at the times $t_1 = 0.09$s, $t_2 = 0.22$s, $t_3 = 0.32$s, $t_4 = 0.40$s and $t_5 = 0.49$s. The corresponding snapshots, now for the MPS-based FPM run, are shown in Figure 8.23 (front view) respectively Figure 8.24 (back view). As for the glass simulation, the second experiment does not have exactly the same times, but they are very close.

Considering corresponding snapshots in comparison, one can clearly see how the two solutions deviate with increasing time. At time t_1, the solutions are quite close. At time t_2, one can already see a difference in the form of the free boundary. At time t_5, finally, the form of the free boundary differs completely between the two simulations. Note that the physical problem itself is very ill conditioned in the sense that a small perturbation in the initial data or the geometry changes the solution at some later time dramatically. For small times, one can get a convergence for number of points going to infinity. However, for the times t_5 a convergence to *the* correct solution is far beyond the available computational power. Hence, at this point, we have no possibility to say which of the two solutions is correct.

On the other hand, for practical purposes, the precise solution is not required anyhow. Important are general properties, such as fluid pressure at specific points. These values can again be very stable. Indeed, the MPS solution shows similar features in the way the flow behaves, as LSQ-based solutions (different circular radii can also yield different solutions at large times). One significant difference can be seen at time t_5. The fluid domain becomes disconnected in the MPS simulation. This can in principle happen in reality. However, real measurements indicate that such a behavior is rather unlikely. In this sense the MPS solution is at least suspicious. Indeed, zooming into the geometry at time t_5 indicates a possible source of problems with the MPS approach: The geometry becomes very thin, thinner as twice the circular neighborhood radius. Consequently, an interior point in this small bridge may use boundary points both from the above and below. Since the MPS selects only 10 neighboring points, either not enough boundary points or not enough interior points might be incorporated into a stencil. Theorem 6.7 guarantees the chosen MPS-points to be nicely distributed, but if the domain is formed particularly unfriendly, problems can occur.

Presumably the MPS approach is slightly more sensitive to complicated geometries, such as thin layers, small tips, etc. One can remedy this problem by enforcing a higher particle resolution in these areas, even higher as required for least squares approaches. As the MPS method yields significantly sparser system matrices, one can certainly afford a higher resolution in these areas. Another possibility is to employ a least squares approximation in the problematic areas. Possibly using a domain decomposition approach in order to not spoil the M-matrix structure of the MPS approach.

Chapter 9

Conclusions and Outlook

In this thesis we have investigated various aspects of meshless Lagrangian particle methods. The particular focus is on the finite pointset method, but most of the considered aspects should similarly apply to other meshless methods. Applying the method to Burgers' equation we have shown that particle management is a fundamental feature, and that correct merging and inserting procedures are crucial.

We have considered the numerical solution of the incompressible Navier-Stokes equations using a projection approach. In this context, meshless Poisson problems have to be solved on the cloud of particles. The finite pointset method uses finite difference approximations to discretize the Poisson equation. As classical finite difference approaches we have considered least squares methods. These are a whole family of methods, from local to moving approaches, and we have also presented a semi-moving approach. We have compared the least squares approaches, both approximating and interpolating. The analysis of stencil values in specific cases as well as the application to a Poisson test problem indicates that firstly interpolating approaches have advantages over approximating approaches in the considered context, and secondly that moving approaches are rarely worth their additional effort compared to local approximation approaches.

As an important property of the finite difference matrices we have considered the M-matrix property. If the discretization matrix is an M-matrix, then various linear solvers can be guaranteed to converge. In particular, an AMLI twogrid solver can be shown to converge for M-matrices without requiring symmetry. We have presented how the system matrix can have such connection properties, that the system matrix is an M-matrix if all arising finite difference stencils are positive.

In the analysis of least squares approaches we have investigated the positivity of stencils. Indeed, least squares methods turn out to generally not yield positive stencils. We have formulated a new approximation approach which enforces the positivity of the finite difference

stencils. We based the approach on linear minimization with sign constraints. This new approach turned out to behave very differently to the least squares methods. The resulting stencils are minimal, i.e. as few points as necessary for a consistent approximation are considered. This property has turned out advantageous since the arising system matrices can be applied very fast, resulting in a fast solution. Additionally, significantly less memory is required. The method is called *minimal positive stencil* method.

We have shown that if the radius of considered neighboring points is sufficiently large (in relation to the maximum hole size of the point cloud), then positive stencils always exist. The minimal positive stencil method yields such a positive stencil, which is a selection of points positioned close to the central point and nicely distributed around it. The arising system matrices are very sparse M-matrices. On the other hand, the approach has various disadvantages. The generated stencils do not depend continuously on the point geometry. The resulting matrices are not reciprocal. Also, the aspect that very few points are selected can be a disadvantage with respect to the convergence speed of iterative solvers and coarsening properties of algebraic multigrid.

We have investigated various methods for solving the linear programs arising in the minimal positive stencil approach. Simplex approaches turned out to perform best for these small systems. The computational effort for finding a minimal positive stencil roughly equals the effort of computing a least squares stencil. An aspect to consider in more detail are approaches to "guess" a positive stencil. Thus the minimal positive stencil approach could be significantly accelerated.

The minimal positive stencil approach yields a new neighborhood concept. As opposed to classical neighborhood concepts, this one is not solely geometrical, but is with respect to the considered operator. An open question is whether the method can be interpreted as an interpolation approach. While least squares methods can be derived from an approach for function reconstruction, the minimal positive stencil approach is derived directly as an approximation to the Laplace operator.

By numerical experiments we have compared the minimal positive stencil method with least squares approaches. Also, different solvers for the arising linear systems have been compared. A general result is that the minimal positive stencil systems are the fastest to solve. However, this speed up compared to least squares approaches is due to the sparsity of the arising matrices. In terms of pure computation speed the least squares matrices are as good or better than the minimal positive stencil matrices. Also, the least squares systems yield slightly smaller errors, such that in the complete balance the speedup is not as significant as the matrix sparsity indicates. Still, a noticeable acceleration can be achieved, for instance in combination with the SAMG solver. The specific implementation of the method can still be improved.

The minimal positive stencil method has been implemented into the finite pointset method and applied in examples of complex time dependent problems. The method has turned out to perform stable and fast. On the other hand, the sparse selection of stencil points can result in problems in the case of thin geometries. We have seen that enforcing specific points into the minimal stencil cannot easily be done from a computational point of view. The aspects of thin geometries require more investigation. In the case of very thin geometries, anisotropic point clouds have to be used. It is not completely clear to which extent the minimal positive stencil method can be applied in this case. The aspect is related to approximating general second order elliptic operators instead of the Laplace operator.

Algebraic multigrid methods have turned out to perform well for the systems arising as finite difference discretization of meshless Poisson problems. A drawback of these methods is the additional effort in the initial setup phase. Of interest can firstly be an acceleration of the setup phase by using geometric information, and secondly the interpretation of coarsening strategies in the geometry of the point cloud. Geometric multigrid approaches can be formulated and compared to the algebraic methods.

List of Figures

2.1 Glass Melting 23

2.2 Tank filling 23

2.3 FPM without particle management, inviscid Burgers' equation 29

2.4 FPM with particle management, inviscid Burgers' equation 29

2.5 FPM without particle management, inviscid Burgers' equation 30

2.6 FPM with incorrect particle management, inviscid Burgers' equation 30

2.7 Particle movement with correct interpolation 32

2.8 Effective shock speed 32

2.9 Particle movement with incorrect interpolation 32

2.10 FPM without particle management, small viscosity 34

2.11 FPM with particle management, small viscosity 34

2.12 FPM without particle management, large viscosity 35

2.13 FPM with particle management, large viscosity 35

3.1 A typical computational domain 38

3.2 Cloud of 40 points 38

3.3 Cloud of 200 points 38

3.4 Cloud of 1000 points 38

3.5 Cloud with domain 38

3.6 Minimum number of points inside a ball 40

3.7 Maximum number of points inside a ball 40

3.8 Circular neighborhood 47
3.9 Delaunay neighborhood 47
3.10 Delaunay neighborhood may yield too few neighbors 47
3.11 Four quadrant neighborhood 48
3.12 Inverse convex hull neighborhood 48
3.13 Positive stencil neighborhood 48

4.1 Stencil solution with 5 points 59
4.2 No stencil solution with 6 points 59
4.3 Multiple stencil solutions with 6 points 59

5.1 Stencil for first derivative, AMLS, quadratic basis 89
5.2 Stencil for second derivative, AMLS, quadratic basis 90
5.3 Stencil for one sided first derivative, AMLS, linear basis 95
5.4 Stencil for second derivative, IMLS, quadratic basis 102
5.5 Stencil for one sided first derivative, IMLS, linear basis 103
5.6 Stencil for second derivative, AMLS vs. IMLS, quadratic basis 106
5.7 Condition numbers for 7 point least squares system 109
5.8 Errors for 7 point least squares system 109
5.9 Condition numbers for 70 point least squares system 110
5.10 Errors for 70 point least squares system 110
5.11 Central stencil acting on fourth order polynomial 112

6.1 Quadratic minimization can yield non-positive stencils 124
6.2 Positive stencil by linear minimization 124
6.3 Minimal positive stencil for varying locality parameter 129
6.4 Hexagonal grid with least squares and two possible MPS stencils 130
6.5 No positive stencil 134
6.6 Positive stencil 134

6.7 Necessary criterion for positive stencils 134

6.8 Case of two negative eigenvalues 135

6.9 Case of opposite signed eigenvalues 135

6.10 Case of one zero eigenvalue, type 1 135

6.11 Case of one zero eigenvalue, type 2 135

6.12 The 2d cone estimate is sharp 137

6.13 Relation between circular neighborhood radius and mesh size 141

6.14 Guaranteeing the cone criterion close to the boundary 141

6.15 Neumann point with circular neighborhood 143

6.16 Minimization function for different angles 143

6.17 Minimal positive Neumann stencil for varying locality parameter 145

6.18 An interior point is not reached 147

6.19 Failure to reach a Dirichlet point 147

6.20 Connection failure with a regular grid 147

6.21 Run times for various optimization methods 150

6.22 Large circular neighborhood 155

6.23 Small circular neighborhood 155

6.24 Delaunay neighborhood 155

6.25 Four quadrant neighborhood 155

6.26 Inverse convex hull neighborhood 155

6.27 Positive stencil neighborhood 155

6.28 MPS mosaic for a given set of points 157

8.1 Point clouds in circular domain 174

8.2 Sparse matrix structure of MPS matrix 176

8.3 Sparse matrix structure of CS matrix 176

8.4 Sparse matrix structure of matrix ILLS 1 176

8.5 Sparse matrix structure of matrix ILLS 2 176
8.6 Number of nonzero entries in system matrix A 177
8.7 Number of positive off-diagonal entries in A 177
8.8 Number of negative entries in A^{-1} 178
8.9 Error with MPS approach 180
8.10 Error with CS approach 180
8.11 Error with ILLS 1 approach 180
8.12 Error with ILLS 2 approach 180
8.13 Bottle 183
8.14 Setup Matrix 183
8.15 Solve with BiCGstab 183
8.16 Memory SAMG 183
8.17 Solve with SAMG 183
8.18 Run times for versions of AMLI solvers 185
8.19 Glass melting simulation with LSQ approach 186
8.20 Glass melting simulation with MPS approach 186
8.21 Tank filling simulation with LSQ approach (front view) 188
8.22 Tank filling simulation with LSQ approach (back view) 188
8.23 Tank filling simulation with MPS approach (front view) 189
8.24 Tank filling simulation with MPS approach (back view) 189

Index

AMLI, 169, 184
  multiplicative, 170
    reverse, 170
    symmetrized, 170

BiCGstab, 163, 181
breaking wave solution, 30
Burgers' equation, 26

Cholesky factorization, 113
Chorin projection method, 10
coarsening
  aggressive, 168
  default, 168
  Ruge–Stüben, 168
cone criterion, 135, 140
  2d, 135
  3d, 137
  boundary, 141
connected index, 53
continuity equation, 6
control volume, 6

Delaunay triangulation, 38, 43

Euler equations, 16
Eulerian point of view, 10

Farkas' Lemma, 132, 144
finite pointset method, 10, 185

Hopf-Cole transformation, 28
hybrid method, 13, 48, 156, 179

immersed interface method, 122
incompressible flow, 8
incompressible fluid, 8
internal energy, 9
iterative scheme, 160
  consistent, 160
  convergent, 160

Krylov subspace, 162

Lagrangian point of view, 10
linear programming, 148
  fundamental theorem, 126

material derivative, 6
matrix
  diagonally dominant
    essentially, 120
    strictly, 119
    weakly, 119
  graph, 53
    symmetric, 53
  inverse positive, 118
  irreducible, 53
    essentially, 53
  L, 118
  M, 52, 108, 119, 125, 161, 168
  Z, 118
minimal positive stencil method, 49, 126, 153
momentum balance equation, 7
moving least squares, 13, 71
  approximating, 72, 77
  interpolating, 72, 96
multigrid, 164
  algebraic, 167, 182
  convergence, 168
  cycle

F, 166
V, 166
W, 166
full, 166
geometric, 166

Navier-Stokes equations, 8, 20
neighborhood, 45, 154
circular, 46, 155
Delaunay, 47, 152, 155
four quadrant, 49, 155
inverse convex hull, 49, 152, 155
positive stencil, 49, 155
symmetric, 46
Neumann boundary point, 52, 58, 75, 141, 154

particle, 10, 17
particle management, 25, 45
point cloud, 37, 51
density, 39
generation, 44
manipulation, 45
mesh size, 39, 140, 153
separation, 39
preconditioner, 160
left, 163
right, 163

Rankine-Hugoniot condition, 27
residual, 160, 182
Reynolds number, 10
Reynolds transport theorem, 6

SAMG, 168, 182
Shepard's method, 26, 74
simplex method, 148
pivot rule, 151
two phase, 149
smoothed particle hydrodynamics, 14
smoothing length, 15, 47, 156
stencil, 52
consistent, 55
Laplace, 39, 52, 65, 122, 133
positive, 56
stress tensor, 7
reduced, 7

Vandermonde matrix, 55, 61, 123, 148
Voronoi cell, 43, 156

Bibliography

[AH70a] A.A. Amsden and F.H. Harlow. A simplified MAC technique for incompressible fluid flow calculations. *J. Comp. Phys.*, 6:322–325, 1970.

[AH70b] A.A. Amsden and F.H. Harlow. The SMAC method: A numerical technique for calculating incompressible fluid flows. Technical Report LA–4370, Los Alamos Scientific Laboratory, 1970.

[Aj95] J.D. Anderson jr. *Computational Fluid Dynamics: The basics with applications*. Mc Graw Hill, 1995.

[AV89] O. Axelsson and P. Vassilevski. Algebraic multilevel preconditioning methods, Part I. *Numer. Math.*, 56:157–177, 1989.

[AV90] O. Axelsson and P. Vassilevski. Algebraic multilevel preconditioning methods, Part II. *SIAM J. Numer. Anal.*, 27:1569–1590, 1990.

[Axe94] O. Axelsson. *Iterative solution methods*. Cambridge University Press, 1994.

[BJ96] I. Babuska and Melenk J.M. The partition of unity finite element method: basic theory and applications. *Comput. Methods Appl. Mech. Engrg.*, 139:289–314, 1996.

[BJ97] I. Babuska and Melenk J.M. The partition of unity method. *Internat. J. Numer. Methods Engrg.*, 40:727–758, 1997.

[BLG94] T. Belytschko, Y.Y. Lu, and L. Gu. Element-free galerkin methods. *Internat. J. Numer. Methods Engrg.*, 37:229–256, 1994.

[BMR84] A. Brandt, S. McCormick, and J. Ruge. Algebraic multigrid (AMG) for sparse matrix equations. In D.J. Evans, editor, *Sparsity and its Applications*, pages 257–284. Cambridge University Press, 1984.

[Bra73] A. Brandt. Multi-level adaptive technique (MLAT) for fast numerical solutions to boundary value problems. In H. Cabannes and R. Temam, editors, *Lecture*

Notes in Physics 18, pages 82–89. Proc. 3rd Int. Conf. Numerical Methods in Fluid Mechanics, 1973.

[Cho68] A.J. Chorin. Numerical solution of the Navier-Stokes equations. *Math. Comput.*, 22:745–762, 1968.

[Chv83] V. Chvátal. *Linear programming*. W. H. Freeman and Company, 1983.

[CM00] A.J. Chorin and J.E. Marsden. *A mathematical introduction to fluid mechanics*. Springer, third edition, 2000.

[CPL] ILOG CPLEX. Webpage at URL: `http://www.ilog.com/products/cplex/`.

[CR98] S.J. Cummins and M. Rudman. Truly incompressible SPH. In *ASME Fluids Engineering Division Summer Meeting*, Washington DC, USA, 1998.

[CR99] S.J. Cummins and M. Rudman. An SPH projection method. *J. Comput. Phys.*, 152(2):584–607, 1999.

[Dev86] L. Devroye. *Non-uniform random variate generation*. Springer, New York, 1986.

[Dil99] G.A. Dilts. Moving least squares particles hydrodynamics I, consistency and stability. *Internat. J. Numer. Methods Engrg.*, 44:1115–1155, 1999.

[DKL84] L. Demkowicz, A. Karafiat, and T. Liszka. On some convergence results for FDM with irregular meshes. *Comput. Methods Appl. Mech. Engrg.*, 42:343–355, 1984.

[DLT96] L. Duarte, T. Liszka, and W. Tworzyako. hp-meshless cloud method. *Comput. Methods Appl. Mech. Engrg.*, 139(1–7):263–288, 1996.

[DO96] C.A. Duarte and J.T. Oden. hp clouds – an hp-meshless method. *Numer. Methods Partial Differential Equations*, 12:673–705, 1996.

[Dua95] C. A. Duarte. A review of some meshless methods to solve partial differential equations. TICAM Report 95-06, The University of Texas at Austin, 1995.

[Eva98] L.C. Evans. *Partial differential equations*, volume 19 of *Graduate Studies in Mathematics*. American Mathematical Society, 1998.

[Fed61] R.P. Federenko. A relaxation method for solving elliptic difference equations. *USSR Comp. Math. and Math. Phys.*, 1(5):1092–1096, 1961.

[FLM97] K. Fukuda, T.M. Liebling, and F. Margot. Analysis of backtrack algorithms for listing all vertices and all faces of a convex polyhedron. *Comput. Geom.*, 8:1–12, 1997.

[For87] S.J. Fortune. A sweepline algorithm for Voronoi diagrams. *Algorithmica*, 2:153–174, 1987.

[FP99] J.H. Ferziger and M. Peric. *Computational methods for fluid dynamics*. Springer, second edition, 1999.

[FPM] ITWM FPM. Webpage at URL: `http://www.itwm.fhg.de/en/tv_tvcompetences/gitterfreie_methoden/`.

[FR04] T. Fujimoto and R.R. Ranade. Characterization of inverse-positive matrices: The Hawkins-Simon condition and the Le Chatelier-Braun principle. *Electron. J. Linear Algebra*, 11:59–65, 2004.

[Fri95] U. Frisch. *Turbulence*. Cambridge University Press, 1995.

[FS01] J. Fürst and T. Sonar. On meshless collocation approximations of conservation laws: Preliminary investigations on positive schemes and dissipation models. *ZAMM Z. Angew. Math. Mech.*, 81:403–415, 2001.

[GDN98] M. Griebel, T. Dornseifer, and T. Neunhoeffer. *Numerical simulation in fluid dynamics: A practical introduction*. Society for Industrial and Applied Mathematics, 1998.

[GLY00] C. Gauger, P. Leinen, and H. Yserentant. The finite mass method. *SIAM J. Numer. Anal.*, 37(6):1768–1799, 2000.

[GM77] R.A. Gingold and J.J. Monaghan. Smoothed particle hydrodynamics – Theory and application to nonspherical stars. *Mon. Not. R. Astron. Soc.*, 181:375, 1977.

[GS00] M. Griebel and M. A. Schweitzer. A particle-partition of unity method for the solution of elliptic parabolic and hyperbolic PDE. *SIAM J. Sci. Comput.*, 22:853–890, 2000.

[GS02a] M. Griebel and M. A. Schweitzer. A particle-partition of unity method – Part II: Efficient cover construction and reliable integration. *SIAM J. Sci. Comput.*, 23:1655–1682, 2002.

[GS02b] M. Griebel and M. A. Schweitzer. A particle-partition of unity method – Part III: A multilevel solver. *SIAM J. Sci. Comput.*, 24:377–409, 2002.

[Hac76] W. Hackbusch. Ein iteratives Verfahren zur schnellen Auflösung elliptischer Randwertprobleme. Report 76-12, Universität Köln, 1976.

[Hac85] W. Hackbusch. *Multigrid methods and applications*. Springer-Verlag, Berlin, 1985.

[Hac93] W. Hackbusch. *Iterative Lösung großer schwachbesetzter Gleichungssysteme*. Teubner, 1993.

[Har96] M. Hartig. *Randbedingungen für Partikelmethoden in der Strömungsmechanik*. Diplomarbeit, Department of Mathematics, University of Kaiserslautern, 1996.

[HK89] L. Hernquist and N. Katz. TREESPH: A unification of SPH with the hierarchical tree method. *Astrophys. J. Suppl.*, 70(6):419–446, 1989.

[HSS00] D. Hietel, K. Steiner, and J. Struckmeier. A finite-volume particle method for compressible flows. *Math. Models Methods Appl. Sci.*, 10:1363–1382, 2000.

[IT02] O. Iliev and S. Tiwari. A generalized (meshfree) finite difference discretization for elliptic interface problems. In *Numerical Methods and Applications*, volume 2502. Lecture notes in computer sciences, Springer, 2002.

[Kar84] N. Karmarkar. A new polynomial time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.

[Kha79] L. Khachiyan. A polynomial time algorithm in linear programming. *Doklady Adademiia Nauk SSSR*, 244:1093–1096, 1979.

[Kle98] M. Kleiber. *Handbook of computational solid mechanics*. Survey and Comparison of Contemporary Methods. Springer, 1998.

[KT01] J. Kuhnert and S. Tiwari. Finite pointset method based on the projection method for simulations of the incompressible Navier-Stokes equations. Bericht 30, Fraunhofer Institut für Techno- und Wirtschaftsmathematik, 2001.

[KT02a] J. Kuhnert and S. Tiwari. Finite pointset method based on the projection method for simulations of the incompressible Navier-Stokes equations. In M. Griebel and M.A. Schweitzer, editors, *Meshfree methods for Partial Differential Equations*, volume 26 of *Lecture Notes in Computational Science and Engineering*, pages 373–388. Springer LNCSE, 2002.

[KT02b] J. Kuhnert and S. Tiwari. A meshfree method for incompressible fluid flows with incorporated surface tension. *Revue aurope'enne des elements finis*, 11/7–8:965–987, 2002.

[KT03] J. Kuhnert and S. Tiwari. Particle method for simulation of free surface flows. In T. Hou and E. Tadmor, editors, *Proceedings of the Ninth International Conference on Hyperbolic Problems*, pages 889–898. Springer, 2003.

[Kuh99] J. Kuhnert. *General Smoothed Particle Hydrodynamics*. Dissertation, Department of Mathematics, University of Kaiserslautern, 1999.

[Kuh02] J. Kuhnert. An upwind finite pointset method for compressible Euler and Navier-Stokes equations. In M. Griebel and M.A. Schweitzer, editors, *Meshfree methods for Partial Differential Equations*, volume 26 of *Lecture Notes in Computational Science and Engineering*, pages 239–250. Springer LNCSE, 2002.

[Kun01] M. Kunle. *Entwicklung und Untersuchung von Moving Least Square Verfahren zur numerischen Simulation hydrodynamischer Gleichungen*. Dissertation, Fakultät für Mathematik und Physik, Eberhard-Karls-Universität zu Tübingen, 2001.

[Lev98] D. Levin. The approximation power of moving least-squares. *Math. Comp.*, 67:1517–1531, 1998.

[Li03] Z.L. Li. An overview of the immersed interface method and its applications. *Taiwanese J. Math.*, 7(1):1–49, 2003.

[Lis84] T. Liszka. An interpolation method for an irregular net of nodes. *Int. J. Num. Meth. Eng.*, 20:1599–1612, 1984.

[Lis06] T. Liszka. Private communication, May 2006.

[LL91] L.D. Landau and E.M. Lifschitz. *Lehrbuch der theoretischen Physik, Band VI: Hydrodynamik*. Akademie-Verlag, Berlin, fifth edition, 1991.

[LMS94] I. Lustig, R. Marsten, and D. Shanno. Interior point methods: Computational state of the art. *ORSA Journal on Computing*, 6:1–14, 1994.

[LO80] T. Liszka and J. Orkisz. The finite difference method at arbitrary irregular grids and its application in applied mechanics. *Comput. & Structures*, 11:83–95, 1980.

[Lor90] R.A. Lorentz. A new conservative discretization of the laplacian on irregular grids. Technical report, Fraunhofer Institut für Methodische Grundlagen, 1990.

[Lor92] R.A. Lorentz. *Multivariate Birkhoff interpolation*. Lecture Notes in Mathematics, Springer, 1992.

[LS81] P. Lancaster and K. Salkauskas. Surfaces generated by moving least squares methods. *Math. Comp.*, 37:141–158, 1981.

[LS86] P. Lancaster and K. Salkauskas. *Curve and surface fitting: An introduction*. Academic Press, 1986.

[Luc77] L. Lucy. A numerical approach to the testing of the fission hypothesis. *Astronomical Journal*, 82:1013–1024, 1977.

[LV92] R.J. Le Veque. *Numerical methods for conservation laws*. Birkhäuser, second edition, 1992.

[MN06] C. Mense and R. Nabben. On algebraic multilevel methods for nonsymmetric system – Convergence results. *SIAM J. Numer. Anal.*, to appear, 2006.

[Mon88] J.J. Monaghan. An introduction to SPH. *Comput. Phys. Comm.*, 48:89–96, 1988.

[Mon92] J.J. Monaghan. Smoothed Particle Hydrodynamics. *Ann. Rev. Astron. Astrophys.*, 30:543–574, 1992.

[Mon94] J.J. Monaghan. Simulating free surface flows with SPH. *J. Comput. Phys.*, 110(2):399–406, 1994.

[MR95] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[MT02] S. Manservisi and S. Tiwari. Modeling incompressible flows by least-squares approximation. *Nepali Math. Sci. Rep.*, 20, 1&2:1–23, 2002.

[NTVR92] B. Nayroles, G. Touzot, P. Villon, and A. Ricard. Generalizing the finite element method: Diffuse approximation and diffuse elements. *Comput. Mech.*, 10(5):307–318, 1992.

[OM05] J. Orkisz and S. Milewski. On higher order approximation in the MFDM method. In *Third MIT Conference on Computational Fluid and Solid Mechanics*, Cambridge, USA, 2005. Massachusetts Institute Of Technology.

[PK75] N. Perrone and R. Kao. A general finite difference method for arbitrary meshes. *Comput. & Structures*, 5:45–58, 1975.

[PS72] S.V. Patankar and D.A. Spalding. A calculation procedure for heat, mass and momentum transfer in three dimensional parabolic flows. *Int. J. Heat and Mass Trans.*, 15:1787–1806, 1972.

[QSS00] A. Quarteroni, R. Sacco, and F. Saleri. *Numerical mathematics*. Springer, 2000.

[Ras00] F.A. Rasio. Particle methods in astrophysical fluid dynamics. In Hiwatari Y. et al., editor, *Proceedings of the 5th International Conference on Computational Physics (ICCP5)*, volume 138 of *Progress of Theoretical Physics Supplement*, pages 609–621, 2000.

[RS87] J.W. Ruge and K. Stüben. Algebraic multigrid. In S.F. MacCormick, editor, *Multigrid methods*, volume 3, pages 73–130. SIAM, Philadelphia, PA, 1987.

[SAM] SCAI SAMG. Webpage at URL: `http://www.scai.fhg.de/samg.0.html`.

[Sei06a] B. Seibold. Multigrid and M-matrices in the finite pointset method. In *Proceedings in Applied Mathematics and Mechanics*, 2006.

[Sei06b] B. Seibold. Multigrid and M-matrices in the finite pointset method for incompressible flows. In M. Griebel and M.A. Schweitzer, editors, *Meshfree methods for Partial Differential Equations*, 2006.

[She68] D. Shepard. A two-dimensional interpolation function for irregularly spaced points. In *Proceedings of A.C.M National Conference*, pages 517–524, 1968.

[She99] J.R. Shewchuk. Lecture notes on Delaunay mesh generation. Lecture notes, University of California at Berkeley, 1999.

[SJ94] I.H. Sloan and S. Joe. *Lattice methods for multiple integration*. Oxford Science Publications, 1994.

[Son05] T. Sonar. Difference operators from interpolating moving least squares and their deviation from optimality. *M2AN Math. Model. Numer. Anal.*, 39(5):883–908, 2005.

[Stü83] K. Stüben. Algebraic multigrid (AMG): Experiences and comparisons. *Appl. Math. Comp.*, 13:419–451, 1983.

[Stü01] K. Stüben. A review of algebraic multigrid. *J. Comput. Appl. Math.*, 128:281–309, 2001.

[TOS01] U. Trottenberg, C.W. Oosterlee, and A. Schüller. *Multigrid*. Academic Press, 2001.

[Van01] R.J. Vanderbei. *Linear programming: Foundations and extensions*. International Series in Operations Research & Management Science. Springer, second edition, 2001.

[Var00] R.S. Varga. *Matrix iterative analysis*. Springer, second edition, 2000.

[Vil99] J.P. Vila. On particle weighted methods and smooth particle hydrodynamics. *Math. Models Methods Appl. Sci.*, 9(2):161–209, 1999.

[Vor07] G. Voronoi. Nouvelles applications des paramtres continus la thorie des formes quadratiques. *J. Reine Angew. Math.*, 133:97–178, 1907.

[Wes92] P. Wesseling. *An introduction to multigrid methods*. John Wiley and Sons, 1992.

[Yse97] H. Yserentant. A particle model of compressible fluids. *Numer. Math.*, 76:111–142, 1997.

[ZR66] Ya.B. Zel'dovich and Yu.P. Raizer. *Elements of gasdynamics and the classical theory of shock waves*. Academic Press, 1966.

Benjamin Seibold

http://www.mathematik.uni-kl.de/~seibold/

Bildungsweg

12/1976	Geboren in Kaiserslautern
07/1983 – 06/1987	Grundschule Hohenecken
07/1987 – 06/1996	Hohenstaufengymnasium Kaiserslautern
06/1996	Abitur
09/1996 – 09/1997	Zivildienst, Deutsches Rotes Kreuz Kaiserslautern
10/1997 – 04/2003	Studium Technomathematik (Nebenfach Physik), Universität Kaiserslautern
09/1999	Vordiplom
09/2000 – 06/2001	Auslandsaufenthalt, UC Berkeley und Lawrence Berkeley Lab Stipendium der Wipprecht Stiftung
04/2003	Diplom in Technomathematik Diplomarbeit: *Optimal prediction in molecular dynamics* Abschluss in Mathematics International, ECMI-Zertifikat
06/2003 – 08/2006	Promotionsstudent in Mathematik, TU Kaiserslautern DFG Stipendium im Graduiertenkolleg *Mathematik und Praxis*

Resume

12/1976	Born in Kaiserslautern
07/1983 – 06/1987	Primary school Hohenecken
07/1987 – 06/1996	High school: Hohenstaufengymnasium Kaiserslautern
06/1996	Abitur
09/1996 – 09/1997	Social service, German Red Cross Kaiserslautern
10/1997 – 04/2003	Studies technomathematics (minor: physics), University of Kaiserslautern
09/1999	Prediploma
09/2000 – 06/2001	Stay at UC Berkeley und Lawrence Berkeley Lab Scholarship of Wipprecht foundation
04/2003	Diploma in technomathematics Diplom thesis: *Optimal prediction in molecular dynamics* Diploma of Mathematics International, ECMI certificate
06/2003 – 08/2006	PhD student in mathematics, University of Kaiserslautern DFG scholarship in graduate college *Mathematics and practice*