Mälardalen University Doctoral Dissertation 300

Minimum Enclosing Balls and Ellipsoids in General Dimensions

Linus Källberg

MÄLARDALEN UNIVERSITY
SWEDEN

Mälardalen University Press Dissertations
No. 300

MINIMUM ENCLOSING BALLS AND ELLIPSOIDS IN GENERAL DIMENSIONS

Linus Källberg

2019

MÄLARDALEN UNIVERSITY
SWEDEN

School of Innovation, Design and Engineering

Copyright © Linus Källberg, 2019
ISBN 978-91-7485-448-0
ISSN 1651-4238
Printed by E-Print AB, Stockholm, Sweden

Mälardalen University Press Dissertations
No. 300

MINIMUM ENCLOSING BALLS AND ELLIPSOIDS IN GENERAL DIMENSIONS

Linus Källberg

Akademisk avhandling

som för avläggande av teknologie doktorsexamen i datavetenskap vid
Akademin för innovation, design och teknik kommer att offentligen försvaras
fredagen den 10 januari 2020, 13.15 i Beta, Mälardalens högskola, Västerås.

Fakultetsopponent: doktor Selin Damla Ahipaşaoğlu,
Singapore University of Technology & Design

MÄLARDALEN UNIVERSITY
SWEDEN

Akademin för innovation, design och teknik

Abstract

In this doctoral thesis, we study the problem of computing the ball of smallest radius enclosing a given set of points in any number of dimensions. Variations of this problem arise in several branches of computer science, such as computer graphics, artificial intelligence, and operations research. Applications range from collision detection for three-dimensional models in video games and computer-aided design, to high-dimensional clustering and classification in machine learning and data mining. We also consider the related and more challenging problem of finding the enclosing ellipsoid of minimum volume. Such ellipsoids can provide more descriptive data representations in the aforementioned applications, and they find further utility in, for example, optimal design of experiments and trimming of outliers in statistics.

The contributions of this thesis consist of practical methods for the efficient solution of these two problems, with a primary focus on problem instances involving a large number of points. We introduce new algorithms to compute arbitrarily fine approximations of the minimum enclosing ball or ellipsoid in general dimensions. In our experimental evaluations, these algorithms exhibit running times that are highly competitive with, and often markedly superior to, those of earlier algorithms from the literature. Moreover, we present a new out-of-core algorithm to compute the exact minimum enclosing ball for massive, low-dimensional point sets residing in secondary storage. In addition to these solution methods, we develop acceleration techniques that can further improve their performance, either by using pruning heuristics to reduce the amount of work performed in each iteration, or by utilizing parallel hardware features of modern processors and graphics processing units. These techniques are also applicable to several existing algorithms.

ISBN 978-91-7485-448-0
ISSN 1651-4238

Abstract

In this doctoral thesis, we study the problem of computing the ball of smallest radius enclosing a given set of points in any number of dimensions. Variations of this problem arise in several branches of computer science, such as computer graphics, artificial intelligence, and operations research. Applications range from collision detection for three-dimensional models in video games and computer-aided design, to high-dimensional clustering and classification in machine learning and data mining. We also consider the related and more challenging problem of finding the enclosing ellipsoid of minimum volume. Such ellipsoids can provide more descriptive data representations in the aforementioned applications, and they find further utility in, for example, optimal design of experiments and trimming of outliers in statistics.

The contributions of this thesis consist of practical methods for the efficient solution of these two problems, with a primary focus on problem instances involving a large number of points. We introduce new algorithms to compute arbitrarily fine approximations of the minimum enclosing ball or ellipsoid in general dimensions. In our experimental evaluations, these algorithms exhibit running times that are highly competitive with, and often markedly superior to, those of earlier algorithms from the literature. Moreover, we present a new out-of-core algorithm to compute the exact minimum enclosing ball for massive, low-dimensional point sets residing in secondary storage. In addition to these solution methods, we develop acceleration techniques that can further improve their performance, either by using pruning heuristics to reduce the amount of work performed in each iteration, or by utilizing parallel hardware features of modern processors and graphics processing units. These techniques are also applicable to several existing algorithms.

Preface

I am glad you have come across my Ph.D. thesis! This thesis follows a "collection-of-papers" format, which means that it is a compilation of already published academic papers, each reprinted as an individual chapter. A list of the six publications included in this thesis is given on page V of this front matter. These papers were written as stand-alone texts and published over the course of several years. Thus, in order to form a more cohesive whole, the thesis additionally contains an introductory part, consisting of five chapters. These chapters serve to provide a comprehensive overview of the research topic and to summarize the contributions of the papers and place them in the context of previous work. Like the papers themselves, this portion of the thesis is for the most part fairly technical. Nonetheless, I have tried my best to make at least the first chapter, in which the motivation behind the research is discussed, as well as the beginning of the second chapter, where an introduction to the minimum enclosing ball problem is given, reasonably easy to follow also for someone with only general knowledge of mathematics and geometry.

The major part of the research presented here was conducted in collaboration with my supervisor Thomas Larsson, who sadly passed away in 2017. Now at the completion of the work we started together, even though I am unable to thank Thomas directly, I would like to express the heartfelt gratitude I have for all the guidance and support he gave me, for the active interest he always took in my doctoral studies, and for the formative influence he has had on my approach to research and academic writing. Furthermore, I want to thank Daniel Andrén for his indispensable assistance during the writing of the final paper of the thesis. A special thank you goes to my supervisor Björn Lisper, not only for his support in my doctoral studies, but also for showing confidence in my abilities throughout the multiple other research projects we have worked on together. I would further like to thank Evan Shellshear at Fraunhofer–Chalmers Centre for the enjoyable collaboration that resulted in Paper D of the thesis. A thank you is also due to Jan Carlson for offering helpful suggestions on an early draft of the first part of the thesis. Finally, I want to thank all my other wonderful colleagues at Mälardalen University for contributing to the friendly and stimulating work environment that I have enjoyed so much being a part of.

This research was funded mainly by the Swedish Foundation for Strategic Research through grant IIS11-0060. Further acknowledgments specific to each of the included papers are given in the respective papers.

List of Publications

Publications Included in the Thesis

Paper A. Thomas Larsson and Linus Källberg. *Fast and Robust Approximation of Smallest Enclosing Balls in Arbitrary Dimensions.* Computer Graphics Forum, 32(5) – Symposium on Geometry Processing 2013.

Paper B. Linus Källberg and Thomas Larsson. *Improved Pruning of Large Data Sets for the Minimum Enclosing Ball Problem.* Graphical Models, 76(6), 2014.

Paper C. Linus Källberg and Thomas Larsson. *Faster Approximation of Minimum Enclosing Balls by Distance Filtering and GPU Parallelization.* Journal of Graphics Tools, 17(3), 2015.

Paper D. Linus Källberg, Evan Shellshear, and Thomas Larsson. *An External Memory Algorithm for the Minimum Enclosing Ball Problem.* The 11th International Conference on Computer Graphics Theory and Applications (GRAPP), 2016.

Paper E. Linus Källberg and Thomas Larsson. *A Filtering Heuristic for the Computation of Minimum-Volume Enclosing Ellipsoids.* The 10th International Conference on Combinatorial Optimization and Applications (COCOA), 2016.

Paper F. Linus Källberg and Daniel Andrén. *Active Set Strategies for the Computation of Minimum-Volume Enclosing Ellipsoids.* Technical report, Mälardalen University (submitted), 2019.

Other Publications

Jan Gustafsson, Andreas Ermedahl, Björn Lisper, Christer Sandberg, and Linus Källberg. *ALF – A Language for WCET Flow Analysis.* The 9th International Workshop on Worst-Case Execution Time Analysis (WCET'09), 2009.

Linus Källberg and Thomas Larsson. *Ray Tracing using Hierarchies of Slab Cut Balls.* Eurographics conference 2010 – short papers.

Thomas Larsson and Linus Källberg. *Fast Computation of Tight-Fitting Oriented Bounding Boxes.* Game Engine Gems 2, 2011.

Niklas Holsti, Jan Gustafsson, Linus Källberg, and Björn Lisper. *Combining Bound-T and SWEET to Analyse Dynamic Control Flow in Machine-Code Programs.* Technical report, Mälardalen University, 2014.

Linus Källberg and Thomas Larsson. *Accelerated Computation of Minimum Enclosing Balls by GPU Parallelization and Distance Filtering.* Proceedings of SIGRAD 2014.

Linus Källberg. *Circular Linear Progressions in SWEET.* Technical report, Mälardalen University, 2014.

Linus Källberg and Thomas Larsson. *Optimized Phong and Blinn–Phong Glossy Highlights.* Journal of Computer Graphics Techniques, 3(3), 2014.

Niklas Holsti, Jan Gustafsson, Linus Källberg, and Björn Lisper. *Analysing Switch-Case Code with Abstract Execution.* The 15th International Workshop on Worst-Case Execution Time Analysis (WCET'15), 2015.

Thomas Larsson, Gabriele Capannini, and Linus Källberg. *Parallel Computation of Optimal Enclosing Balls by Iterative Orthant Scan.* Computers & Graphics, 56, 2016.

Contents

I Thesis **1**

1 Introduction **3**
- 1.1 Outline of the Thesis . . . 7

2 Minimum Enclosing Balls **9**
- 2.1 Introduction . . . 9
- 2.2 Problem Formulations . . . 19
- 2.3 Methods of Solution . . . 22
  - 2.3.1 Exact Algorithms . . . 25
  - 2.3.2 Approximation Algorithms . . . 28
  - 2.3.3 Acceleration Techniques . . . 31
- 2.4 Applications . . . 32
  - 2.4.1 Example: Collision Detection . . . 33
  - 2.4.2 Example: Machine Learning . . . 34
- 2.5 Related Problems . . . 38

3 Minimum-Volume Enclosing Ellipsoids **41**
- 3.1 Introduction . . . 41
- 3.2 Problem Formulations . . . 45
  - 3.2.1 Recovering a Non-Centered Solution . . . 49
- 3.3 Methods of Solution . . . 50
  - 3.3.1 Exact Algorithms . . . 50
  - 3.3.2 Approximation Algorithms . . . 52
  - 3.3.3 Acceleration Techniques . . . 56
- 3.4 Applications . . . 57
  - 3.4.1 Example: Optimal Design of Experiments . . . 58
- 3.5 Related Problems . . . 60

4 Contributions **63**
4.1 Research Process 65
4.2 Summary of Papers 67

5 Conclusion **73**
5.1 Further Studies 74

Bibliography **77**

II Included Papers **93**

6 Paper A:
Fast and Robust Approximation of Smallest Enclosing Balls in Arbitrary Dimensions **97**
6.1 Introduction 99
6.2 Algorithm Description 101
6.2.1 Analysis 104
6.3 Benchmarks 107
6.3.1 First Experiment: Large Problems 108
6.3.2 Second Experiment: Varying d and ϵ 110
6.3.3 Third Experiment: Low Dimensions 112
6.4 Generalization to Other Types of Input 114
6.5 Conclusions 115
6.6 Further Studies 116
Bibliography 117

7 Paper B:
Improved Pruning of Large Data Sets for the Minimum Enclosing Ball Problem **123**
7.1 Introduction 125
7.2 Theoretical Results 127
7.2.1 Viable Balls 127
7.2.2 Improved Bounds 130
7.3 Pruning 134
7.3.1 Distance Caching 136
7.4 Evaluation 139
7.4.1 Large Problem Sizes 141
7.4.2 Three-Dimensional Polygon Meshes 146
7.5 Conclusions 148

7.A Equivalence Result 150
Bibliography 151

**8 Paper C:
Faster Approximation of Minimum Enclosing Balls by Distance Filtering and GPU Parallelization 157**
8.1 Introduction 159
8.2 Distance Filtering 160
8.2.1 Triangle Inequality Filtering 162
8.2.2 Filtering with Accumulated Distances 163
8.2.3 Cauchy–Schwarz Inequality Filtering 164
8.2.4 Simple Performance Model 165
8.3 Farthest-Point Queries on the GPU 166
8.3.1 Tailor-Made Kernels 167
8.3.2 Distance Filtering on the GPU 170
8.4 Auto-Tuning of GPU Kernel Parameters 172
8.5 Experiments 174
8.5.1 Filtering Effectiveness 174
8.5.2 Kernel Performance 177
8.5.3 Moderate to High Dimensions 177
8.5.4 Polygon Meshes 182
8.6 Conclusions 183
Bibliography 184

**9 Paper D:
An External Memory Algorithm for the Minimum Enclosing Ball Problem 191**
9.1 Introduction 193
9.2 Related Work 194
9.3 External Memory Algorithm for MEB 195
9.3.1 I/O Memory Model 195
9.3.2 The Algorithm 196
9.3.3 Filtering Heuristics 197
9.3.4 Analysis 200
9.4 Experiments 201
9.4.1 Discussion of Results 204
9.5 Conclusion 206
Bibliography 207

10 Paper E:
A Filtering Heuristic for the Computation of Minimum-Volume Enclosing Ellipsoids **213**
10.1 Introduction 215
10.2 Related Work 216
10.3 Khachiyan's Algorithm 217
10.4 Approach 219
10.4.1 Practical Considerations 222
10.5 Experiments 223
Bibliography 225

11 Paper F:
Active Set Strategies for the Computation of Minimum-Volume Enclosing Ellipsoids **231**
11.1 Introduction 233
11.2 Minimum-Volume Enclosing Ellipsoids 235
11.2.1 First-Order Algorithms 237
11.3 Active Set Strategies 239
11.3.1 Elimination Strategies 242
11.3.2 Distance Filtering 242
11.4 Experiments 243
11.5 Discussion 246
Bibliography 247

I

Thesis

Chapter 1

Introduction

This thesis studies a class of mathematical problems in which the goal is to fit a simple geometric shape to enclose a given collection of objects as tightly as possible. Such problems arise in numerous applications within computer science and related fields. Two types of enclosing shapes are considered in this thesis, namely, *balls* and *ellipsoids*. The term ball is used here as a generic term to denote a shape defined by a center point and a radius in any number of dimensions. An ellipsoid, in turn, generalizes the concept of an ellipse, which is a plane curve, to general dimensions. Loosely speaking, an ellipsoid is a ball that has been "squeezed" and/or "stretched out" along one or several directions. The *minimum enclosing ball* (MEB) of one or more given objects is the ball of smallest radius such that all the objects are fully contained in it. Similarly, the *minimum-volume enclosing ellipsoid* (MVEE) is the ellipsoid of smallest volume that fully contains the objects. Figure 1.1 shows an example of a 3D model together with its MEB as well as its MVEE.

For the most part, the scope of the thesis is narrowed to the case where the geometry to be enclosed is a finite set of points. Although this assumption might be too restrictive in some situations, it does hold in a wide range of important applications. Furthermore, more complex objects can often be adequately approximated by a finite set of point samples for the purpose of fitting the enclosing shape. In some cases, using such a representation can even yield the correct solution also for the original object. For example, the model in Figure 1.1 is made up of a mesh of polygons. For such models, it suffices to compute the MEB or

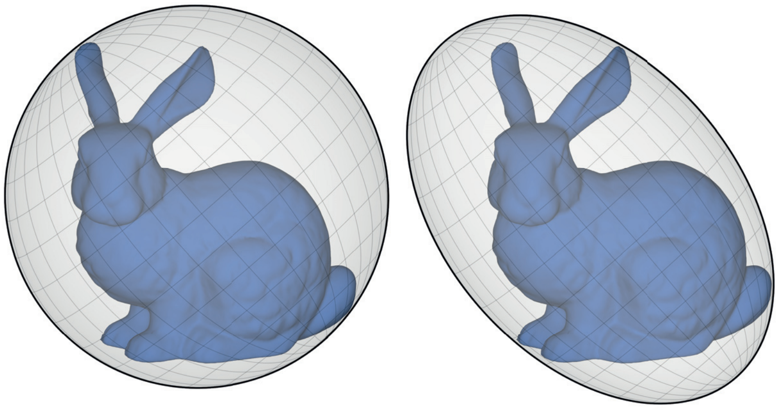


Figure 1.1. The “Stanford Bunny”, shown trapped in its MEB (left) and MVEE (right). The model is provided by the Stanford University Computer Graphics Laboratory: `http://graphics.stanford.edu/data/3Dscanrep/`.

MVEE over only the vertices of the polygons, since this guarantees that also the full polygons are enclosed.

The study of the MEB problem dates back at least as far as the mid-19th century, when English mathematician James Joseph Sylvester posed the two-dimensional instance of the problem [119]. Today, the general-dimensional problem has a rich body of literature and is considered a classical problem in computational geometry. The earliest published results on the MVEE problem date back to the 1930s [13]. The MVEE is also commonly known in the research literature as the Löwner or Löwner–John ellipsoid, named so after the mathematicians Karel Löwner and Fritz John, who both made seminal contributions to the study of the problem [69, 63].

In the plane, there exist a number of applications for finding the smallest circle enclosing a set of points, often revolving around spatial or geographical information [82, 138]. For example, in the minimax facility location problem, the goal is to find a location at which the distance to the most remote neighbor from a list of given locations is as short as possible [45, 140]. This problem arises, for example, when planning the placement of an emergency service such as a fire department, where the

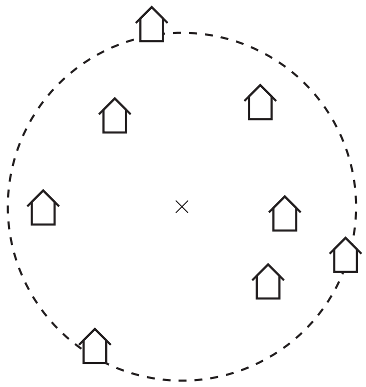


Figure 1.2. The solution to a minimax facility location problem. The cross marks the spot at which the maximum distance to any of the houses is minimized, which might make a suitable location to build an emergency service. In other words, every other location must have a larger distance to at least one of the houses. The location is at the same time the center of the smallest circle enclosing the houses.

primary concern is minimizing the maximum distance to the sites to be serviced by it. Figure 1.2 illustrates that the solution to this problem is simply the center of the smallest circle enclosing the specified locations.

As has already been exemplified in Figure 1.1, enclosing balls and ellipsoids can provide conservative approximations of more complex geometric information. Their simple description can be exploited to speed up various tasks in a number of applications. For example, collision (or interference) detection is the problem of determining whether two or more given objects intersect or not, which is essential in, e.g., robot path planning [99, 101, 107], virtual reality [104], computer-aided design [30, 92], video game engines [47], as well as any type of simulation of the physical world in which moving objects are not supposed to pass straight through each other. If each object in the simulation is enclosed in a simple but tight-fitting shape, then cheap intersection tests between these shapes can be performed before the more intricate geometries of their enclosed objects are examined. If no intersection is found by such a preliminary test, then no further computations are necessary since there can be no collision between the objects. Compared to the MEB, the MVEE can provide more exact approximations of the underlying geometry—the MVEE can be said to contain less “air” than the MEB does. This can help to reduce the number of “false alarms” in these applications, that is, occurrences where the enclosing shapes intersect but the objects do not. This advantage must, however, be weighed against the drawback that the more complex description of an ellipsoid leads to computationally more expensive intersection tests. Another type of interference detection arises in the rendering of three-dimensional scenes, where the visibility of objects in the scene from the current viewpoint needs to be assessed [20, 8, 70]. If the enclosing shape of an object can be determined to be invisible—that is, it does not “collide” with the current field of view—then the whole object must also be invisible. It then does not need to be considered further by the rendering algorithm.

An example of an application in which higher-dimensional MEBs and MVEEs find applicability is an image database. A common type of query in such a database is to find the image or images that most closely resemble a given reference image, without the use of manually entered search terms. One approach to measuring the similarity between different images is to first encode each of them as a sequence of numerical values based on their pixel data. This representation can be interpreted as a point in a high-dimensional space, in which points that correspond to similar images tend to end up close together [85]. Processing the database query then amounts to searching this space for the points that are closest to the point corresponding to the reference image. In large databases, this search can be accelerated by letting clusters of nearby points in the database be enclosed in, e.g., tight-fitting, high-dimensional balls [79, 26]. During the search, entire such clusters of points can then be dismissed if the distance to the surface of the ball enclosing a cluster is greater than the distance to the nearest points found so far; in such cases, no point inside of the ball can possibly make a candidate to be among the nearest points. Other applications involving higher-dimensional problem instances include classification [103, 22, 122, 132, 123] and cluster analysis [111], which are both important tasks in, e.g., machine learning and data mining. Specific applications for the MVEE are also found in statistics [48, 128, 5].

Although it might not appear, judging from Figures 1.1 and 1.2, too challenging to determine a MEB or MVEE, it turns out to be far from trivial when there are many dimensions or very complex geometries involved. In fact, finding the exact solution is often not practically feasible, which means that approximation methods must be used. The goal of this thesis is to present new and efficient methods to compute the MEB and MVEE for large point sets in general dimensions. The results include new exact and approximating solution methods, several speed-up techniques that can be applied to these and existing methods, as well as new theoretical results.

1.1 Outline of the Thesis

The thesis is organized into two parts. The main scientific contributions are presented in the second part, in the form of six research papers that are reprinted in their entirety as separate chapters. The first five of these papers, labeled A–E, have already been published in peer-reviewed venues. At the time of completion of this thesis, the sixth paper, Paper F, is published in the form of a technical report.

In the remainder of this introductory part of the thesis, deeper background and motivation is first given for the problems tackled in these publications. Chapter 2 is devoted to the MEB problem. Here, an introduction to the problem as well as formal problem statements are given, followed by a fairly exhaustive survey of relevant research literature. Then follows a more detailed discussion of a few selected applications for the MEB problem, as well as a discussion of related problems. Chapter 3 discusses the MVEE problem following an analogous structure. Chapter 4 summarizes the papers included in the thesis and highlights how they are connected to each other. Finally, Chapter 5 concludes this first part and suggests some directions for future studies.

Chapter 2

Minimum Enclosing Balls

2.1 Introduction

As a gentle introduction, we begin by considering the problem of finding the minimum enclosing circle (MEC) of a given set of points in the plane. To derive some important properties of this problem in a fairly informal way, we start from the most trivial problem instances and then build toward more challenging cases. First we note that the MEC of a single point is the somewhat degenerate circle having the point in question as its center and a radius of zero. This special case is of no particular interest, and it will be assumed henceforth that there are at least two points involved.

When exactly two points are given, the smallest circle must have its center at the midpoint of the line segment connecting them, with both points touching its circumference, as in Figure 2.1a. Clearly, the radius cannot be made any smaller in this position, and moving the center in any direction must create a larger distance to at least one of the points, thereby enforcing a larger radius.

When there are exactly three points, provided they are not located along a straight line, a circle passing through all of them can be constructed using a method that goes back to Euclid[1]: Draw the perpendicular bisectors of two sides of the triangle formed by the points; the point of intersection of these bisectors then gives the center of the circle. (Any

[1] *Elements*, Book IV, Proposition 5.

two bisectors can be chosen, as all three of them intersect in the same point.) It might seem apparent that this circle would be the smallest possible circle enclosing the points. However, this is the case only under specific circumstances, namely, when the points form an acute or right triangle. As is exemplified in Figure 2.1b, if the triangle has only acute angles, then the center defined in this way falls strictly on the inside of the triangle. In analogy to the previous two-point case, moving the center in any direction from this position will create a larger distance to at least one of the points; the center must therefore be the sought optimal center. If the triangle has a right angle, then the center ends up at the midpoint of the edge opposite that angle. This situation is essentially the same as if the endpoints of this edge were the only points given. Again, this center must be optimal. If one angle is obtuse, on the other hand, then the center ends up outside of the triangle, as in Figure 2.1c. An even smaller enclosing circle can then be constructed by disregarding the point at the obtuse corner, and simply use the circle defined by the remaining two points, as before. The ignored point will then appear in the interior of this circle; see Figure 2.1d. In the case where all three angles are acute, it is not possible to obtain a smaller enclosing circle in this way, since any circle defined by only two points will not enclose the third point.

Due to the lack of flexibility in the curvature of a circle, it is in general not possible to define a circle that passes through more than three points, unless the points are configured in such a way that a circle drawn through only three of them automatically passes through the remaining points. Nevertheless, we have already seen that a circle defined by only two points can be the MEC of more than two points. This property can be generalized to say that a circle defined by a subset of either two or three points using the rules above, can give the MEC of a set of four or more points. As long as such a circle encloses also the remaining points, then it must be the MEC of the full point set since, as was noted before, any adjustment of the center would increase the distance to at least one of the defining points on the circumference. Furthermore, even if some of the remaining points happen to also end up on the circle circumference, they are not necessary to define the circle. We call such a subset of two or three points—whose MEC has all the points of the subset on its perimeter and is at the same the MEC of the full point set—a *support set*, and refer to the points themselves as *support points*. We further add the requirement of the support set being *inclusion-minimal*, meaning that

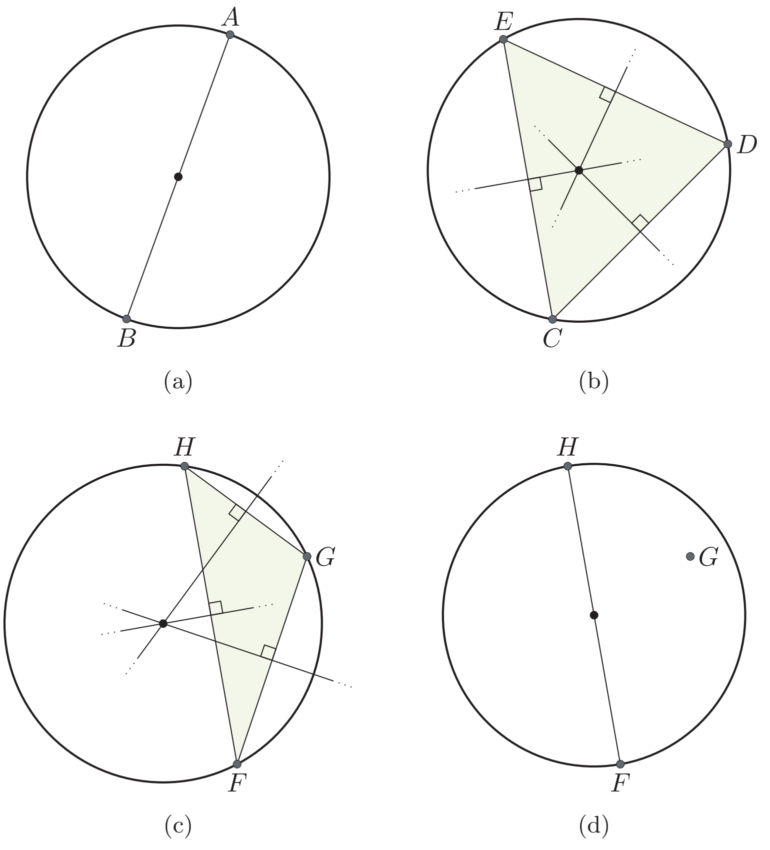


Figure 2.1. (a) The smallest enclosing circle of two points has its center midway between the points. (b) & (c) The circle passing through three points has its center at the intersection of the perpendicular bisectors of the sides of the triangle spanned by the points. In (b), the triangle CDE has only acute angles, which implies that the circle is optimal. In (c), the triangle FGH has an obtuse angle at G, which causes the center to end up outside of the triangle. A smaller enclosing circle is then given by setting the center to the midway point between F and H, which leaves G on the inside of the circle as is shown in (d).

every point in it must be essential to define the MEC. Thus, for example, if three points on the circumference form a right triangle, then the point at the right-angled corner is non-essential and therefore not part of the support set. A few example problem instances involving more than three points are given in Figure 2.2.

A simple proof by contradiction shows that the MEC must be unique: If there were two distinct MECs, then an even smaller circle could be drawn around their intersection, and this circle would too enclose all of the points. (The support set, however, is not necessarily unique, as is exemplified in Figure 2.2c.) Although in this thesis we are mainly concerned with finite point sets, it should also be remarked that any *bounded* point set has a unique MEC [17]. It is straightforward to extend the above discussion to infinite such point sets, to realize that the same properties regarding the support points must hold.

A Simple Geometric Procedure

It is evident that the task of computing the MEC can be approached as the combinatorial problem of determining which points should be included in the support set. A straightforward method would be to generate all circles passing through two or three points, filter out the ones not enclosing all of the points, and keep the smallest circle of those remaining. However, such an exhaustive enumeration would be practical only for fairly small point sets: If there are n given points, then there are

$$\binom{n}{2} + \binom{n}{3} = \frac{n(n-1)}{2} + \frac{n(n-1)(n-2)}{6} \tag{2.1}$$

subsets containing two or three points. If, for example, $n = 100$, then the number of such subsets is 166,650. Bear in mind that for each generated circle, the remaining $n-2$ or $n-3$ points in the set must be tested for containment in the circle to determine if it is indeed an enclosing circle. If $n = 1000$, then the number of possible support sets grows to 166,666,500.

We now sketch a more efficient procedure, and apply it to an example point set, shown in Figure 2.3a. First, an initial circle is created by selecting an arbitrary center point—for example, one of the given input points—and setting the radius to the distance to the farthest point. As can be seen in Figure 2.3b, this clearly gives an enclosing circle.

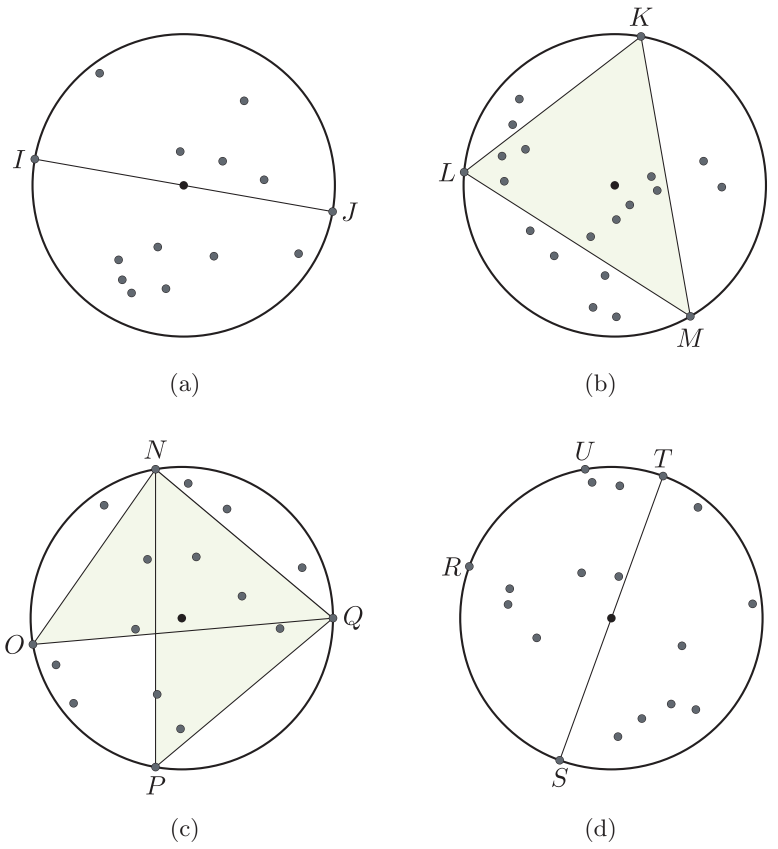


Figure 2.2. Example point sets and their smallest enclosing circles. All points on the circle circumferences are labeled, and those that form support sets are highlighted with their connecting line segment or triangle. In (a) and (b), the support sets contain two and three points, respectively. In (c), there are two alternative support sets, namely, $\{N, O, Q\}$ and $\{N, P, Q\}$. No other combination of points on the circumference span a diameter or a triangle containing the center. In (d), only S and T form a support set: No other pair of points span a diameter; moreover, in any set of three points containing both S and T, the third point is clearly redundant, and the only other two possible sets of three points form triangles not containing the center.

Furthermore, it is clear that a smaller circle can be obtained by shifting the center in the direction of the point determining the current radius. In Figure 2.3c, the center has been shifted—and the radius shrunk accordingly—by the largest permissible amount without any points passing outside of the circle. In this configuration, there are now two points touching the perimeter.

The circle can now be further reduced by moving its center toward the position midway between these two points. As this makes the center travel along the perpendicular bisector of the line segment connecting the points, it will remain equidistant to them, so that both points can be kept fixed to the circle as it shrinks. Now, either it is possible to move the center all the way to the midway point without any points passing outside of the circle, or its movement is stopped by a third point ending up on the circumference. In the first case, the solution has been found along with a support set of two points, according to the earlier arguments. In the second case, which is shown in Figure 2.3d, there are now three points on the boundary, and the center cannot be moved further without the third point ending up on the outside of the circle. As discussed above, if these points form an acute or right triangle, then the MEC has been found along with a support set of three points. Otherwise, the circle can be further reduced by having it let go of the point at the obtuse angle and then shifting its center toward the point halfway between the two remaining points, that is, by continuing the procedure from the previous step.

Notice that from step two onward, each step of this procedure amounts to finding the smallest possible circle having two points fixed on its perimeter while enclosing all of the remaining points. In each step, if a third point interrupts the movement of the center, this third point is uniquely determined by the two fixed points. Furthermore, since the circle always shrinks with each step, the same pair of points can never be repeated. Thus, we can conclude that it cannot take more than $n(n-1)/2$ steps, which is same as the number of possible pairs of points, until the MEC has been found with this procedure. Compared to the brute-force approach, the number of possible support sets evaluated has thus been reduced by at least $n(n-1)(n-2)/6$, the second term of (2.1). While this is a major improvement, however, significantly more efficient methods are known, as will be seen in Section 2.3.

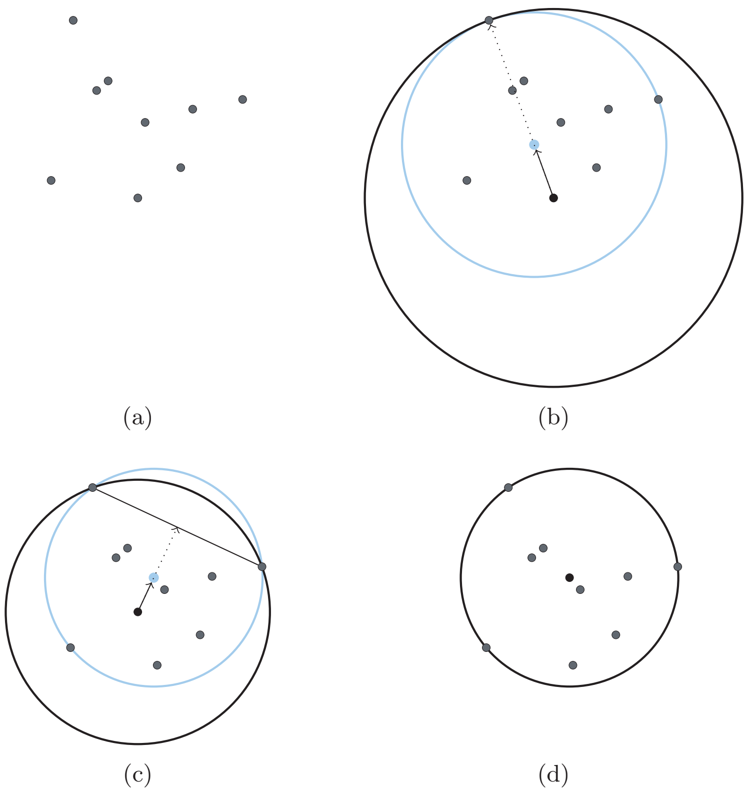


Figure 2.3. A simple geometric procedure to compute the smallest enclosing circle, applied to an example point set. In (b) and (c), the dotted arrow indicates the attempted movement of the center, and the solid arrow indicates the maximum permissible movement without letting any points pass to the outside of the circle as it shrinks. The shaded circle shows the maximally shifted and shrunk circle, which becomes the starting point of the next step.

Generalizing to d Dimensions

The problem can be generalized to any number of dimensions if the given points as well as the center point are allowed to be defined in d dimensions. In this thesis, we call the enclosing shape a ball whenever d is not specified, and use the terms circle and sphere for the cases $d = 2$ and $d = 3$, respectively. In dimensions above three, the term hypersphere is sometimes used in the literature. (Technically, the term ball generally refers to the whole enclosed volume, whereas the term (hyper-)sphere refers only to the surface of a ball. The term circle, on the other hand, usually refers only to the circumference, whereas the term disc can be used to refer to the area contained in the circle. However, these semantic differences bear no relevance to the problems discussed in this thesis.) As in the two-dimensional case, the d-dimensional MEB exists and is unique, also for infinite, bounded point sets [17].

In what follows, we let a ball with center $c \in \mathbb{R}^d$ and radius $r \in \mathbb{R}$ be denoted by $\mathcal{B}_{c,r}$, i.e.,

$$\mathcal{B}_{c,r} := \{x \in \mathbb{R}^d : \|x - c\| \leq r\}, \tag{2.2}$$

where $\|\cdot\|$ is the Euclidean norm. Furthermore, we denote the set of n given points by $\mathcal{P} := \{p_1, \ldots, p_n\} \subset \mathbb{R}^d$, and say that $\mathcal{P}$ is enclosed by $\mathcal{B}_{c,r}$ if $\mathcal{B}_{c,r} \supset \mathcal{P}$. Moreover, we let the MEB of $\mathcal{P}$ be denoted by $\mathrm{MEB}(\mathcal{P})$ and define c^* and r^* to satisfy $\mathcal{B}_{c^*,r^*} = \mathrm{MEB}(\mathcal{P})$. We now proceed to generalize the properties derived above for the support set to the d-dimensional case.

Let $\mathcal{S}$ be a subset of m points of $\mathcal{P}$, renamed as $s_1, \ldots, s_m$. A ball $\mathcal{B}_{c,r}$ having $\mathcal{S}$ on its boundary is given by a solution (c, r) to the system of equations

$$(s_i - c)^{\mathrm{T}}(s_i - c) = r^2, \quad i = 1, \ldots, m. \tag{2.3}$$

This system is in essence a linear system of only $m - 1$ equations in d variables: If we subtract the m-th equation from the other equations, they can be rewritten as

$$(s_i - s_m)^{\mathrm{T}} c = (s_i^{\mathrm{T}} s_i - s_m^{\mathrm{T}} s_m)/2, \quad i = 1, \ldots, m - 1.$$

A solution c to this system[2] determines a unique nonnegative value of r, which is given by substituting c into any of the original equations

[2]Notice that this system can alternatively be written as $(c - s_m)^{\mathrm{T}}(s_i - s_m) = (s_i - s_m)^{\mathrm{T}}(s_i - s_m)/2$, $i = 1, \ldots, m - 1$, which expresses a generalization of the Euclidean rule used earlier to d dimensions and m points.

in (2.3) and taking a square root. From this formulation, it is clear that if the points $s_1, \ldots, s_m$ are affinely independent, then (2.3) must have solutions, because this is equivalent to the vectors $s_i - s_m$, $i = 1, \ldots, m-1$, being linearly independent. On the other hand, if the points are affinely dependent, then either the system is inconsistent—i.e., no ball exists that passes through all m points—or one or more equations are superfluous—i.e., the corresponding points already lie on any ball given by the other equations. Thus, the property from before that a circle can be drawn through at most three points generalizes to the property that a ball can be defined through at most $d+1$ points. For ease of discussion, we henceforth assume that the points of $\mathcal{S}$ are affinely independent.

If $m = d+1$, then (2.3) has a unique solution, and if $m < d+1$, there are infinitely many solutions. In the latter case we are interested in the *smallest* ball passing through the points of $\mathcal{S}$. The affine hull of $\mathcal{S}$ is then a lower-dimensional hyperplane, and it can easily be shown that the smallest ball having $\mathcal{S}$ on its boundary must have its center in the same hyperplane: Otherwise, there would exist a parallel hyperplane separating the center from the points; the center could then be translated along the plane normal to be closer to all of the points, and this would contradict that the current ball is the smallest. Notice that in this case, the ball can be seen as an $(m-1)$-dimensional ball, defined by $m < d+1$ points on its surface, that simply extends into the additional $d-m+1$ empty dimensions. We have already seen that when $d = m = 2$, the center of the smallest circle must lie on the line passing through the two points—that is, their affine hull.

To ensure that c ends up in the affine hull of $\mathcal{S}$, we augment the system (2.3) with the $d+1$ equations

$$c = \sum_{i=1}^{m} v_i s_i, \tag{2.4}$$

$$\sum_{i=1}^{m} v_i = 1 \tag{2.5}$$

and the variables $v_i \in \mathbb{R}$, $i = 1, \ldots, m$. It is straightforward to rewrite the system (2.3)–(2.5) as a linear system with $m-1$ equations in only the variables $v_1, \ldots, v_{m-1}$ [56]. This system has a unique solution, with the ball $\mathcal{B}_{c,r}$ given by $c = \sum_{i=1}^{m-1} v_i p_i$ and $r = \|s_1 - c\|$.

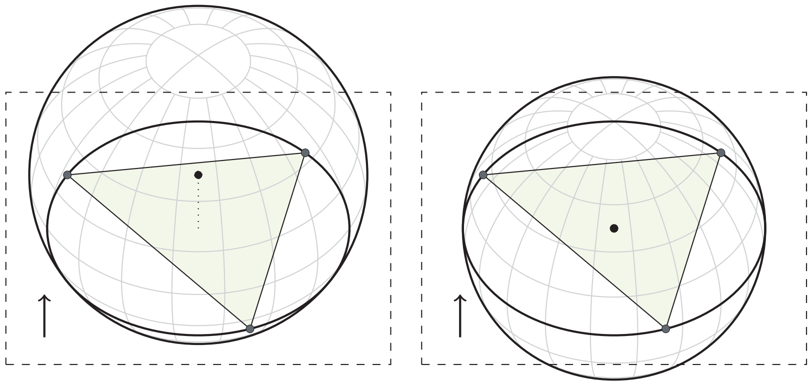


Figure 2.4. Two three-dimensional balls defined to pass through three given points. The plane that is the affine hull of the points is indicated with a dashed rectangle, and its normal vector is shown as an arrow. Also shown is the triangle that is the convex hull of the points, as well as the circle where both balls intersect the affine hull. In (a), there is some distance between the ball center and the affine hull; thus, the ball is not the smallest possible. Shifting the center toward the plane along the plane normal allows smaller balls to be defined while maintaining all three points on the boundary. The ball in (b) is the smallest possible one, as its center coincides with the affine hull. It is also the MEB of the points, as its center lies in their convex hull.

Just as the circle passing through three points is not necessarily their MEC, however, the ball $\mathcal{B}_{c,r}$ given by (2.3)–(2.5) is not necessarily the MEB of $\mathcal{S}$. A similar argument as before shows that $\mathcal{B}_{c,r} = \text{MEB}(\mathcal{S})$ if and only if c lies in the *convex hull* of $\mathcal{S}$, i.e., if $v_i \geq 0$ for $i = 1, \ldots, m$: Otherwise, there would exist a hyperplane separating c from $\mathcal{S}$, which means that c could be moved along the normal of this plane to be closer to all the points of $\mathcal{S}$—a contradiction. This generalizes the statement from before that a circle passing through three points is their MEC only if its center lies inside the triangle spanned by the points. As before, if also $\text{MEB}(\mathcal{S}) \supset \mathcal{P}$ holds, then $\text{MEB}(\mathcal{S}) = \text{MEB}(\mathcal{P})$, and $\mathcal{S}$ is a support set also of $\text{MEB}(\mathcal{P})$. On the other hand, if c does not lie within the convex hull, then at least one of the points s_i is not a support point of

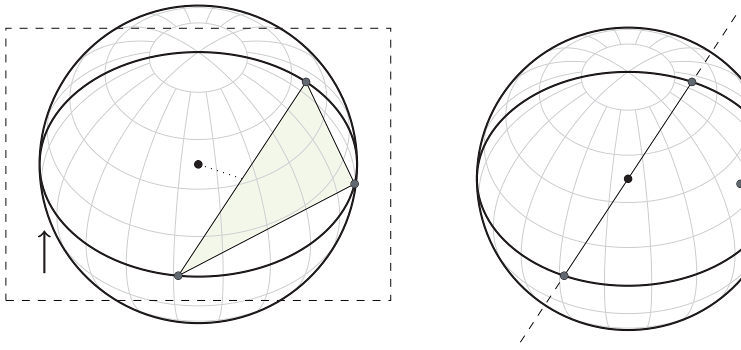


Figure 2.5. (a) The smallest three-dimensional ball passing through three points configured similarly to those of Figures 2.1c and 2.1d. The center lies in the affine hull of the points, but since it has some distance to the convex hull, the ball cannot be the MEB of the points. As in two dimensions, the MEB, shown in (b), has its center midway between two of the points, and has the third point strictly inside of its boundary. The affine hull of the two support points is indicated with a dashed line, and their convex hull is indicated with a solid line. The circle where the MEB intersects the plane of all three points is also highlighted.

MEB($\mathcal{S}$). (Depending on how $\mathcal{S}$ was selected, however, that point might yet be a support point of MEB($\mathcal{P}$).) Two further examples illustrating the considerations discussed in this section, this time with $d = m = 3$, are given in Figures 2.4 and 2.5.

2.2 Problem Formulations

Since the smallest possible radius of an enclosing ball with a given center c is determined by the largest distance from c to any point in $\mathcal{P}$, the optimal center c^* is given by the solution to the optimization problem

$$(\mathbf{P}'_{\mathrm{B}}) \quad \min_c \max_{i=1}^n \|p_i - c\|. \tag{2.6}$$

The optimal radius r^* is then given directly by $\max_{i=1}^n \|p_i - c^*\|$. The objective function (2.6) is clearly convex, as it is the point-wise maximum of the convex functions $\|p_i - c\|$, $i = 1, \dots, n$. However, it is not differentiable. We can obtain an equivalent but more convenient problem formulation by squaring all the distances and introducing the squared radius as an explicit variable s, constrained to satisfy $s \geq \max_{i=1}^{n} \|p_i - c\|^2$:

$$(\mathbf{P}_\mathrm{B}) \quad \min_{c,s} \quad s \tag{2.7}$$

$$\text{s.t.} \quad \|p_i - c\|^2 \leq s, \quad i = 1, \dots, n. \tag{2.8}$$

In addition to being convex, this formulation has continuously differentiable objective and constraint functions. It is a quadratically constrained linear program with the same quadratic term, $c^\mathrm{T} c$, in all of the constraints. Thus, it could alternatively be formulated as a linear program with a single additional, quadratic constraint [44], or as a convex quadratic program with all constraints linear [61]. As will be seen in Section 2.3, its resemblance to a linear problem makes the MEB problem amenable to solution methods analogous to some methods for linear programming.

Introducing multipliers $u := (u_1, \dots, u_n)^\mathrm{T} \in \mathbb{R}^n$, we define the Lagrangian of problem $(\mathbf{P}_\mathrm{B})$ as

$$L(c, s, u) := s + \sum_{i=1}^{n} u_i \big(\|p_i - c\|^2 - s\big). \tag{2.9}$$

For $u_i \geq 0$, $i = 1, \dots, n$, it is immediate that $L(c, s, u) \leq s$ for any (c, s) satisfying (2.8). A lower bound on the optimum s^* of $(\mathbf{P}_\mathrm{B})$ can thus be obtained by minimizing $L(c, s, u)$ over (c, s). It is evident that $L(c, s, u)$ is bounded below as a function of s only when

$$\sum_{i=1}^{n} u_i = 1, \tag{2.10}$$

in which case s vanishes from (2.9). Then $L(c, s, u)$ is minimized as a function of c at a stationary point $\bar{c}$, i.e.,

$$\nabla_c L(\bar{c}, s, u) = \sum_{i=1}^{n} 2u_i (p_i - \bar{c}) = 0,$$

which can be rewritten as

$$\bar{c} = \sum_{i=1}^{n} u_i p_i. \tag{2.11}$$

In summary,

$$\begin{aligned} s^* &\geq \min_{c,s} L(c,s,u) && (2.12)\\ &= \sum_{i=1}^{n} u_i \|p_i - \bar{c}\|^2 && (2.13)\\ &= \sum_{i=1}^{n} u_i p_i^{\mathrm{T}} p_i - \Big(\sum_{i=1}^{n} u_i p_i\Big)^{\mathrm{T}} \Big(\sum_{i=1}^{n} u_i p_i\Big), && (2.14) \end{aligned}$$

where (2.10) and (2.11) are used in (2.13) and (2.14), respectively. The tightest possible lower bound on s^* is obtained by maximizing (2.14), which is a concave function of u. This leads to the Lagrangian dual of problem ($\mathbf{P}_{\mathrm{B}}$):

$$\begin{aligned} (\mathbf{D}_{\mathrm{B}}) \quad \max_u \quad & \sum_{i=1}^{n} u_i p_i^{\mathrm{T}} p_i - \Big(\sum_{i=1}^{n} u_i p_i\Big)^{\mathrm{T}} \Big(\sum_{i=1}^{n} u_i p_i\Big), && (2.15)\\ \text{s.t.} \quad & \sum_{i=1}^{n} u_i = 1, && (2.16)\\ & u_i \geq 0, \quad i = 1, \ldots, n. && (2.17) \end{aligned}$$

Problem ($\mathbf{P}_{\mathrm{B}}$) clearly satisfies Slater's condition [21], which requires that there exist strictly feasible solutions: Any enclosing ball $\mathcal{B}_{c,r}$ of $\mathcal{P}$ with no points touching its boundary corresponds to such a solution. Slater's condition implies that strong duality holds between ($\mathbf{P}_{\mathrm{B}}$) and ($\mathbf{D}_{\mathrm{B}}$), i.e., that their optima coincide. This, together with convexity, implies that the Karush–Kuhn–Tucker (KKT) optimality conditions are both necessary and sufficient. Denoting by u^* an optimal solution to problem ($\mathbf{D}_{\mathrm{B}}$), we can state the complete KKT conditions as

$$\begin{aligned} u_i^* \geq 0, \quad i = 1, \ldots, n, && (2.18)\\ \|p_i - c^*\|^2 - s^* \leq 0, \quad i = 1, \ldots, n, && (2.19)\\ \sum_{i=1}^{n} u_i^* = 1, && (2.20)\\ c^* = \sum_{i=1}^{n} u_i^* p_i, && (2.21)\\ u_i^*(\|p_i - c^*\|^2 - s^*) = 0, \quad i = 1, \ldots, n. && (2.22) \end{aligned}$$

Note that strong duality implies that the inequality (2.12) becomes equality for $u = u^*$. This, in turn, implies that $(c, s) = (c^*, s^*)$ minimizes $L(c, s, u^*)$, since

$$\begin{aligned} s^* &= \min\nolimits_{c,s} L(c, s, u^*) \\ &\leq L(c^*, s^*, u^*) \\ &\leq s^*, \end{aligned}$$

where both inequalities must be equalities. From this, conditions (2.20)–(2.22) follow.

Conditions (2.18), (2.20), and (2.21) together give that c^* must be a convex combination of the points of $\mathcal{P}$. It further follows from (2.22) that only points on the boundary of the MEB can have non-zero weights in this convex combination. Carathéodory's theorem [24], in turn, gives that no more than $d + 1$ such points are necessary. Thus, we have confirmed the statement of the previous section that the MEB must have its center in the convex hull of up to $d + 1$ points on its boundary.

Any feasible solution u to problem ($\mathbf{D}_\mathrm{B}$) can be translated into a ball $\mathcal{B}_{c,r}$ by

$$c = \sum_{i=1}^{n} u_i p_i, \tag{2.23}$$

$$r = \sqrt{\sum_{i=1}^{n} u_i \|p_i - c\|^2}. \tag{2.24}$$

Note that (2.24) simply expresses the square root of the dual objective (2.15). Thus, for $u = u^*$, c and r given by these equations coincide with c^* and r^*, respectively, and $\mathcal{B}_{c,r} = \mathrm{MEB}(\mathcal{P})$. For any feasible but suboptimal u, it follows from weak duality that $r < r^*$, which means that $\mathcal{B}_{c,r}$ has points of $\mathcal{P}$ outside of it.

2.3 Methods of Solution

The earliest known algorithm for the MEB problem appears in a publication from 1860 by Sylvester [120], in which a procedure is outlined for computing the minimum enclosing circle of points in the plane. The procedure, which is attributed to Peirce, is in essence identical to the one sketched in Section 2.1 of this thesis. The same procedure was then rediscovered 25 years later by Chrystal [32].

Since these early works, numerous methods have been proposed in the literature. These can be categorized in a number of ways, for example, as fixed- versus general-dimensional, exact versus approximate, or direct versus iterative. Many algorithms can also be naturally characterized as either primal or dual [62]. An algorithm is called primal if it arrives at the optimum through a sequence of primal feasible candidate solutions $(c, r) = (c^k, r^k)$, $k = 1, 2, \ldots$ Geometrically, this means that it starts out with a trial ball that fully encloses all of the given points, and then works to "deflate" this ball—not necessarily in a monotonic fashion—to the optimal fit without letting any points pass outside of it. The Peirce–Chrystal algorithm above belongs to this category.

A dual algorithm, in contrast, generates a sequence of dual feasible solutions $u = u^k$, $k = 1, 2, \ldots$, where the current iterate u is either represented as an explicit variable or is implicit in the current values of c and r through (2.23) and (2.24), respectively. Interpreted geometrically, such an algorithm starts out with an undersized trial ball, which it then proceeds to "inflate" (possibly non-monotonically) until all the points are enclosed, which implies that the ball is optimal.

A simple procedure for the planar case by Elzinga and Hearn [45] falls into the latter category. Similarly to the Peirce–Chrystal method, it generates a sequence of candidate support sets, each containing two or three points, along with the sequence of circles passing through all the points in each set. Unlike in the Peirce–Chrystal method, however, each support set is computed from the previous one in a manner that ensures that its corresponding circle is also its smallest enclosing circle. This means that every circle except the last one (which is the optimum) is the smallest enclosing circle of a strict subset, and thus too small to cover the whole point set. Each candidate support set is generated by selecting a point outside of the current circle and then computing the smallest circle enclosing the union of the current support set and the new point. An example computation is given in Figure 2.6. The new circle is necessarily larger than the previous one, and must have the new point on its circumference. If any of the previous support points are now left in the interior of the new circle, they are evicted from the set. Notice that this occurs in the first and last updates shown in Figure 2.6.

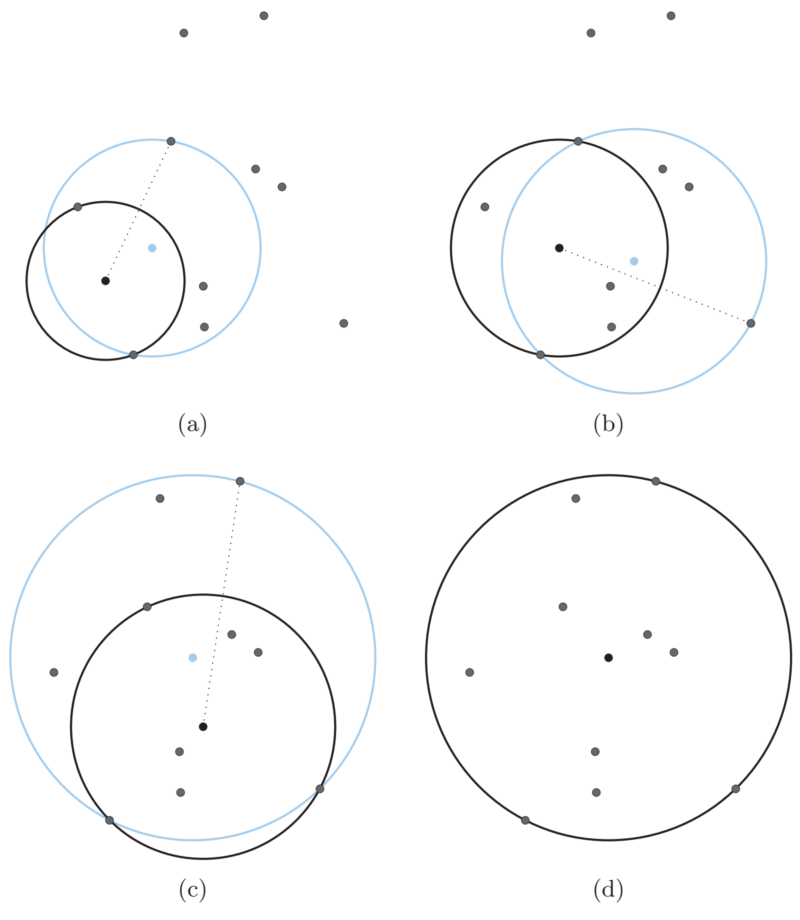


Figure 2.6. The last four iterates of the dual algorithm by Elzinga and Hearn [45] invoked on an example point set. In each step, a dotted line is drawn to the next point chosen to enter the candidate support set, and the shaded circle gives the next circle, which is the smallest circle enclosing the current support set and the new point.

2.3.1 Exact Algorithms

It is clear that both the Peirce–Chrystal and Elzinga–Hearn algorithms must terminate in a finite number of steps: The former generates monotonically shrinking circles and the latter generates monotonically growing circles, which means that none of the candidate support sets defining these circles can be repeated. As was noted in Section 2.1, the Peirce–Chrystal method finishes in $O(n^2)$ steps. Since each step takes $\Theta(n)$ time, the overall asymptotic time complexity is $O(n^3)$. It is fairly easy to realize that any point ending up on the circumference of the current trial circle sometime during this algorithm must be an extreme point of the convex hull of the point set, since the circle is itself a convex shape fully enclosing the convex hull. Taking this fact into account gives the refined bound $O(h^2 n)$, where h is the total number of extreme points. Although h can be as large as n in the worst case, for many problem instances it is much smaller [15].

To analyze the number of steps required by the Elzinga–Hearn method, we first note that each new candidate circle encloses at least one new point that was not enclosed by the previous circle. However, this point might be evicted from the support set in a future iteration, and might eventually end up outside of the candidate circle again. (Notice in Figure 2.6 that one point that is a support point in (a) is left outside of the circle in (c).) Thus, the same point can be selected more than once. A coarse upper bound on the number of steps is given by (2.1), the number of possible support sets, which is $O(n^3)$. Since selecting the next point takes $O(n)$ time, the total time complexity is $O(n^4)$. If the *farthest* point from the current circle, as opposed to an arbitrary point outside of it, is chosen in each iteration, all candidate support sets will again be constructed only from extreme points of the convex hull of the points. Then the time complexity can instead be expressed as $O(h^3 n)$.

In [46], Elzinga and Hearn derive the primal and dual problem formulations ($\mathbf{P}_\mathrm{B}$) and ($\mathbf{D}_\mathrm{B}$) and present an algorithm resembling a generalization of their planar procedure to d dimensions. Each iterate consists of a subset of $d+2$ points and their MEB. The subproblem of computing the MEB of each subset is solved by applying the simplex method of quadratic programming to the corresponding dual formulation. Since at most $d+1$ of the points can have a positive weight in the solution, one point with zero weight can be replaced in each step by a point outside of the current MEB. As in the planar procedure, the radii of the candidate MEBs must be monotonically increasing, and because there are a finite number of candidate support sets, the algorithm is guaranteed to terminate.

Shamos and Hoey [109] noted that the optimal circle, if not defined by two points, must have its center at a vertex of the farthest-point Voronoi diagram. Using this data structure, which takes $O(n \log n)$ time to construct, it only takes $O(n)$ time to either locate the two points determining the optimal circle or to find the vertex at which the optimal center must be lie. The smallest enclosing circle is thus computed in $O(n \log n)$ time, a clear improvement over the previous methods.

In 1983, the worst-case time complexity of the MEB problem in d dimensions was determined to be linear in n, when Megiddo [88, 89] and, independently, Dyer [43, 44] presented a multidimensional prune-and-search technique by which linear programming problems can be solved in deterministic linear time (in the number of constraints). The same method can be adapted to several other optimization problems, including the MEB problem. Although this was a major theoretical breakthrough, however, the method is practical only for a very small number of dimensions: The hidden constant in the time complexity of the original method is $2^{O(2^d)}$, which was later improved to 3^{d^2} by Dyer [44] and Clarkson [33].

It was later shown that more practical, *randomized* algorithms can solve the problem in *expected* $O(n)$ time in fixed dimension, where the expectation is over the random choices made internally by the algorithm. Such an algorithm by Welzl [139], which generalizes an earlier algorithm for linear programming by Seidel [108], is easily described as a short recursive procedure. However, its dependence on d is $(d+1)(d+1)!$, which limits its applicability in practice to dimensions up to, say, $d = 10$. A similar algorithm given by Sharir and Welzl [110], and later analyzed more fully by Matoušek et al. [87], achieves a subexponential dependence on d. The algorithm is presented in an abstract way that relies less on the geometry of the problem, which enables it to be applied to a broader class of optimization problems, provided that a few concrete subroutines are available for the specific problem at hand. This class of so called *LP-type problems* further includes linear programming (as the name suggests), as well as the MVEE problem, among others. For the MEB problem, an algorithm by Gärtner [55] is used as a concrete subroutine to compute the MEB of subsets of at most $d + 2$ points. A randomized algorithm for linear programming by Clarkson [34] can also be modified to fit into the LP-type framework. Chazelle and Matoušek [31] applied derandomization techniques to Clarkson's algorithm to achieve deterministic linear time complexity (but with a worse dependence on d). A summary of subsequent work along this line of research, as well as a recent result, are given by Chan [28].

Using Welzl's algorithm as a subroutine, Gärtner [56] gives an algorithm that can be viewed as a direct d-dimensional generalization of the dual Elzinga–Hearn method. This means that it computes a sequence of candidate support sets $\mathcal{S}^k \subseteq \mathcal{P}$ and balls $\mathcal{B}^k := \text{MEB}(\mathcal{S}^k)$, $k = 1, 2, \dots,$ until the true support set of MEB($\mathcal{P}$) has been found. In iteration k, the point in $\mathcal{P}$ that is farthest from the center of $\mathcal{B}^k$ is located. If this farthest point, call it q^k, is found on the inside of $\mathcal{B}^k$, then $\mathcal{B}^k$ must be the optimum (i.e., $\mathcal{B}^k = \text{MEB}(\mathcal{P})$), and the procedure is terminated. Otherwise, $\text{MEB}(\mathcal{S}^k \cup \{q^k\})$ is computed using Welzl's algorithm to become the next ball $\mathcal{B}^{k+1}$. The support set of $\mathcal{B}^{k+1}$ becomes the next candidate support set $\mathcal{S}^{k+1}$, and satisfies $q^k \in \mathcal{S}^{k+1}$ and $|\mathcal{S}^{k+1}| \leq d + 1$. An algorithm by Dearing and Zeck [39] uses the same overall strategy as the Elzinga–Hearn and Gärtner algorithms, but uses a novel directional search technique to update the support set and its MEB in each iteration.

The algorithm by Fischer et al. [50], on the other hand, can be viewed as a generalization of the primal Peirce–Chrystal algorithm to d dimensions. In their algorithm, each trial ball $\mathcal{B}^k$ has the current candidate support set $\mathcal{S}^k$ on its boundary while at the same time satisfying $\mathcal{B}^k \supset \mathcal{P}$. The center c^k and radius r^k of each $\mathcal{B}^k$ therefore satisfy the earlier system of equations (2.3)–(2.5) from Section 2.1, with $s_1, \dots, s_m$ denoting the points of $\mathcal{S}^k$. The algorithm terminates when $v_i^k \geq 0$ holds for $i = 1, \dots, m$, i.e., when c^k falls in the convex hull of $\mathcal{S}^k$. In each iteration until this is fulfilled, one point $p_i \in \mathcal{S}^k$ with $v_i < 0$ is dropped from $\mathcal{S}^k$, which allows the current ball to be moved and shrunk until another point enters its boundary. Then this point is added to $\mathcal{S}^k$ to generate $\mathcal{S}^{k+1}$. While the dual algorithms above appear to be practically usable only in dimensions up to, say, $d = 50$, experimental results provided by Fischer et al. indicate that this algorithm remains practical for thousands of dimensions. Their algorithm builds on an earlier idea by Hopp and Reeve [64]. A related algorithm is also given by Botkin and Turova-Botkina [19].

2.3.2 Approximation Algorithms

Although some of the exact methods above have proven to be efficient in practice on fairly large problem instances, approximation methods still provide an attractive alternative and have received much consideration in the literature. Such algorithms can be considerably faster, and even exhibit real-time performance. In addition, many algorithms offer the possibility to trade approximation quality for speed across a continuous spectrum. Another advantage of these methods is that they are often fairly straightforward to adapt to other types of geometric primitives given as input, such as balls or ellipsoids.

Coarse approximations can be computed in $O(dn)$ time. For example, a 3/2-approximation of the MEB—i.e., an enclosing ball with radius $r \leq 3/2r^*$—can be computed in the streaming model, where only a single pass over the input is allowed [146, 29]. A simple two-pass algorithm is given by Ritter [102]. Tian [125] gives a three-pass algorithm based on a conservative ball enlargement operation.

Algorithms that compute $(1+\epsilon)$-approximations provide fine control over the approximation quality through the parameter $\epsilon > 0$. Most algorithms in this category are iterative methods that compute (explicitly or implicitly) a sequence of dual feasible solutions u^k, $k = 1, 2, \ldots$, converging to u^* in the limit. In each iteration, a center c^k and radius r^k can be defined using (2.23) and (2.24) with $u = u^k$, and a primal feasible radius is given by $R^k := \max_{i=1}^n \|p_i - c^k\|$. It follows from weak duality that $r^k \leq r^* \leq R^k$; thus, if $(R^k - r^k)/r^k \leq \epsilon$, then the ball $\mathcal{B}_{c^k,R^k}$, which must enclose all of $\mathcal{P}$, has a relative error no larger than $(1+\epsilon)$ in the radius. An early such method given by Lawson [83] is based on the simple recursion

$$u_i^{k+1} = \left(\frac{\|p_i - c^k\|}{\sum_{j=1}^n \|p_j - c^k\| u_j^k} \right) u_i^k, \quad i = 1, \ldots, n. \tag{2.25}$$

As a byproduct, several of these algorithms also construct a subset of $\mathcal{P}$ called a *core-set* [11]. Core-sets are a general tool of computational geometry for efficiently approximating various extent measures of a point set, such as its diameter, width, or the volume of its minimum enclosing box [1]. The defining property of a core-set is that the extent measure in question can be approximated to within a guaranteed precision by being computed exactly only for the core-set, which is significantly faster than computing the exact extent measure for the whole point set. In the context of the MEB problem, a subset $\mathcal{K} \subseteq \mathcal{P}$ is called an *ϵ-core-set* if MEB($\mathcal{K}$), when enlarged by a factor of at most $(1+\epsilon)$, encloses $\mathcal{P}$. Clearly, the enlarged ball must be a $(1+\epsilon)$-approximation of MEB($\mathcal{P}$). Surprisingly, it has been shown [10, 77] that there always exists an ϵ-core-set of size $O(1/\epsilon)$, where the hidden constant is independent of both n and d. In the dual algorithms, the returned core-set typically satisfies $\mathcal{K} = \{p_i \in \mathcal{P} : \tilde{u}_i > 0\}$, where $\tilde{u}$ is the final iterate.

A common strategy used to build the core-set is to begin with an empty set, and then add the farthest point from the current trial ball in each iteration. After each such update, the candidate core-set is used in turn to improve on the solution. A sketch of such a procedure is given in Algorithm 2.1. The strategy of always adding the farthest point p_j to the core-set can be motivated in several ways. Going back to the primal problem formulation ($\mathbf{P}_\mathrm{B}$), the j-th constraint is the only constraint limiting how small the radius can be made for a given c (unless, of course, other points in $\mathcal{P}$ happen to be at the exact same distance from c as p_j, which is rare when $c \neq c^*$). Thus, adding p_j to the core-set and adjusting the ball in the direction of p_j is clearly beneficial.

Considering instead the dual problem ($\mathbf{D}_\mathrm{B}$), the gradient of its objective function has n components, with each ℓ-th component given by

$$\frac{\partial}{\partial u_\ell}\left(\sum_{i=1}^{n} u_i p_i^\mathrm{T} p_i - \Big(\sum_{i=1}^{n} u_i p_i\Big)^\mathrm{T}\Big(\sum_{i=1}^{n} u_i p_i\Big)\right) = p_\ell^\mathrm{T} p_\ell - 2\sum_{i=1}^{n} u_i p_i^\mathrm{T} p_\ell.$$

Using (2.23) to rewrite the right-hand side as $\|p_\ell - c\|^2 - c^\mathrm{T} c$ makes it clear that the j-th component is the largest. This is significant for algorithms that are based on solving linearizations of the dual problem. For example, in the Frank–Wolfe algorithm [53], when applied to problem ($\mathbf{D}_\mathrm{B}$), a linearization of the problem is formed at the current solution u in each iteration. The next solution is then computed by shifting u in the direction of the maximum point of this subproblem using a line search. Because the feasible region of the dual problem is the unit $(n-1)$-simplex, this maximum point is always a vertex of the simplex. Specifically, it is the vertex given by e_j, which we define as the vector with its j-th component equal to 1 and all other components equal to 0.

The asymptotic time complexity of Algorithm 2.1 depends on how the initial solution is computed on Line 1, and how it is updated in each iteration on Line 9. For example, simply setting the initial solution to $c = q$, $r = 0$, and $\mathcal{K} = \{q\}$, where q is an arbitrary point in $\mathcal{P}$,

Algorithm 2.1. Generic $(1+\epsilon)$ MEB algorithm.

```
Input:  $\mathcal{P} := \{p_1, \ldots, p_n\} \subset \mathbb{R}^d, \epsilon > 0$
Output: $\mathcal{B}_{c,r}$ such that $\mathcal{B}_{c,r} \supset \mathcal{P}, r \leq (1+\epsilon)r^*$
 1: Initialize $c$, $r$, and $\mathcal{K}$
 2: loop
 3:     $j \leftarrow \arg\max_{i=1}^{n} \|p_i - c\|$
 4:     $R \leftarrow \|p_j - c\|$
 5:     if $R \leq (1+\epsilon)r$ then
 6:         return $\mathcal{B}_{c,R}$
 7:     end if
 8:     $\mathcal{K} \leftarrow \mathcal{K} \cup \{p_j\}$
 9:     Update $c$, $r$ using $\mathcal{K}$
10: end loop
```

and then invoking an exact MEB algorithm on $\mathcal{K}$ in each step, yields a $(1+\epsilon)$-approximation in $O(1/\epsilon)$ iterations [10]. Because each farthest-point query on Line 3 takes $\Theta(dn)$ time, the total time complexity of such an approach is $O(dn/\epsilon)$. (This strategy is similar to the exact dual algorithms above, except that the core-set is not filtered to contain only the support points).

Another very simple algorithm by Bădoiu and Clarkson [10] proceeds in a gradient descent-type manner, where the center is moved in a straight line toward the farthest point in each iteration, by an amount inversely proportional to the current iteration count. In their original formulation, only a center point is maintained. Thus, a predetermined number of iterations of $1/\epsilon^2$ is used in place of the termination criterion in Algorithm 2.1. The resulting time complexity of this algorithm is $\Theta(dn/\epsilon^2)$. A variant of their algorithm is described by Clarkson [35].

Kumar et al. [77] formulate the MEB problem for the case where the input is a set of balls as a second-order cone program (SOCP), and use a SOCP solver to approximate MEB($\mathcal{K}$) in each iteration. The result is a $(1+\epsilon)$ algorithm with time complexity $O(dn/\epsilon + 1/\epsilon^{4.5}\log(1/\epsilon))$.

The $O(dn/\epsilon)$ algorithm by Panigrahy [96] makes repeated invocations of a procedure similar to Algorithm 2.1. In each invocation, the radius r is kept constant throughout the iterations, and only the center c is moved in each iteration so that the ball $\mathcal{B}_{c,r}$ touches the point p_j. Successively improved approximations of the optimal radius are used for the fixed radius from each invocation to the next.

Yıldırım [144] proposes two algorithms that also fit the outline given in Algorithm 2.1. Both of them are adaptations of Frank and Wolfe's algorithm [53] to the dual problem ($\mathbf{D}_{\mathrm{B}}$). In the first algorithm, the solution u is always moved in the direction of the corner e_j of the unit simplex by an amount that maximizes the growth of the objective function. Such an update consists of first increasing u_j and then rescaling the remaining components of u to maintain primal feasibility. The second algorithm also incorporates Wolfe's away steps [141]. This means that in each iteration, there is also the choice of moving u away from the corner e_{j_-}, where $i = j_-$ minimizes $\|p_i - c\|$ subject to $u_i > 0$. Such an update consists instead of decreasing u_{j_-}, also by an optimal amount but without letting it be reduced below zero, before rescaling the rest of u. The choice between these two types of updates is made dynamically based on which of them will improve the working solution the most. Since away steps may sometimes reduce components of u to zero, the second algorithm has the potential to return smaller core-sets. Moreover, although both algorithms have asymptotic running time $O(dn/\epsilon)$, the second algorithm typically exhibits faster convergence in practice.

In Nielsen and Nock's algorithm [94], the update step amounts to approximating MEB($\mathcal{K}$) using Bădoiu and Clarkson's simple $\Theta(dn/\epsilon^2)$ algorithm as a subroutine. This gives a total time complexity of $O(dn/\epsilon + d/\epsilon^4)$.

2.3.3 Acceleration Techniques

When n is large, the most time-consuming part of the strategy outlined in Algorithm 2.1 tends to be the repeated searches for the farthest point on Line 3. When implemented as a sequential loop over all the points in $\mathcal{P}$, each search takes $\Theta(dn)$ time. For this reason, a number of acceleration techniques targeting this portion of the computations have been proposed.

In a paper accompanying [144], Ahipaşaoğlu and Yıldırım [4] give an acceleration heuristic that is based on eliminating any points from $\mathcal{P}$ that cannot lie on the boundary of MEB($\mathcal{P}$). Since only points on the boundary are needed to define the MEB, removing interior points between iterations has no effect on the final result but can speed up subsequent searches for the farthest point considerably. For a ball $\mathcal{B}_{c,r}$ satisfying $r \leq r^*$ and $\mathcal{B}_{c,(1+\epsilon)r} \supset \mathcal{P}$, it is shown in [4] that if a point p_i

satisfies

$$\|p_i - c\| < (1 - \sqrt{2\epsilon + \epsilon^2})r, \tag{2.26}$$

then it must be interior to MEB($\mathcal{P}$). Thus, in each iteration, any point satisfying this condition with the current value of $\max_{i=1}^{n} \|p_i - c\|/r - 1$ substituted for ϵ can be removed, or "pruned". Condition (2.26) is an example of a *screening rule*, a concept which has also been applied for several other optimization problems [100].

Pronzato [98] derives a bound that is tighter than the bound (2.26), and proves that it cannot be improved further. His bound coincides with an earlier bound given in Paper B of this thesis; see Section 4.

Nielsen and Nock [94] use a simple filtering heuristic to bypass many of the point-point distance computations during each query, but without removing any points from the input. Using the Cauchy–Schwarz inequality, they derive the following test to skip over points when searching for the farthest point:

$$\sqrt{\|p_i\|^2 + \|c\|^2 + 2\|p_i\|\|c\|} < R_i, \tag{2.27}$$

where R_i is the largest distance $\|p_\ell - c\|$ for $\ell < i$. Any point p_i satisfying (2.27) cannot be the farthest point in the current farthest-point query, thus the exact distance $\|p_i - c\|$ does not need to be computed. By precomputing $\|p_i\|$ for $i = 1, \ldots, n$ only once, and computing $\|c\|$ before each query, this test can be performed in constant time per point, in contrast to the $\Theta(d)$ cost of computing the exact distance.

It should be noted that the above techniques clearly are applicable to any algorithm based on the overall strategy of Algorithm 2.1. We further remark that while general acceleration techniques is a broad topic, the term is restricted here to techniques that are "transparent", meaning that they bring performance improvements while at the same time having no effect on the final result or the intermediate solutions generated by the algorithm.

2.4 Applications

Since the computation of the MEB is such a fundamental optimization problem, it is no surprise that it finds numerous applications across multiple domains, as was alluded to in Chapter 1. In this section, we give a slightly more detailed survey of a few interesting applications involving both low-dimensional and high-dimensional data.

2.4.1 Example: Collision Detection

As discussed briefly in Chapter 1, efficient collision detection is a critical task in many applications involving simulation of the physical world. For example, in a racing video game the vehicles should not be allowed to pass straight through each other or drive off the track. Thus, their intersection status with each other as well the environment must be detected and handled in each frame of the game loop. This is a challenge when the game objects are represented using complex geometry, such as highly detailed triangle meshes or implicit surface descriptions, since it becomes prohibitively expensive to compute and evaluate all possible contact points.

In the context of these applications, the enclosing shapes are usually called *bounding volumes*. Apart from bounding spheres [99, 65, 104, 67, 92] and ellipsoids [101, 84, 80, 25], other well-known choices of bounding volume shapes include axis-aligned [107, 23] and oriented [58] bounding boxes, and discrete oriented polytopes [74]. As mentioned before, by having each model in the virtual world enclosed by a tight-fitting but simple such bounding volume, quick intersection tests between them can serve as initial rejection tests.

A refinement of this idea involves arranging the bounding volumes in a tree structure called a *bounding volume hierarchy* [47]. The leaves of this tree store the individual geometric primitives that make up the model, and each internal node stores a bounding volume that encloses all the primitives of its subtree. With this data structure, a collision query between two objects can be performed at successively increasing levels of detail. If the initial test between the two top-level bounding volumes enclosing the entire models returns an intersection, then the next level of the hierarchies, consisting of several smaller bounding volumes, are examined, and so on. This way, a collision query between two objects is realized as a parallel traversal of their respective trees, where any pair of subtrees whose bounding volumes do not intersect can be excluded from further processing.

Bounding volumes can also be used to accelerate various rendering techniques used in computer-generated imagery. For example, in ray tracing, where realistic images are produced by simulating rays of light traveling through a virtual scene, bounding volume hierarchies can be used to perform efficient intersection detection between the rays and the objects in the scene [137].

The minimum bounding sphere has a number of advantages over other types of bounding volume shapes. Firstly, since the center point and radius can be represented using only four floating-point values, it has very low memory requirements, which is important in large simulations involving many objects. Secondly, its simple shape enables very cheap intersection tests. For example, testing whether two spheres $\mathcal{B}_{c_1,r_1}$ and $\mathcal{B}_{c_2,r_2}$ intersect amounts to evaluating the condition

$$\|c_1 - c_2\|^2 \leq (r_1 + r_2)^2.$$

This test, where the squared distance is used to avoid taking a square root, takes only 10 arithmetic operations and one comparison. Finally, in animated environments where the models undergo rigid-body transformations, i.e., translations and/or rotations, their bounding spheres can be updated cheaply by simply applying the same transformations to their center points. In particular, the sphere is invariant under rotation about its center.

The effectiveness of a bounding volume, however, is also closely tied to how accurately it approximates the underlying shape, since loosely fitting bounding volumes can incur excessive false positives in the collision tests. Unfortunately, a spherical shape does not in general provide a good approximation of geometric models encountered in practice. For this reason, the minimum bounding sphere might not be an appropriate choice in some applications. Some authors propose combining the minimum bounding sphere with some other more tight-fitting bounding volume [81, 30, 70].

2.4.2 Example: Machine Learning

Classification and anomaly detection are important problems in machine learning, with numerous applications such as e-mail spam filtering [7], medical image analysis [91], and condition monitoring of industrial machines [121]. The task of classification is to categorize previously unseen observations (or “objects” or “examples”, e.g., an e-mail or an fMRI image) as belonging to a set of possible classes (e.g., spam/not spam or benign/malignant). Novelty detection, also called outlier detection, is concerned with determining whether a new observation in some way deviates from a known pattern of “normal” behavior. For example, if a vibration sensor measures abnormal vibrations in a machine, this might indicate a failing component.

Support Vector Data Descriptions

The *support vector data description* (SVDD), introduced by Tax and Duin [122], is an approach to anomaly detection based on comparing incoming observations against a set of known observations representing normal conditions. Initially, each observation in the "normal" set is encoded as a numerical vector called a *feature vector*. Then the SVDD undergoes a "training" step, in which the MEB of these vectors is computed. This ball is then used as a description of the training set, so that any incoming observations whose feature vectors fall outside of it are deemed to be anomalies.

A natural performance measure of the SVDD is the number of anomalies it detects/fails to detect, as well as the amount of false alarms it generates. These parameters are directly related to how well the SVDD describes the training examples. Unfortunately, a spherical shape does not in general provide an accurate representation of the training set, as it tends to contain too much empty space. Thus, any anomalies that happen to fall in that empty space will go undetected. A standard method used in several machine learning methods to improve their accuracy is to let the feature vectors be transformed to a higher-dimensional space using a nonlinear map ϕ. In the case of the SVDD, this allows the MEB to be computed in a space in which it does provide a tight fit around the training examples. Any new observations are also transformed and tested against the MEB in the same space.

A technique known as the *kernel trick* makes it possible to transform the vectors into infinite-dimensional feature spaces. The idea is to make the transformations by the map ϕ implicit, by replacing all inner products involved in fitting the MEB and testing observations against it with a *kernel function* $k(x, y) := \phi(x)^{\mathrm{T}}\phi(y)$. The dual objective (2.15), when modified for the problem of computing the MEB of the transformed points $\phi(p_1), \dots, \phi(p_n)$, can then be written as

$$\begin{aligned}\sum_{i=1}^{n} u_i \phi(p_i)^{\mathrm{T}}\phi(p_i) &- \sum_{i=1}^{n}\sum_{j=1}^{n} u_i u_j \phi(p_i)^{\mathrm{T}}\phi(p_j) \\ &= \sum_{i=1}^{n} u_i k(p_i, p_i) - \sum_{i=1}^{n}\sum_{j=1}^{n} u_i u_j k(p_i, p_j).\end{aligned}$$

With c^* and r^* defined from u^* using (2.23) and (2.24), the test for whether the image of an observation x under ϕ is contained in the MEB

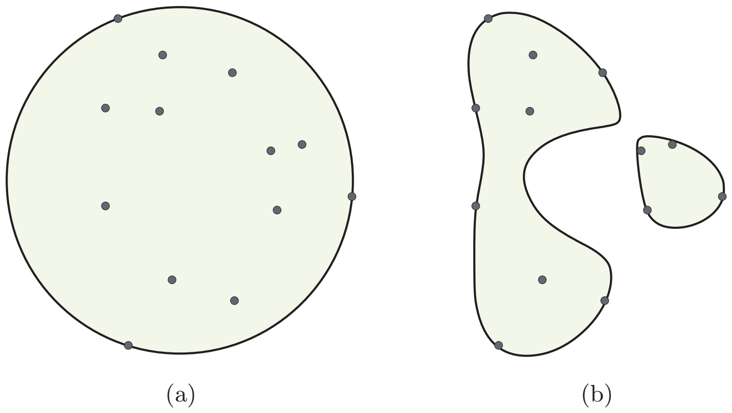


Figure 2.7. Support vector data descriptions, rendered in the original feature space. (a) The regular inner product kernel $k(x,y) = x^{\mathrm{T}}y$ has been used to fit the SVDD. (b) The Gaussian kernel $k(x,y) = -\exp(\|x - y\|^2/s^2)$ has been used with $s = 100$.

can be written as

$$\begin{aligned}(r^*)^2 &\geq \|\phi(x) - c^*\|^2\\ &= \phi(x)^{\mathrm{T}}\phi(x) - 2\phi(x)^{\mathrm{T}}c^* + (c^*)^{\mathrm{T}}c^*\\ &= \phi(x)^{\mathrm{T}}\phi(x) - 2\sum_{i=1}^{n} u_i^*\phi(x)^{\mathrm{T}}\phi(p_i) + \sum_{i=1}^{n}\sum_{j=1}^{n} u_i^*u_j^*\phi(p_i)^{\mathrm{T}}\phi(p_j)\\ &= k(x,x) - 2\sum_{i=1}^{n} u_i^*k(x,p_i) + \sum_{i=1}^{n}\sum_{j=1}^{n} u_i^*u_j^*k(p_i,p_j),\end{aligned}$$

where (2.23) is used in the second equality.

Popular choices of kernel include the Gaussian (or radial basis function, RBF) kernel: $k(x,y) = -\exp(\|x-y\|^2/s^2)$, where $s > 0$ is a tunable parameter, and the polynomial kernel: $k(x,y) = (x^{\mathrm{T}}y)^D$, where D determines the degree of the polynomial. Figure 2.7 shows an SVDD trained in the original feature space, as well as one trained in the feature space induced by a Gaussian kernel.

The above variant of the SVDD can be called the *hard-margin* SVDD since it fully encloses all of the training examples. In contrast, a *soft-margin* SVDD is trained with slack variables added to each constraint in the primal MEB formulation. This makes the method more robust when new training examples are added or if the training set contains outliers, as these can be left outside of the MEB. An additional penalty parameter controls the trade-off between the volume of the data description and the errors. Another variation supports *negative examples* in the training set, which should be left out of the data description.

Support Vector Machines

The *support vector machine* (SVM) [37] is among the most widely used methods for classification. In the case of the standard SVM, each training example is labeled as belonging to one of two classes, and the training step amounts to finding a hyperplane that separates the feature vectors of one class from those of the other class with maximum margin. Any new observations are then assigned a class label based on which side of this hyperplane they fall.

As with the SVDD, a soft-margin version of the SVM, which tolerates training vectors ending up on the wrong side of the hyperplane, can be constructed by adding slack variables and a penalty parameter to the hyperplane fitting problem. This makes the SVM more robust to the training set being updated with new examples. It also enables training the SVM on sets that are not linearly separable, i.e., for which no separating hyperplane exists. In applications where this is not sufficient, the kernel trick can be employed to implicitly transform the training examples to a higher-dimensional feature space in which a separating hyperplane can be found.

It was shown by Tsang et al. [132] that for certain choices of kernel, the optimization problem of finding the separating hyperplane is equivalent to the MEB problem in the kernel-induced feature space. Specifically, this holds for any kernel function with the property that $k(x, x)$ is a constant for all x. This includes, e.g., the Gaussian kernel mentioned above. It had already been shown by Schölkopf et al. [106] that with such kernels, the SVDD fitting problem above is equivalent to that of finding a hyperplane that separates the training examples from the origin with maximum margin. A model based on such a hyperplane is commonly referred to as a one-class SVM. Tsang et al. exploit the insight that also the two-class SVM training problem can be expressed as a MEB problem, and obtain improved training times by using the $(1 + \epsilon)$-approximation MEB algorithm by Bădoiu and Clarkson [11] in place of standard training methods. The approach has also been further generalized to enable the use of other popular kernels that do not satisfy the property above [133].

2.5 Related Problems

There are a number of ways in which generalized variants of the MEB problem can be obtained. Although these problems are not addressed directly in the papers in this thesis, we survey some of them here for additional context.

In the *weighted* MEB problem, introduced by Francis [52], each of the n points p_i has a weight $w_i > 0$ attached to it, and the center c is sought that minimizes the maximum weighted distance $w_i\|p_i - c\|$, $i = 1, \ldots, n$. In facility location planning, such weights can be used to model that certain sites are more important, or more difficult to reach, than others. Hearn and Vijay [62] give several algorithms for the weighted problem in the plane, including generalizations of the Peirce–Chrystal and Elzinga–Hearn procedures discussed in Section 2.3. Linear-time algorithms (in fixed dimension) based on the multidimensional search technique by Megiddo are given by Dyer [44] and Megiddo [90]. Recent approaches can be found in, e.g., [78, 41, 40].

Another generalization is to use other distance measures than the Euclidean distance between the center and the points. For example, the metric induced by the 1-norm $\|x\|_1 := \sum_{i=1}^{d} |x_i|$, called the rectilinear or Manhattan distance, is more appropriate in facility location scenarios where the sites can only be reached via roads arranged in a grid [45]. Intuitively, a circle defined using this metric has the shape of a "diamond", a square rotated by 45 degrees. Nielsen and Nock [95] discuss a generalization of the MEB problem that uses a class of distance measures known as Bregman divergences, and adapt the simple algorithm by Bădoiu and Clarkson [10] to find a $(1 + \epsilon)$-approximation.

A fair amount of attention has also been devoted in the literature to the problem of finding the MEB of a finite set of balls, as opposed to points. Given a set of balls with centers c_i and radii $r_i \geq 0$, $i = 1, \ldots, n$, the task is thus to minimize the maximum of $\|c_i - c\| + r_i$. When interpreted as a facility location problem, each radius r_i represents as an additional, constant distance incurred at each site [45]. Another application is the construction of sphere hierarchies, where the sphere stored in each internal node is to be fitted to the spheres of its descendant nodes [99, 65, 104, 67, 47]. It was shown by Megiddo [90] that the $O(n)$ bound holds also for this variant of the problem. Analogously to the point case, the MEB of a given set of balls is defined by at most $d+1$ of the balls that are internally tangent to the boundary of the MEB. Despite this, however, the combinatorial methods discussed in Section 2.3.1, which compute the exact MEB by finding the support points, cannot be adapted in a trivial way to this problem [49]. The $(1+\epsilon)$-approximation algorithms of Section 2.3.2 that are based on Algorithm 2.1, on the other hand, are easily modified for the case of ball sets if each ball is interpreted as an infinite set of points: Simply altering Line 3 of Algorithm 2.1 to instead compute $j \leftarrow \arg\max_{i=1}^{n} \|c_i - c\| + r_i$, and substituting the point $c_j + \frac{r_j}{\|c_j - c\|}(c_j - c)$ for p_j, yields a $(1+\epsilon)$-approximation algorithm with unchanged asymptotic time complexity. Note that the core-set constructed by such a modified algorithm will still contain only points, and will have the same bound on its cardinality as before.

In another example of an effort toward computing the MEB for more general input primitives, Muthuganapathy et al. [92] adapt the randomized algorithm by Welzl [139] to compute the MEB of a set of free-form hypersurfaces in expected linear time.

Chapter 3

Minimum-Volume Enclosing Ellipsoids

3.1 Introduction

An ellipse can be specified in a number of ways. For example, when defined using two focal points, it is the curve of points at which the sum of the distances to the focal points equals a specified constant. This is depicted in Figure 3.1a. The sum of the distances to the focal points is then smaller for points strictly inside of the ellipse, and larger for points outside of it. Another way of specifying an ellipse, which also generalizes beyond the planar case in a straightforward manner, is as a circle that has been stretched and/or squeezed along two mutually perpendicular directions. This is illustrated in Figure 3.1b.

Analogously, an ellipsoid in d dimensions can be defined as a ball that has been scaled along d mutually perpendicular directions. (In other words, an ellipsoid is viewed in this thesis as being the whole subset of $\mathbb{R}^d$ on and inside of its boundary; hence, the term ball is used here instead of the term (hyper-)sphere. See remarks in Section 2.1.) If the ball is also allowed to be moved, then the choice of ball can, without loss of generality, be restricted to the unit ball centered at the origin. More formally, an ellipsoid is then given by the affine transformation $Ax + c$ being applied to the points $x \in \mathbb{R}^d$ such that $\|x\| \leq 1$, where $c \in \mathbb{R}^d$ gives the center of the ellipsoid and the columns of $A \in \mathbb{R}^{d \times d}$

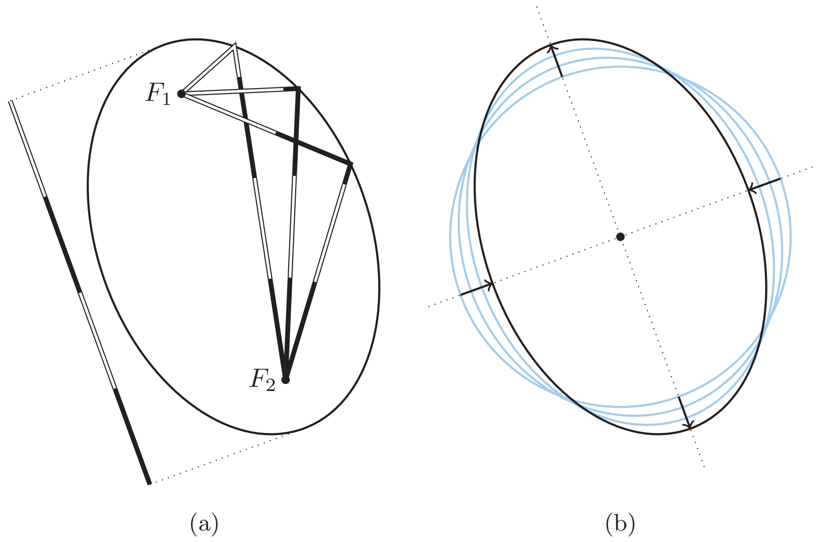


Figure 3.1. Two ways to specify an ellipse. (a) The ellipse is defined using two focal points F_1 and F_2 as the curve where at each point, the sum of the distances to F_1 and F_2 equals a constant. This property is illustrated for three points on the ellipse. Also illustrated is that the same constant gives the longest diameter of the ellipse, called the major axis. (b) The ellipse is defined as a circle that has been scaled along two perpendicular axes.

are orthogonal vectors determining its orientation and extents. This is illustrated in Figure 3.2. Our focus in this thesis is on full-dimensional ellipsoids, therefore it will be assumed henceforth that A has full rank, i.e., that its columns form an orthogonal basis for $\mathbb{R}^d$. The line segments $\{(1-t)c + ta_j : 0 \leq t \leq 1\}$, $j = 1, \ldots, d$, where a_j denotes the j-th column vector of A, are called *semi-axes* of the ellipsoid.

The same ellipsoid can be described as

$$\{x \in \mathbb{R}^d : \|A^{-1}(x-c)\| \leq 1\},$$

i.e., as the set of points that, subject to the inverse affine transformation above, end up inside the unit ball centered at the origin. By squaring

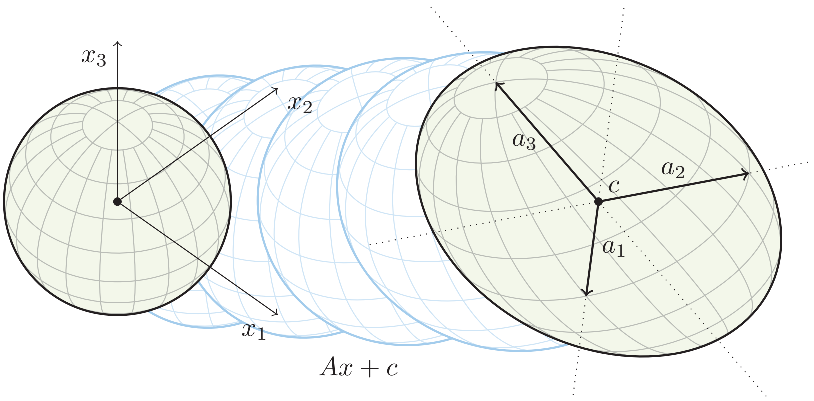


Figure 3.2. Any ellipsoid can be described as the unit ball subject to an affine transformation. The coordinate axes are labeled x_1 through x_3. The vectors a_1 through a_3 are the columns of A.

both sides of the inequality, it can be written as

$$\begin{aligned} 1 &\geq \|A^{-1}(x-c)\|^2 \\ &= (A^{-1}(x-c))^{\mathrm{T}}(A^{-1}(x-c)) \\ &= (x-c)^{\mathrm{T}}(A^{-1})^{\mathrm{T}}A^{-1}(x-c). \end{aligned}$$

Thus, the shape of the ellipsoid can be equivalently specified using the symmetric matrix $Q := (A^{-1})^{\mathrm{T}}A^{-1}$, which must be positive-definite since A has full rank. Conversely, any symmetric positive-definite matrix Q' permits an eigendecomposition $Q' = VDV^{\mathrm{T}}$, where V is an orthogonal matrix whose columns are eigenvectors of Q', and D is a diagonal matrix whose entries are the corresponding eigenvalues, which must all be positive [134]. Thus, any such matrix describes the shape of an ellipsoid with $A = VD^{-1/2}$, where V gives the rotation of the ellipsoid and the j-th diagonal element of $D^{-1/2}$ gives the length of the j-th semi-axis. Interestingly, this further implies that the earlier requirement that A has orthogonal columns is not necessary in principle, since a matrix satisfying this requirement can always be obtained via the corresponding matrix Q. Moreover, applying any affine transformation to a full-dimensional ellipsoid always gives another ellipsoid: Say that such an ellipsoid is defined as $A_1x + c_1$ for $\|x\| \leq 1$; the transformation $A_2(A_1x + c_1) + c_2$

then gives an ellipsoid with matrix $A = A_2 A_1$ and center $c = A_2 c_1 + c_2$.

In what follows, we will denote an ellipsoid with center $c \in \mathbb{R}^d$ and symmetric positive-definite shape matrix $Q \in \mathbb{R}^{d \times d}$ by

$$\mathcal{E}_{Q,c} \coloneqq \{x \in \mathbb{R}^d : (x - c)^{\mathrm{T}} Q (x - c) \leq 1\}. \tag{3.1}$$

The matrix Q can be used to define the *ellipsoidal norm*

$$\|x\|_Q \coloneqq \sqrt{x^{\mathrm{T}} Q x}. \tag{3.2}$$

Thus, the ellipsoid defined in (3.1) can be interpreted as the set of points within distance 1, measured in the ellipsoidal norm, from its center, with the points at distance exactly 1 defining its boundary. As before, we denote the given points by $\mathcal{P} \coloneqq \{p_1, \ldots, p_n\} \subset \mathbb{R}^d$. The ellipsoid $\mathcal{E}_{Q,c}$ is an enclosing ellipsoid of $\mathcal{P}$ if $\mathcal{E}_{Q,c} \supset \mathcal{P}$, which is equivalent to $(p_i - c)^{\mathrm{T}} Q (p_i - c) \leq 1$ being satisfied for $i = 1, \ldots, n$.

The volume of $\mathcal{E}_{Q,c}$ is given by

$$\operatorname{vol} \mathcal{E}_{Q,c} = \zeta \det Q^{-1/2}, \tag{3.3}$$

where ζ is the volume of a d-dimensional unit ball[1]. The MVEE of $\mathcal{P}$, from now on denoted by MVEE($\mathcal{P}$), is thus the enclosing ellipsoid $\mathcal{E}_{Q,c}$ of $\mathcal{P}$ that minimizes (3.3). For the MVEE to be full-dimensional, the points $\mathcal{P}$ must have affine hull $\mathbb{R}^d$; otherwise, they inhabit a lower-dimensional hyperplane, which means the MVEE can be made infinitely thin. As is the case with the MEB, the MVEE is unique and determined by the points that end up on its boundary [69]. Due to the additional degrees of freedom in describing an ellipsoid, however, the support set contains *at least* $d + 1$ and at most $d(d + 3)/2$ such points.

John [69] further showed that

$$d^{-1} \operatorname{MVEE}(\mathcal{P}) \subset \operatorname{conv}(\mathcal{P}) \subset \operatorname{MVEE}(\mathcal{P}), \tag{3.4}$$

where conv($\mathcal{P}$) denotes the convex hull of $\mathcal{P}$ and d^{-1} MVEE($\mathcal{P}$) denotes MVEE($\mathcal{P}$) scaled about its center by d^{-1}. The MVEE is thus said to provide a *d-rounding* of conv($\mathcal{P}$). If $\mathcal{P}$ is symmetric about the origin, i.e., if $\mathcal{P} = -\mathcal{P}$, then the scaling factor can be improved to $d^{-1/2}$.

[1]Note that $\det(Q^{-1/2}) = (\det Q)^{-1/2}$, where $Q^{-1/2}$ denotes a matrix such that $Q^{-1/2} Q^{-1/2} = Q^{-1}$. Thus, parentheses can be left out without ambiguity.

3.2 Problem Formulations

Dividing out the constant factor ζ from the volume formula (3.3) and squaring the result allows the problem of computing MVEE($\mathcal{P}$) to be formulated as

$$
\begin{aligned}
(\mathbf{P}'_{\mathrm{E}}) \quad & \min_{Q,c} \quad && \det Q^{-1} && (3.5)\\
& \text{s.t.} && (p_i - c)^{\mathrm{T}} Q (p_i - c) \le 1, \quad i = 1, \dots, n, && (3.6)\\
& && Q \text{ is positive-definite.} && (3.7)
\end{aligned}
$$

If $\mathcal{P}$ is symmetric about the origin, then MVEE($\mathcal{P}$) must have its center at the origin. Substituting $c = 0$ in problem ($\mathbf{P}'_{\mathrm{E}}$) simplifies it to the problem of computing the minimum-volume *centered* ellipsoid enclosing $\mathcal{P}$. Although the symmetry property does not hold for general $\mathcal{P}$, it can be made to hold by replacing $\mathcal{P}$ with the "lifted" point set

$$\widehat{\mathcal{P}} := \{\pm \widehat{p}_i : i = 1, \dots, n\} \subset \mathbb{R}^{d+1}, \quad \widehat{p}_i := \begin{bmatrix} p_i \\ 1 \end{bmatrix}. \tag{3.8}$$

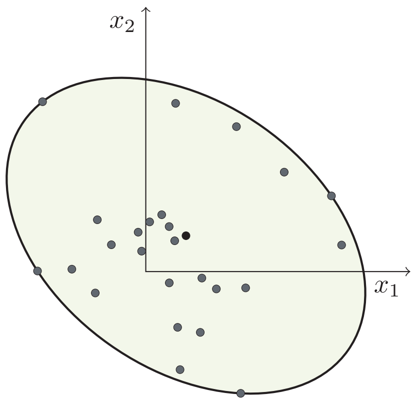
Given the centered solution MVEE($\widehat{\mathcal{P}}$) in $d+1$ dimensions, the non-centered solution MVEE($\mathcal{P}$) in d dimensions is then given by the intersection of MVEE($\widehat{\mathcal{P}}$) and the hyperplane

$$\mathcal{H} := \left\{ \begin{bmatrix} x \\ 1 \end{bmatrix} : x \in \mathbb{R}^d \right\} \tag{3.9}$$

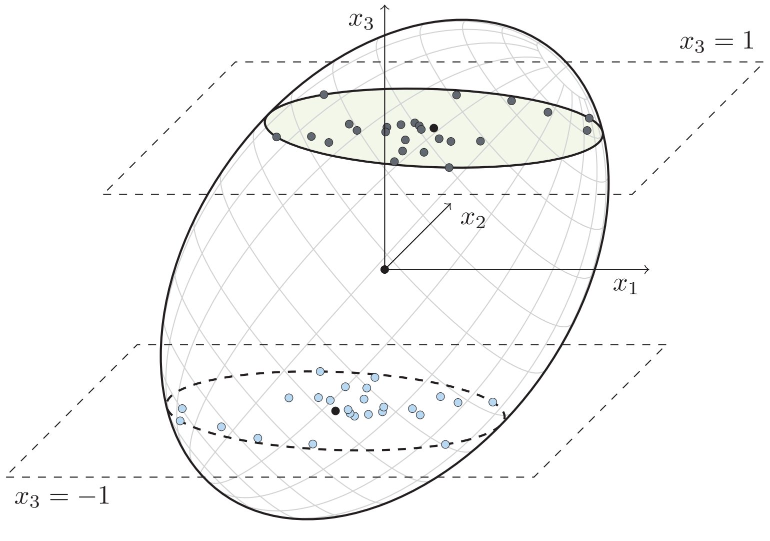
[126, 71]. The idea is shown in Figure 3.3. We will return shortly to the details of computing this intersection, which can be done analytically. Note that although this lifting comes with the cost of one added dimension, the mirror image $-\widehat{p}_i$ of each $\widehat{p}_i$ does not need to be represented explicitly, as it will automatically be covered by any centered ellipsoid covering $\widehat{p}_i$. Thus, there are still n points to consider.

A further modification of problem ($\mathbf{P}'_{\mathrm{E}}$) is to apply the natural logarithm to the determinant, which gives a new objective function that is strictly convex on the cone of positive-definite matrices. In fact, for convenience we follow the example of Todd [129] and define the operator

$$\operatorname{logdet} M := \begin{cases} \log\det M & \text{if } M \text{ is positive-definite,} \\ -\infty & \text{otherwise.} \end{cases} \tag{3.10}$$



(a)



(b)

Figure 3.3. (a) A point set in $\mathbb{R}^2$ together with its minimum enclosing ellipse. (b) The corresponding lifted point set in $\mathbb{R}^3$ together with its centered MVEE. The points shown in a lighter shade do not need to be represented explicitly. The ellipse that is the intersection of the MVEE with the plane $x_3 = 1$ is a lifted version of that in (a).

The resulting problem, over symmetric matrices $M \in \mathbb{R}^{(d+1)\times(d+1)}$, becomes

$$\begin{aligned}
(\mathbf{P}_{\mathrm{E}}) \quad & \min_M && -\operatorname{logdet} M && (3.11)\\
& \text{s.t.} && \widehat{p}_i^{\mathrm{T}} M \widehat{p}_i \le 1, \quad i = 1, \ldots, n. && (3.12)
\end{aligned}$$

Using Lagrange multipliers $u \coloneqq (u_1, \ldots, u_n)^{\mathrm{T}}$, define

$$L(M, u) \coloneqq -\operatorname{logdet} M + \sum_{i=1}^{n} u_i(\widehat{p}_i^{\mathrm{T}} M \widehat{p}_i - 1). \tag{3.13}$$

Clearly, $L(M, u) \le -\operatorname{logdet} M$ is satisfied for nonnegative u and any feasible M. $L(M, u)$ is strictly convex and differentiable everywhere it is finite, and therefore minimized at a stationary point $M = \overline{M}$, i.e.,

$$\nabla_M L(\overline{M}, u) = -\overline{M}^{-1} + \sum_{i=1}^{n} u_i \widehat{p}_i \widehat{p}_i^{\mathrm{T}} = 0,$$

where $\nabla \operatorname{logdet} \overline{M} = (\overline{M}^{-1})^{\mathrm{T}}$ was used. Hence, $\overline{M} = V(u)^{-1}$, where

$$V(u) \coloneqq \sum_{i=1}^{n} u_i \widehat{p}_i \widehat{p}_i^{\mathrm{T}}. \tag{3.14}$$

In summary, for any nonnegative u such that $V(u)$ is positive-definite,

$$\begin{aligned}
-\operatorname{logdet} M^* &\ge \min_M L(M, u) && (3.15)\\
&= \operatorname{logdet} V(u) + \sum_{i=1}^{n} u_i \widehat{p}_i^{\mathrm{T}} V(u)^{-1} \widehat{p}_i - \sum_{i=1}^{n} u_i && (3.16)\\
&= \operatorname{logdet} V(u) + \operatorname{trace}\big(V(u)^{-1} V(u)\big) - \sum_{i=1}^{n} u_i && (3.17)\\
&= \operatorname{logdet} V(u) + d + 1 - \sum_{i=1}^{n} u_i, && (3.18)
\end{aligned}$$

where (3.17) uses (3.14) and basic properties of the trace operator. Strong duality, which must hold since $(\mathbf{P}_{\mathrm{E}})$ satisfies Slater's constraint qualification [21], states that the maximum of (3.18) over u coincides with the minimum of $(\mathbf{P}_\mathrm{E})$. If we denote by u^* the maximizer of (3.18), the complete KKT conditions can thus be stated as

$$\widehat{p}_i^\mathrm{T} M^* \widehat{p}_i \leq 1, \quad i = 1, \ldots, n, \tag{3.19}$$

$$u_i^* \geq 0, \quad i = 1, \ldots, n, \tag{3.20}$$

$$M^* = V(u^*)^{-1}, \tag{3.21}$$

$$u_i^* (\widehat{p}_i^\mathrm{T} M^* \widehat{p}_i - 1) = 0, \quad i = 1, \ldots, n. \tag{3.22}$$

Substituting (3.21) in (3.22) and summing up for $i = 1, \ldots, n$ gives

$$\begin{aligned} \sum_{i=1}^n u_i^* &= \sum_{i=1}^n u_i^* \widehat{p}_i^\mathrm{T} V(u^*)^{-1} \widehat{p}_i \\ &= \operatorname{trace}\left(V(u^*)^{-1} V(u^*)\right) \\ &= d + 1. \end{aligned}$$

Thus, when formulating the Lagrangian dual of $(\mathbf{P}_\mathrm{E})$, we can further constrain u to satisfy $\sum_{i=1}^n u_i = d+1$, which causes the last three terms in (3.18) to cancel out. However, by instead constraining u to sum to 1, we get an equivalent problem known as the *D-optimal design problem* [126]:

$$(\mathbf{D}_\mathrm{E}) \quad \max_u \quad \operatorname{logdet}\left(\sum_{i=1}^n u_i \widehat{p}_i \widehat{p}_i^\mathrm{T}\right) \tag{3.23}$$

$$\text{s.t.} \quad \sum_{i=1}^n u_i = 1, \tag{3.24}$$

$$u_i \geq 0, \quad i = 1, \ldots, n. \tag{3.25}$$

(See Section 3.4.1.) It is clear that if u is a feasible solution of $(\mathbf{D}_\mathrm{E})$, then $(d+1)u$ is a feasible solution of the true dual of $(\mathbf{P}_\mathrm{E})$, with the corresponding objective function value given simply by adding $(d+1)\log(d+1)$ to that of $(\mathbf{D}_\mathrm{E})$.

Thus, a trial ellipsoid $\mathcal{E}_{M,0}$ defined from a feasible solution u of $(\mathbf{D}_\mathrm{E})$ as

$$M = \frac{1}{d+1} V(u)^{-1}, \tag{3.26}$$

coincides with $\mathrm{MVEE}(\widehat{\mathcal{P}})$ for $u = u^*$. (Note that the factor $1/(d+1)$ translates to the added term $(d+1)\log(d+1)$ when substituting (3.26) in (3.11).) For any other feasible u, weak duality gives that $\mathcal{E}_{M,0}$ is undersized, i.e., that it has points of $\widehat{\mathcal{P}}$ outside of it.

3.2.1 Recovering a Non-Centered Solution

We now address in more detail how a trial solution $\mathcal{E}_{M,0}$ defined via (3.26) for $\widehat{\mathcal{P}}$ corresponds to a trial solution $\mathcal{E}_{Q,c}$ for $\mathcal{P}$. Firstly, we note that if $c \in \mathbb{R}^d$ and $Q \in \mathbb{R}^{d \times d}$ satisfy

$$c = \sum_{i=1}^{n} u_i p_i, \quad Q = \frac{1}{d}\Big(\sum_{i=1}^{n} u_i p_i p_i^{\mathrm{T}} - cc^{\mathrm{T}}\Big)^{-1}, \tag{3.27}$$

then it can be shown using (3.8) and (3.14) that $V(u)$ satisfies

$$V(u) = \begin{bmatrix} \sum_{i=1}^{n} u_i p_i p_i^{\mathrm{T}} & c \\ c^{\mathrm{T}} & 1 \end{bmatrix}, \quad V(u)^{-1} = d \begin{bmatrix} Q & -Qc \\ -(Qc)^{\mathrm{T}} & c^{\mathrm{T}} Q c + d^{-1} \end{bmatrix}. \tag{3.28}$$

To verify that c and Q then gives the intersection of $\mathcal{E}_{M,0}$ with the hyperplane $\mathcal{H}$ in (3.9), first note that the intersection should be exactly the points $x \in \mathbb{R}^d$ such that

$$\widehat{x}^{\mathrm{T}} M \widehat{x} \le 1,$$

where $\widehat{x} := (x^{\mathrm{T}}, 1)^{\mathrm{T}}$. Now, by (3.26) this is equivalent to

$$\widehat{x}^{\mathrm{T}} V(u)^{-1} \widehat{x} \le d + 1,$$

which can be written using (3.28) as

$$(x - c)^{\mathrm{T}} Q (x - c) \le 1.$$

It is also clear that $\operatorname{vol} \mathcal{E}_{Q,c}$ can be expressed as a monotone function of $\operatorname{vol} \mathcal{E}_{M,0}$, since

$$\begin{aligned} \det Q^{-1} &= d^d \det \Big(\sum_{i=1}^{n} u_i p_i p_i^{\mathrm{T}} - cc^{\mathrm{T}}\Big) \\ &= d^d \det V(u) \\ &= \frac{d^d}{(d+1)^{d+1}} \det M^{-1}, \end{aligned}$$

where the first and last steps use (3.27) and (3.26), respectively, and the second step applies Schur's determinant identity to $V(u)$ in (3.28). This establishes that $\mathcal{E}_{Q^*,c^*} = \mathrm{MVEE}(\mathcal{P})$, where c^* and Q^* are defined using (3.27) with $u = u^*$: Otherwise, the matrix Q' of $\mathrm{MVEE}(\mathcal{P})$ would satisfy $\det(Q')^{-1} < \det(Q^*)^{-1}$, and an enclosing ellipsoid of $\widehat{\mathcal{P}}$ with a smaller volume than $\mathrm{MVEE}(\widehat{\mathcal{P}})$ could be defined by substituting Q' and the center c' of $\mathrm{MVEE}(\mathcal{P})$ in (3.28) and applying (3.26)—a contradiction.

Finally, we point out the following correspondence between the ellipsoidal distance of a point x from the center of $\mathcal{E}_{Q,c}$, and the distance of the point $\widehat{x}$ from the center of $\mathcal{E}_{M,0}$:

$$(x-c)^{\mathrm{T}} Q (x-c) = \frac{1}{d}\big((d+1)\widehat{x}^{\mathrm{T}} M \widehat{x} - 1\big), \tag{3.29}$$

$$\widehat{x}^{\mathrm{T}} M \widehat{x} = \frac{1}{d+1}\big(d(x-c)^{\mathrm{T}} Q (x-c) + 1\big). \tag{3.30}$$

Notice from this relation, which can verified using (3.28), that $(x - c)^{\mathrm{T}} Q(x-c)$ can be viewed as a monotonic function of $\widehat{x}^{\mathrm{T}} M \widehat{x}$ and vice versa. In other words, the sets $\{p_1, \dots, p_n\}$ and $\{\widehat{p}_1, \dots, \widehat{p}_n\}$ are order-isomorphic w.r.t. the orders induced by the distances to the respective ellipsoid centers.

3.3 Methods of Solution

Several of the earliest algorithms for the MVEE problem were developed in the context of the D-optimal design problem. The equivalence with the centered MVEE problem was shown by Silvey [114] and Sibson [112], and the transformation of the non-centered problem to the centered one using the aforementioned lifting trick is due to Titterington [126].

Like the algorithms for the MEB problem of Section 2.3, many of the algorithms to be discussed here can be categorized as either primal or dual, depending on whether they generate intermediate solutions to problem ($\mathbf{P}_{\mathrm{E}}$) or ($\mathbf{D}_{\mathrm{E}}$), respectively. In fact, several of the methods can be described, at least in broad strokes, as direct adaptations of those previously discussed methods for the MEB problem.

3.3.1 Exact Algorithms

In the research literature on optimal experimental design, an early algorithm that computes an exact result is found in [115]. Here Silvey and Titterington sketch an algorithm that uses a similar strategy as the dual algorithm for the MEB problem by Gärtner [56] discussed in Section 2.3.1: Select an initial candidate support set $\mathcal{S}^0 \subseteq \widehat{\mathcal{P}}$ containing at least $d+1$ affinely independent points and compute its MVEE (D-optimal design) $\mathcal{E}^0$. If $\mathcal{E}^0 \supset \widehat{\mathcal{P}}$, then $\mathcal{E}^0 = \text{MVEE}(\widehat{\mathcal{P}})$, so exit. Otherwise select the point $\widehat{p}_i = \widehat{p}_j$ with the largest ellipsoidal distance $\widehat{p}_i^{\mathrm{T}} M^0 \widehat{p}_i$, where M^0 is the shape matrix of $\mathcal{E}^0$. Then compute the MVEE $\mathcal{E}^1$ of $\mathcal{S}^0 \cup \{\widehat{p}_j\}$, and let $\mathcal{S}^1 \subseteq \mathcal{S}^0 \cup \{\widehat{p}_j\}$ be the support points of this ellipsoid. Then repeat. Analogously to the MEB algorithm, in each iteration k where the optimum has not been attained, the farthest point $\widehat{p}_j$ must satisfy $\widehat{p}_j \notin \mathcal{E}^k$ and therefore $\widehat{p}_j \in \mathcal{S}^{k+1}$. Furthermore, $\operatorname{vol}\mathcal{E}^{k+1} > \operatorname{vol}\mathcal{E}^k$, which implies that termination is guaranteed in a finite number of iterations. The question of how to compute the MVEE of each candidate support set is not addressed directly in [115]. Silverman and Titterington [113] present a specialization of this algorithm for the planar case, and provide detailed procedures to find the minimum ellipse for each candidate support set, which must contain either 3, 4, or 5 points.

On the other hand, Post [97] gives an algorithm that can be viewed as an analogue of the primal MEB algorithm by Fischer et al. [50]. Each iterate consists of a candidate support set $\mathcal{S}^k \subseteq \mathcal{P}$ containing between $d+1$ and $d(d+3)/2$ points, as well as the smallest ellipsoid $\mathcal{E}^k$ enclosing all of $\mathcal{P}$ while having the points of $\mathcal{S}^k$ on its boundary. Unless $\mathcal{E}^k = \text{MVEE}(\mathcal{P})$, one point in $\mathcal{S}^k$ that cannot be on the boundary of the MVEE is eliminated from further consideration. Then the ellipsoid is shrunk (and moved) while still enclosing all of $\mathcal{P}$ and maintaining the rest of $\mathcal{S}^k$ on its boundary. Either this shrinking process ends by the ellipsoid coinciding with MVEE($\mathcal{P}$), or it is interrupted by a new point entering the shrinking boundary. In the latter case, the new point replaces the eliminated point, which gives a new candidate support set $\mathcal{S}^{k+1}$, and the shrunk ellipsoid becomes the next candidate ellipsoid $\mathcal{E}^{k+1}$. Since each step of this procedure takes $O(n)$ time (for fixed d), and one point is eliminated from further consideration in each step, this algorithm has a $O(n^2)$ time complexity. The dependence on the dimension, however, is exponential.

Like the MEB problem, the MVEE problem belongs to the abstract class of LP-type problems, and can thus be solved in expected linear time (in n) using the same randomized algorithms [87, 34], or in deterministic linear time using derandomizations thereof [31, 28].

Welzl's adaptation to the MVEE problem [139] of the randomized LP algorithm by Seidel [108] computes the MVEE in expected $O(\delta\delta!n)$ time, where $\delta := d(d+3)/2$. Formulae necessary for realizing this algorithm in two dimensions are given by Gärtner and Schönherr [57]. Deterministic linear-time algorithms can also be designed in a different abstract framework due to Dyer [42].

3.3.2 Approximation Algorithms

In applications dealing with higher dimensions, it is usually necessary to settle for approximate solutions. Given a parameter $\epsilon > 0$, an ellipsoid $\mathcal{E}$ is called a $(1+\epsilon)$-approximation of MVEE($\mathcal{P}$) if it satisfies $\mathcal{E} \supset \mathcal{P}$ and $\operatorname{vol}\mathcal{E} \leq (1+\epsilon)\operatorname{vol}\text{MVEE}(\mathcal{P})$. Let u be a dual feasible solution and let $\mathcal{E}_{M,0}$ be a corresponding ellipsoid defined via (3.26). Then the following $(1+\eta)$-*approximate* version of the KKT optimality condition (3.19) can be defined:

$$\widehat{p}_i^{\mathrm{T}} M \widehat{p}_i \leq 1+\eta, \quad i = 1, \ldots, n. \tag{3.31}$$

Clearly, if $\mathcal{E}_{M,0}$ satisfies this condition then $\sqrt{1+\eta}\mathcal{E}_{M,0} \supset \widehat{\mathcal{P}}$; that is to say, $\mathcal{E}_{M,0}$ can then be scaled about its center by $\sqrt{1+\eta}$ to yield a primal feasible ellipsoid. (The resulting ellipsoid is at the same time a $\sqrt{(1+\eta)(d+1)}$-rounding of conv($\widehat{\mathcal{P}}$) [71].) Since such a scaling corresponds to a multiplication of M by $(1+\eta)^{-1}$, it can be seen from the volume formula (3.3) that the scaled ellipsoid gets a volume $(1+\eta)^{(d+1)/2}$ times that of $\mathcal{E}_{M,0}$. Thus, if $\eta = \eta_1$ is used, where

$$\eta_1 := (1+\epsilon)^{2/(d+1)} - 1, \tag{3.32}$$

then the enlarged ellipsoid is a $(1+\epsilon)$-approximation of MVEE($\widehat{\mathcal{P}}$).

Now let $\mathcal{E}_{Q,c}$ be the intersection of $\mathcal{E}_{M,0}$ with the hyperplane $\mathcal{H}$ of (3.9). It can be seen from (3.26), (3.28), and (3.27) that a scaling of $\mathcal{E}_{M,0}$ by $\sqrt{1+\eta}$ corresponds to a scaling of $\mathcal{E}_{Q,c}$ by the factor $\sqrt{((d+1)(1+\eta)-1)/d}$ about c, which multiplies its volume by $(((d+1)(1+\eta)-1)/d)^{d/2}$. Thus, if instead $\eta = \eta_2$ is used, where

$$\eta_2 := (d(1+\epsilon)^{2/d} + 1)/(d+1) - 1, \tag{3.33}$$

then the intersection of $\sqrt{1+\eta}\mathcal{E}_{M,0}$ with $\mathcal{H}$ provides a $(1+\epsilon)$-approximation of MVEE($\mathcal{P}$). However, since $\eta_1 < \eta_2$ for $\epsilon > 0$ (which can be seen, e.g., by considering that equality holds for $\epsilon = 0$, and that $\frac{\partial \eta_1}{\partial \epsilon} < \frac{\partial \eta_2}{\partial \epsilon}$

holds for $\epsilon \geq 0$), this is true also with $\eta = \eta_1$. In other words, if a candidate solution $\mathcal{E}_{M,0}$ satisfies (3.31) with $\eta = \eta_1$, then this implies that a $(1+\epsilon)$-approximation of MVEE($\mathcal{P}$) has been found.

Barnes [12] develops a coordinate ascent algorithm for the dual of problem ($\mathbf{P}'_{\mathrm{E}}$) with the center fixed, and a gradient technique to find the optimal center. Titterington [127] gives a simple numerical method for the D-optimal design problem based on the recursion

$$u_i^{k+1} = \widehat{p}_i^{\mathrm{T}} M^k \widehat{p}_i u_i^k, \quad i = 1, \ldots, n, \tag{3.34}$$

(cf. (2.25) on page 28 of this thesis) and proves monotonicity and convergence. This type of algorithm is commonly called *multiplicative*, since it scales all the weights u_i in each iteration. Other multiplicative algorithms are studied in, e.g., [127, 128, 116, 86, 60, 131].

In addition, a number of interior-point methods have been proposed. The path-following Newton method of Nesterov and Nemirovskii [93], when combined with a preprocessing step proposed by Khachiyan [71], computes a $(1+\epsilon)$-approximation of the MVEE in $O(n^{3.5} \log(n/\log(1+\epsilon)))$ time. Another algorithm in this category is due to Sun and Freund [118]. Vandenberghe et al. [135] give an interior-point algorithm for the more general problem of maximizing the determinant of a matrix subject to linear matrix inequalities. The MVEE problem can also be transformed to the problem of finding the maximal ellipsoid inscribed in a given d-dimensional polytope defined by n linear inequalities [72]. Interior-point methods for this problem are studied in, e.g., [72, 93, 147, 148].

Since the computational burden of these approaches is prohibitive for large n, an *active set* strategy is developed as a complement in [118]. The active set is a sample of $n' \ll n$ points from $\mathcal{P}$ that appear likely to be support points, on which the original algorithm is invoked as a subroutine to generate a trial MVEE. As long as the MVEE of the active set does not enclose the whole point set $\mathcal{P}$, the active set is repeatedly refined by adding and/or removing points. For example, in the algorithm of Sun and Freund, each Newton step requires forming and factorizing an $n \times n$ matrix. The active set strategy reduces this to the forming and factorizing of an $n' \times n'$ matrix. This method can also be referred to as a *subspace ascent* method [59], since the dual weights of the active points define a subspace in which the solution is improved while the remaining weights are kept fixed.

Algorithm 3.1. Generic $(1+\epsilon)$ MVEE algorithm.

```
Input:  $\mathcal{P} := \{p_1, \dots, p_n\} \subset \mathbb{R}^d$ s.t. $\mathcal{P} = -\mathcal{P}$ and $\operatorname{aff}(\mathcal{P}) = \mathbb{R}^d, \epsilon > 0$
Output: $\mathcal{E}_{Q,c}$ s.t. $\mathcal{E}_{Q,c} \supset \mathcal{P}$ and $\operatorname{vol}\mathcal{E}_{Q,c} \leq (1+\epsilon)\operatorname{vol}\operatorname{MVEE}(\mathcal{P})$
 1: $\eta \leftarrow (d(1+\epsilon)^{2/d} + 1)/(d+1) - 1$
 2: Initialize $M$ and $\mathcal{K}$
 3: loop
 4:     $j \leftarrow \arg\max_{i=1}^{n} \widehat{p}_i^{\mathrm{T}} M \widehat{p}_i$
 5:     $\kappa \leftarrow \widehat{p}_j^{\mathrm{T}} M \widehat{p}_j$
 6:     if $\kappa \leq 1 + \eta$ then
 7:         return $\sqrt{\kappa}\mathcal{E}_{M,0} \cap \mathcal{H}$
 8:     end if
 9:     $\mathcal{K} \leftarrow \mathcal{K} \cup \{p_j\}$
10:     Update $M$ using $\mathcal{K}$
11: end loop
```

Like the dual MEB problem $(\mathbf{D}_{\mathrm{B}})$, problem $(\mathbf{D}_{\mathrm{E}})$ seeks to maximize a concave and continuously differentiable function over the unit simplex. Not surprisingly, therefore, a series of algorithms based on the Frank–Wolfe method [53], closely resembling those discussed in Section 2.3.2, have been developed, initially for the D-optimal design problem, and later studied in the context of the MVEE problem. In Algorithm 3.1, a strategy analogous to that of Algorithm 2.1 is outlined for computing a $(1+\epsilon)$-approximation of the MVEE as well as an ϵ-core-set $\mathcal{K}$. In this context, a subset $\mathcal{K} \subseteq \mathcal{P}$ is called an ϵ-core-set if there exists a scaling factor s such that $s\operatorname{MVEE}(\mathcal{K}) \supset \mathcal{P}$ and $\operatorname{vol} s\operatorname{MVEE}(\mathcal{K}) \leq (1+\epsilon)\operatorname{MVEE}(\mathcal{P})$. Although it is not visible in the pseudocode, the algorithms in this category generally maintain a dual feasible solution u, which defines the current ellipsoid $\mathcal{E}_{M,0}$ via (3.26) and the core-set as $\mathcal{K} = \{p_i \in \mathcal{P} : u_i > 0\}$.

The partial derivative of the objective function (3.23) with respect to u_ℓ, $\ell = 1, \dots, n$, is

$$\begin{aligned}\frac{\partial}{\partial u_\ell} \operatorname{logdet} V(u) &= \operatorname{trace}\left(V(u)^{-1}\widehat{p}_\ell\widehat{p}_\ell^{\mathrm{T}}\right)\\ &= \widehat{p}_\ell^{\mathrm{T}} V(u)^{-1}\widehat{p}_\ell,\end{aligned}$$

which is proportional to $\widehat{p}_\ell^{\mathrm{T}} M \widehat{p}_\ell$. Thus, similarly to before, the index j computed on Line 4 names the largest component of the gradient of the objective function. Furthermore, e_j is the corner of the unit simplex that maximizes a linearization of problem ($\mathbf{D}_\mathrm{E}$) at u, where $e_j \in \mathbb{R}^n$ has a 1 in the j-th position and 0 everywhere else. The point $\widehat{p}_j$ is the currently farthest point from the origin, measured in the ellipsoidal norm $\|\cdot\|_M$, and p_j is the farthest point from c in the norm $\|\cdot\|_Q$ (see Section 3.2.1). If Line 4 is implemented in a naïve way, where the full quadratic form is evaluated for $i = 1, \ldots, n$, computing j takes $\Theta(d^2 n)$ time per iteration. However, in several of the algorithms considered here, the update to u in each iteration translates to a rank-1 or rank-2 update of M. Taking this into account allows each distance to be computed incrementally in $\Theta(d)$ time from the distance of the previous iteration, for a total cost of $\Theta(dn)$ time per iteration [130, 36].

The termination test on Line 6 makes sure that the $(1 + \eta)$-approximate optimality condition (3.31) is satisfied. Since Algorithm 3.1 uses the value $\eta = \eta_2$ of (3.33), the ellipsoid returned on Line 7 is a $(1 + \epsilon)$-approximation of MVEE($\mathcal{P}$). (As mentioned, $\eta = \eta_1$ also gives a correct result; however, using $\eta = \eta_2$ might be preferable, as it gives a slightly less conservative termination criterion.) The number of iterations required to fulfill this depends on how the initial solution is computed on Line 2, as well as how it is updated in each iteration on Line 10.

Wynn [142] gives an algorithm for the D-optimal design problem that shifts the current iterate u toward the vertex e_j of the unit simplex by a distance determined by the current iteration number. For the initial solution u^0, ℓ weights are selected and set to the value $1/\ell$. Then the solution is shifted by $1/(\ell + k)$, where k is the current iteration count. A similar algorithm independently developed by Fedorov [48] uses a step length that maximizes the growth of the objective function. This optimal step length is shown to be given by a closed formula. Atwood [9] presents a modified version of the algorithm by Fedorov that uses Wolfe's away steps [141]. In an analogous fashion as in Yıldırım's second MEB algorithm [144] discussed in Section 2.3.2, a choice is made in each iteration between adjusting u toward e_j or away from e_{j_-}, where $i = j_-$ minimizes $\widehat{p}_i^\mathrm{T} M \widehat{p}_i$ subject to $u_i > 0$.

Khachiyan's algorithm [71] for the MVEE problem is, apart from the initialization and termination criterion, equivalent to the algorithm of Fedorov [48]. Khachiyan's analysis shows that if an initial solution u^0 is used where $u_i^0 = 1/n$, $i = 1, \ldots, n$, then the algorithm finishes in $O(d(\eta^{-1} + \log d + \log\log n))$ iterations. Todd's analysis [129] improves this bound to $O(d(\eta^{-1} + \log\log n))$.

Kumar and Yıldırım [75] employ a volume approximation algorithm by Betke and Henk [16] to initially select $\max(2d, n)$ weights that are all set to $1/\max(2d, n)$ in the starting solution u^0. Selecting the weights in this way guarantees a certain initial duality gap, and this reduces the number of iterations to $O(d(\eta^{-1} + \log\log d))$ (using Todd's analysis [129]). The algorithm also computes a core-set $\mathcal{K} = \{p_i \in \mathcal{P} : u_i > 0\}$, whose cardinality has the same bound as the number of iterations.

Todd and Yıldırım [130] further augment the algorithm with Atwood's away steps [9]. Since away steps reduce weights in the solution u, sometimes to zero, this has the potential to reduce the size of the core-set. Away steps also enable the algorithm to employ, in addition to the $(1 + \eta)$-approximate optimality condition (3.31), the following $(1 - \eta)$-approximation of the complementary slackness condition (3.22):

$$u_i > 0 \Rightarrow \widehat{p}_i^{\mathrm{T}} M \widehat{p}_i \geq 1 - \eta, \quad i = 1, \ldots, n. \tag{3.35}$$

The number of iterations as well as the size of the core-set remains bounded by $O(d(\eta^{-1} + \log\log d))$. Ahipaşaoğlu et al. [2] prove local linear convergence when incorporating away steps.

Whereas all the methods above generate the next iterate by increasing one weight (either u_j or $u_{j\_}$) and then scaling the remaining weights to maintain dual feasibility, the *vertex exchange* method by Böhning [18] increases u_j and decreases $u_{j\_}$ simultaneously by equal amounts, which makes any rescaling unnecessary. Such an update moves u in parallel with the edge connecting the corners e_j and $e_{j\_}$ of the unit simplex, and gives rise to a rank-2 modification of the matrix M. Cong et al. [36] combine the Kumar–Yıldırım initialization scheme with Böhning's updates, and prove the same asymptotic time complexity and size of the core-set.

The *cocktail* algorithm by Yu [145] computes each iterate through a combination of the update schemes of Fedorov and Böhning with the multiplicative algorithms. The recent *randomized exchange* algorithm by Harman et al. [59] combines Böhning's vertex exchange updates with a subspace ascent step.

3.3.3 Acceleration Techniques

As mentioned previously, the search for the farthest point on Line 4 of Algorithm 3.1 takes $\Theta(d^2 n)$ time per iteration, or $\Theta(dn)$ time if incremental updates of the distances are possible. Even in the latter case, this search tends to dominate the execution time of the algorithms discussed here. Therefore, methods to lower the cost of the searches are of much benefit.

Harman and Pronzato [60] derive an inequality that must be satisfied by the support points of a D-optimal design or MVEE. According to the same principle as that used by Ahipaşaoğlu and Yıldırım in the context of the MEB problem [4] (see Section 2.3.3), this enables the elimination of inessential points from the input to speed up subsequent searches for the farthest point. In the notation of Algorithm 3.1, any point p_i satisfying

$$\widehat{p}_i^{\mathrm{T}} M \widehat{p}_i < 1 + \frac{\delta}{2} - \frac{\sqrt{\delta\big(4 + \delta - 4/(d+1)\big)}}{2}, \tag{3.36}$$

where $\delta := (d+1)(\kappa - 1)$, cannot be a support point of MVEE($\mathcal{P}$) and can thus be removed from $\mathcal{P}$.

Galkovskyi et al. [54] introduce an elimination heuristic applicable to certain algorithms for the MVEE problem. The key idea is to compute the convex hull of each candidate support set, and eliminate any input points that fall in its interior, as these points clearly cannot be on the boundary of the MVEE. Large performance gains are reported for two-dimensional problem examples. However, the approach is unlikely to be practical for more than three dimensions, due to the prohibitive cost of computing the convex hull.

3.4 Applications

As mentioned in Chapter 1, enclosing ellipsoids find use in many of the same practical applications as do enclosing balls. Due to their more flexible description, ellipsoids can provide tighter approximations of the underlying data, albeit at increased computational costs. This is expressed in a formal way by the property (3.4), which states that the MVEE of any polytope approximates it to within a guaranteed accuracy. Another useful property of the MVEE in these applications is its affine invariance, which states that MVEE($T(\mathcal{X})$) = T(MVEE($\mathcal{X}$)) for any set $\mathcal{X}$ and affine map T [21].

Using ellipsoids as bounding volumes in collision detection [101, 84, 80, 25] and ray tracing [20] can reduce the number of false positives compared to using other geometric shapes, such as balls or axis-aligned boxes. Furthermore, approaches for classification and novelty detection, similar to those discussed in Section 2.4.2, have been devised that use ellipsoids as less conservative boundary descriptions in the feature space [103, 123, 124]. In clustering, the volume of the MVEE has proven to be an effective criterion to rate the quality of candidate cluster allocations [111]. In robust statistics, the MVEE of a sample can be used to trim off outliers, as the outliers will end up as support points of the MVEE [128]. The MVEE can also provide a direct tool for parameter estimation in statistics [5].

The next subsection gives a brief overview of the relation between the MVEE problem and the subject of optimal experimental design. We remark that the exposition follows closely Section 1.3 of Todd's monograph on the MVEE problem [129].

3.4.1 Example: Optimal Design of Experiments

A linear regression model assumes that the response variable y depends on the predictor variable x according to the linear relationship

$$y = \theta_1 f_1(x) + \cdots + \theta_k f_k(x) + \epsilon, \tag{3.37}$$

where the θ_j are k unknown parameters, the f_j are k known functions, and ϵ is a random variable following a normal distribution with zero mean and standard deviation σ. In an experiment aimed at estimating the parameters θ_j, a series of measurements are carried out with different values of x, and then a regression estimator is applied to these measurements to yield an estimate. The term ϵ in the model represents measurement error. The *design* of an experiment determines which values of x, called *design points*, should be used, as well as how many observations to take at each such point. A design is called *optimal* if the design points are selected in an optimal way w.r.t. some statistical criterion. The benefit of such a design over a suboptimal one is that it achieves higher-precision estimates with a given number of measurements.

For simplicity, we assume that the design space $\mathcal{X}$ from which x is drawn is discretized as n distinct points $x_1, \ldots, x_n$. Then a design can be specified as $w := (m_1/M, \ldots, m_n/M)^{\mathrm{T}}$, where each $m_i \geq 0$ is an integer giving the number of measurements to take at point x_i, and $M := \sum_{i=1}^{n} m_i$. Let $\bar{y} := (\bar{y}_1, \ldots, \bar{y}_n)^{\mathrm{T}}$ be a given a set of observations, where $\bar{y}_i$ is the mean outcome of the m_i measurements taken at point x_i. Denote by X the n-by-k matrix with the ij-th element equal to

$f_j(x_i)$, and let W be the diagonal matrix having w as its diagonal. The *weighted least-squares* estimate $\widehat{\theta} := (\widehat{\theta}_1, \ldots, \widehat{\theta}_k)^{\mathrm{T}}$ of the vector $\theta := (\theta_1, \ldots, \theta_k)^{\mathrm{T}}$, using the values m_i/M as weights, is then the minimizer of $\|W^{1/2}(\bar{y} - X\theta)\|^2$, and is given by

$$\widehat{\theta} = (X^{\mathrm{T}}WX)^{-1}X^{\mathrm{T}}W\bar{y}. \tag{3.38}$$

What is the quality of this estimate, and how is the quality affected by the choice of w? Firstly, it can be seen from (3.37) that $\bar{y}$ is a sample of the random variable $X\theta + (MW)^{-1/2}\vec{\epsilon}$, where $\vec{\epsilon}$ is an n-dimensional random variable with each element normally distributed with zero mean and variance σ^2 (note that the i-th element of $(MW)^{-1/2}\vec{\epsilon}$ follows the distribution of sample means when taking samples of size m_i of the last term in (3.37)). This, in turn, implies that $\widehat{\theta}$ is a sample of the random variable $\widehat{\Theta} := \theta + M^{-1/2}(X^{\mathrm{T}}WX)^{-1}X^{\mathrm{T}}W^{1/2}\vec{\epsilon}$, which has mean θ and variance

$$E[(\widehat{\Theta} - \theta)(\widehat{\Theta} - \theta)^{\mathrm{T}}] = \sigma^2 M^{-1}(X^{\mathrm{T}}WX)^{-1}. \tag{3.39}$$

Different choices of w give rise to different values of this variance, and the goal is to achieve an optimal variance according to some criterion. One such criterion is D-optimality, which seeks to minimize the determinant of (3.39) or, equivalently, maximize $\operatorname{logdet}(X^{\mathrm{T}}WX)$. Other choices are A-optimality, which minimizes the trace of (3.39), or E-optimality, which maximizes the smallest eigenvalue of $X^{\mathrm{T}}WX$. Another criterion known as G-optimality, which minimizes the maximum value of $x_i^{\mathrm{T}}(X^{\mathrm{T}}WX)^{-1}x_i$ over $i = 1, \ldots, n$, was shown by Kiefer and Wolfowitz [73] to be equivalent to D-optimality.

Finding an exact design w such that each component can be written as m_i/M, where m_i is an integer, is a hard integer programming problem. It is therefore more practical to compute an approximate design $u := (u_1, \ldots, u_n)^{\mathrm{T}}$ that is only required to satisfy $u_i \geq 0$ for $i = 1, \ldots, n$ and $\sum_{i=1}^{n} u_i = 1$. Then each u_i gives roughly the optimal proportion of the measurements to take at x_i. The D-optimal design problem can then be stated as

$$\max_u \quad \operatorname{logdet}(X^{\mathrm{T}}UX) \tag{3.40}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} u_i = 1, \tag{3.41}$$

$$u_i \geq 0, \quad i = 1, \ldots, n, \tag{3.42}$$

where U is a diagonal matrix with its entries given by u. Clearly, this problem is the same as problem ($\mathbf{D}_{\mathrm{E}}$) with each $\widehat{p}_i$ replaced by the i-th column of X^{T}. In other words, computing the D-optimal design for a discrete design space $\{x_1, \ldots, x_n\}$ is equivalent to finding the centered MVEE of the set $\{(f_1(x_i), \ldots, f_k(x_i))^{\mathrm{T}} : i = 1, \ldots, n\}$.

3.5 Related Problems

We conclude this chapter with brief discussions of some relevant problems related to the basic variant of the MVEE problem treated so far.

Firstly, as with the MEB, it can be of interest to compute the MVEE of collections of other types of geometric objects than points. One example is finite sets of ellipsoids, to which the first-order algorithms discussed in Section 3.3.2 can be adapted if the search for the farthest point in each iteration is modified to search for the farthest point inside each of the given ellipsoids [143, 66]. Such a modification does not change the bound on the number of iterations, and the time complexity remains linear in n (now denoting the number of ellipsoids). However, it does increase the dependence on d due to the higher cost of the farthest-point search. For another example of more general input primitives, see [6], where efforts are presented toward finding the smallest enclosing ellipse of sets of freeform planar curves using Welzl's algorithm [139].

Another variation of the problem is that of finding the MVEE whose axes are constrained to be aligned with the coordinate axes of the space. Although such an *axis-aligned* MVEE generally provides a looser fit than the freely oriented one, its simpler description enables cheaper operations, such as intersection tests. At the same time, it does provide a tighter fit than, e.g., the MEB. This problem is amenable to first-order approximation strategies similar to those discussed in Section 3.3.2, but with the added complication that the line search in each iteration does not have a closed-form solution [76, 68].

A problem related to the centered MVEE problem, and whose dual is equivalent to a generalization of the D-optimal design problem, is that of computing an ellipsoidal cylinder passing through the origin while enclosing a given point set, with the objective of minimizing the area (or volume) of its intersection with a fixed s-dimensional subspace [115, 126]. The problem of computing a not necessarily centered such cylinder, which can be reduced to the centered problem using the lifting trick discussed earlier, finds utility in, e.g., collision detection of moving objects [3]. The corresponding problem in optimal experimental design is called the D_s-optimal design problem, and arises in contexts where $s \leq k$ of the parameters $\theta_1, \ldots, \theta_k$ are of interest. Frank–Wolfe-type algorithms are given by Ahipaşaoğlu and Todd [3, 129].

Chapter 4

Contributions

The overarching goal of this research has been to develop efficient solution methods for the MEB and MVEE problems, as well as to generate theoretical insights that can aid in the design of such methods. The primary focus has been on large-scale instances of the problems, involving data sets that contain large numbers of points while the number of dimensions is relatively low. For this reason, the research efforts have mainly centered around first-order methods adhering to the strategies sketched in Algorithms 2.1 and 3.1 of Sections 2.3.2 and 3.3.2, respectively, since such methods generally allow for running times linear in the number of points. Although the research has not been motivated by a specific application, due to the ever-increasing amounts of data used today in applications such as computer graphics, data mining, and machine learning, contributions in this area should be of much benefit.

While the developed methods rest on rigorous theoretical analysis of the problems, a stronger emphasis has in general been put on their practical efficiency than on their theoretical worst-case behavior. This means that the various techniques presented in the thesis are mostly of a heuristic nature, in the sense that they produce palpable effects on performance by exploiting characteristics common to many problem instances, while lacking guarantees to do so in the absence of these characteristics. In difficult cases in which they fail to be effective, such techniques inevitably incur certain degrees of performance degradation compared to either not using them or to using other known techniques. For the most part, however, the heuristics developed in this thesis have shown to incur relatively small such performance hits. In the cases where the asymptotic time or memory complexities are affected by the use of a certain technique, this is clearly stated and discussed in the context of alternative approaches.

The results of this research are presented in six academic papers that are included in Part II of this thesis, labeled as A–F according to the chronological order in which they were written and published. Before each of the papers are discussed in more detail, the highlights of the contributions are listed below, categorized as either theoretical results, new algorithms, or acceleration techniques. Under the category of theoretical results, the main highlight is:

- In Paper B, lower bounds on the distance from the center of a sub-optimal trial MEB to the boundary of the true MEB are derived. We prove that if the ball in question corresponds to a dual feasible solution, then a less conservative bound than that by Ahipaşaoğlu and Yıldırım [4], discussed in Section 2.3.3 of this thesis, can be derived.

The category of new algorithms includes:

- In Paper A, an algorithm for computing a $(1+\epsilon)$-approximation of the MEB in any number of dimensions is proposed. The algorithm is shown to be highly competitive to state-of-the-art algorithms, both in terms of its asymptotic time complexity and its efficiency in practice.

- In Paper D, an algorithm is developed for computing the exact MEB of a three-dimensional point set under an external memory model of computation. Under this model, it is assumed that the input data set is too large to fit in main memory in its entirety and therefore has to be fetched in parts from much slower secondary storage during the computations.

- In Paper F, an efficient $(1 + \epsilon)$-approximation algorithm for the MVEE problem in general dimensions is proposed. The algorithm derives its efficiency from an active set strategy, which is shown empirically to be superior to previous such strategies found in the literature.

Finally, under the category of acceleration techniques, the main highlights are:

- In Paper B, the elimination heuristic by Ahipaşaoğlu and Yıldırım [4] (see Section 2.3.3) for the computation of MEBs is extended to use the less conservative bounds derived in the paper. Our experimental evaluation finds that when incorporating the heuristic into the algorithm of Paper A as well as the two algorithms by Yıldırım [144] (see Section 2.3.2), the tighter bounds give up to 2× speedups over the previous bounds.

- In Paper C, the distance filtering idea of Nielsen and Nock [94] to accelerate the farthest-point queries (see Section 2.3.3) is developed further. A modified heuristic based on the triangle inequality is shown empirically to improve efficiency over the heuristic based on the Cauchy–Schwarz inequality.

- In addition to the filtering heuristic, Paper C proposes a parallel implementation of the farthest-point queries on a graphics processing unit (GPU). The implementation is amenable to auto-tuning, which is a technique to dynamically adjust the implementation parameters for maximum performance across different GPU architectures.

- In Paper D, a filtering heuristic is developed that applies the triangle inequality to blocks of data in secondary storage in order avoid unnecessary fetches of these blocks. The heuristic is shown to be able to reduce the number of I/O transactions close to the optimum of only one transaction per block.

- In Paper E, the distance filtering idea is carried over to the computation of MVEEs, to improve the performance of algorithms based on the strategy outlined in Algorithm 3.1.

4.1 Research Process

The above contributions have been developed through a process based on literature studies as well as practical experiments. Literature studies were initially conducted to gain broad knowledge of the research area and to identify open questions, but have also remained a constant component as the scope of the research has broadened and the focus has shifted throughout the process. Implementing known methods from the literature as well as examining the source code of existing implementations have also been valuable means to facilitate deeper understanding of the studied topics. Ideas for specific research question and hypotheses to pursue have developed either from these activities or, frequently, as byproducts of ongoing studies. Experiments, in turn, have been used not only to demonstrate the strengths and weaknesses of the proposed methods in the final published works, but throughout all stages of each project to quickly gauge whether a particular idea or hypothesis has merit or not. Indeed, hypotheses have frequently undergone multiple revisions based on the outcomes of these experiments.

As the main focus has generally been on the practical efficiency of the studied methods, a prominent part of this work has been to perform timing measurements, using both commercial profiling tools and specially designed benchmarking programs. In general, it is a challenge to obtain reliable measurements of execution time. This is particularly the case for very short-running computations, since their execution times are more sensitive to interference from other processes running concurrently on the same system. Another potential source of problems when dealing with such computations is low resolution of the system clock used to take the measurements. The primary technique used throughout this work to address these issues has been to take the mean execution time of many identical repetitions of each run, by taking a single measurement where the same algorithm is invoked repeatedly on the same input in a loop, and then dividing the total execution time by the number of repetitions. This way, the running time is scaled up from, say, milliseconds to the order of seconds, so that the available clock resolution can be utilized more fully. Another important aspect to consider is that execution times are strongly affected by details of the implementation. To avoid bias, it is therefore important to ensure that equal efforts have been spent optimizing the code of all methods included in a trial. It is, for the same reason, also important to include implementation-independent measures of performance, e.g., iteration counts, in a comparison between different methods.

For an empirical evaluation to be relevant, test data must be selected that faithfully represent “typical” real-world scenarios. Furthermore, to ensure that the results of a study are reproducible, the test data should be made available to the research community and the details of the experimental setup should be clearly described. The test cases used throughout this work include 3D models, some of which are well-known and are publicly available online, as well as randomly generated data sets. In order for the benchmarks to paint a fairly complete picture, several different statistical distributions have been employed when generating the data sets, and the measurements have been repeated multiple times with different data sets. In Paper F, the data sets are generated in an identical manner as in prior literature [2, 118]. Yet another concern is how to report the results of a study in a clear way that answers the questions posed and does not introduce bias [51, 117]. Given the significant effects on performance generally seen in the included empirical evaluations, we believe that the risk that any crucial errors have been made in this regard is minimal.

4.2 Summary of Papers

Paper A

Thomas Larsson and Linus Källberg. *Fast and Robust Approximation of Smallest Enclosing Balls in Arbitrary Dimensions.* Computer Graphics Forum, 32(5) – Symposium on Geometry Processing 2013.

In this paper, a new algorithm is proposed for computing a $(1+\epsilon)$-approximation of the MEB in $O(dn/\epsilon + d/\epsilon^3)$ time. The algorithm uses the general strategy outlined in Algorithm 2.1, and computes an ϵ-core-set of size $O(1/\epsilon)$. The candidate ball is initialized from a coarse approximation of the diameter of the point set. Then, after a new point has been inserted into the core-set in each iteration, the ball is updated by invoking a subroutine that computes a $(1+\epsilon/2)$-approximation of the MEB of the updated core-set. This subroutine also follows the strategy of Algorithm 2.1, except that it does not build its own core-set. Here the solution is updated in each iteration using the cheap geometric operation by Tian [125]. Interestingly, as is later shown in the appendix of Paper B, this update operation is simply the geometric consequence of the update used in Yıldırım's dual algorithms [144], which moves the current iterate toward the corner of the unit simplex corresponding to the farthest point (see Section 2.3.2). This means that although the algorithm in Paper A is described in an entirely geometrical way, it can be interpreted as a dual algorithm.

The above strategy of repeatedly solving the problem for only the points currently in the core-set can be called an active set strategy, and is also used, in a more elaborate form, in the MVEE algorithm proposed in Paper F. If the algorithm is interpreted as a dual algorithm, the approach can also be called a subspace ascent method [59], since the weights of the points not in the core-set are kept fixed (at zero) while the solution is improved in the subroutine over the subspace defined by the remaining weights. A simplified version of the algorithm is also described, where the subroutine is replaced by a single application of the Tian update operation. This modification reduces the time complexity to $O(dn/\epsilon)$, but appears to negatively affect performance in practice in most cases. An additional consequence of the result given in the appendix of Paper B is that this simplified algorithm is equivalent to Yıldırım's first algorithm, which does not use away steps.

The paper also introduces the concept of a "viable" ball, which is an approximation $\mathcal{B}_{c,r}$ of the MEB having at least a hemisphere fully enclosed in the MEB, i.e., $\sqrt{\|c - c^*\|^2 + r^2} \leq r^*$. Intuitively, viability is a certificate that a candidate ball is not too far off from the sought solution, and the analysis of the algorithm is based on showing that all the intermediate solutions satisfy the viability property. The concept is adopted from Tian [125], who uses the term "bubble" to the describe this type of ball.

A clarification of my contributions to this paper, as I am not its main author: I developed Lemmas 1–3 and their proofs, which establish the correctness, convergence, as well as the time complexities of the algorithms proposed in the paper.

Paper B

Linus Källberg and Thomas Larsson. *Improved Pruning of Large Data Sets for the Minimum Enclosing Ball Problem.* Graphical Models, 76(6), 2014.

In this paper, we show that if a ball $\mathcal{B}_{c,r}$ is related to a dual feasible solution u through (2.23) and (2.24) (page 22), then the lower bound derived by Ahipaşaoğlu and Yıldırım [4] on the distance from c to the boundary of MEB($\mathcal{P}$) can be tightened. The improved bound gives the following condition for when a point p_i lies in the interior of MEB($\mathcal{P}$):

$$\|p_i - c\| < \left(\sqrt{1 + \epsilon + \epsilon^2/2} - \sqrt{\epsilon + \epsilon^2/2}\right)r, \tag{4.1}$$

where, as before, ϵ is such that $\mathcal{B}_{c,(1+\epsilon)r} \supset \mathcal{P}$. Condition (4.1) is a slight paraphrase of the condition that results by combining Theorems 2 and 4 in Paper B. A comparison of the terms inside the parenthesis to the corresponding terms in (2.26) on page 32 immediately gives that the right-hand side of (4.1) is larger and thus gives a less conservative condition.

In addition, the paper includes a computational study in which the effects of using the improved bounds in the pruning heuristic by Ahipaşaoğlu and Yıldırım are assessed. When incorporating the heuristic into Yıldırım's two algorithms [144] as well as the main algorithm from Paper A, performance improvements of up to 2× are seen when using the new bounds. Compared to not using the heuristic, speedups of up to two orders of magnitude are seen. However, the effects vary depending on the number of iterations performed and the statistical distribution of the test data sets.

Since the publication of Paper B, Pronzato [98] has independently derived a bound that can be shown, by a short calculation, to coincide with the bound in the right-hand side of (4.1). Pronzato additionally proves that the bound cannot be improved further.

Paper C

Linus Källberg and Thomas Larsson. *Faster Approximation of Minimum Enclosing Balls by Distance Filtering and GPU Parallelization.* Journal of Graphics Tools, 17(3), 2015.

The contribution of this paper is twofold. Firstly, we give alternatives to the distance filtering technique proposed by Nielsen and Nock [94] (see Section 2.3.3). We argue that the effectiveness of our techniques, which are based on the triangle inequality as opposed to the Cauchy–Schwarz inequality, is less sensitive to how the points are configured in space, especially in higher dimensions. Our experiments seem to confirm this: While Nielsen and Nock's filtering method outperforms our methods in low dimensions (on test cases selected to be reasonably favorable for their method), its effectiveness diminishes drastically as the dimension increases. Our filtering approach, on the other hand, remains effective in dimensions at least as high as 10,000.

These filtering heuristics address certain drawbacks of pruning heuristics such as those studied in Paper B. Firstly, since the condition for eliminating a point depends on the approximation quality of the current solution, the pruning is usually fairly ineffective during the initial iterations of the algorithm. This can lead to meager positive effects on performance in cases where the algorithm does not require many iterations. Secondly, pruning can be expected to be less effective for high-dimensional problems. Note that the pruning conditions (2.26) and (4.1) can be interpreted geometrically as testing whether a point falls inside a shrunken copy of the current candidate ball. As the number of dimensions increases, the ratio of the volume of this ball to the volume of the MEB decreases toward zero, an example of a phenomenon commonly referred to as the "curse of dimensionality". This means that it becomes increasingly unlikely for any points in the input to satisfy the pruning condition. On the other hand, while the filtering condition too can be interpreted as testing a point for containment in a ball, the ball in this case has a much less conservative radius, which gives a larger volume ratio.

In the second part of Paper C, we study how GPUs can be utilized to accelerate the farthest-point queries. Computing the farthest point in parallel is, in principle, not a challenging problem (it could suitably be categorized as an "embarrassingly parallel" problem). However, the efficient utilization of various parallel hardware features and architectures is far from trivial. Efficient parallel implementations of the farthest-point routine were investigated already in Paper A, where multithreading and SIMD instructions on the CPU were used. In Paper C this investigation is continued by considering solutions based on the general-purpose GPU platform CUDA [105]. Using off-the-shelf routines from the Thrust library for CUDA [14], as well as handwritten code amenable to auto-tuning [38], we demonstrate speedups of up to 25×. An implementation of distance filtering on the GPU is also considered, but is not entirely successful.

Paper D

Linus Källberg, Evan Shellshear, and Thomas Larsson. *An External Memory Algorithm for the Minimum Enclosing Ball Problem.* The 11th International Conference on Computer Graphics Theory and Applications (GRAPP), 2016.

This paper studies the exact MEB problem in an external memory setting, where the input data resides in secondary storage, and only parts of it can fit in main memory at any given time. Such situations might arise under memory-constrained conditions, e.g., in mobile applications, or when very large data sets are involved, such as 3D-scanned point clouds. External memory, or out-of-core, algorithms are designed with the objective to both reduce the number of expensive transactions of data from secondary storage, and to improve the efficiency of these transactions by increasing the locality of the data accesses [136].

While crude approximations of the MEB can be computed under the streaming model, where each data element is allowed to be read only once, currently no exact algorithms tailored for external memory conditions seem to exist. We present a new algorithm primarily designed for low-dimensional problems, and compare it empirically to a naïve adaptation of Gärtner's algorithm [56] for external memory. By use of a variant of the distance filtering heuristic of Paper C, we achieve close to the optimal I/O of only one access per data element on large scanned point clouds.

Paper E

Linus Källberg and Thomas Larsson. *A Filtering Heuristic for the Computation of Minimum-Volume Enclosing Ellipsoids.* The 10th International Conference on Combinatorial Optimization and Applications (COCOA), 2016.

In this paper, a distance filtering heuristic for the MVEE problem is developed. Similar in concept to the techniques developed for the MEB problem in Paper C, the heuristic uses safe bounds to filter out unnecessary distance computations during the search for the farthest point in algorithms adhering to the strategy outlined in Algorithm 3.1. Whereas the computation on Line 3 of Algorithm 2.1 uses the Euclidean norm, however, the computation on Line 4 of Algorithm 3.1 uses the ellipsoidal norm induced by the current candidate ellipsoid. Thus, the triangle inequality cannot be used. Instead, specialized bounds based on the eigenvalues of the shape matrix of the current ellipsoid are used to formulate the filtering condition. Our computational results indicate that when implemented in the Todd–Yıldırım MVEE algorithm [130], the new filtering heuristic outperforms the elimination heuristic by Harman and Pronzato [60] for input data in 25 dimensions or more.

Paper F

Linus Källberg and Daniel Andrén. *Active Set Strategies for the Computation of Minimum-Volume Enclosing Ellipsoids.* Technical report, Mälardalen University (submitted), 2019.

In this work, we propose a new $(1 + \epsilon)$-approximation algorithm for the MVEE problem. The algorithm follows the general strategy of Algorithm 3.1, and can be viewed as an adaptation of the active set algorithm in Paper A to the MVEE problem. Here, the algorithm by Todd and Yıldırım [130] is invoked as a subroutine to compute a $(1+\epsilon)$-approximation of the MVEE of the current core-set in each iteration. However, in addition to the apporach of always adding only the farthest point to the core-set, as is done in the algorithm of Paper A and the pseudocode in Algorithm 3.1, alternative approaches are investigated in which more than one point outside of the current ellipsoid are selected. Several schemes for selecting these points are developed and compared, including an improved (according to our evaluation) variant of the scheme used in the active set strategy by Sun and Freund [118].

In addition, a new pruning heuristic is developed, which can be directly incorporated into any algorithm based on Algorithm 3.1. The heuristic is aggressive, in the sense that it is not based on a conservative condition that ensures that no support points of the MVEE are eliminated. Instead, all points outside of the current ellipsoid that are not part of the core-set are eliminated in each iteration. To ensure correctness, once $(1 + \epsilon)$-optimality has been reached for the set of remaining points, the original point set is restored and the optimality condition is evaluated again. If $(1 + \epsilon)$-optimality also holds over the full point set, then the solution is returned. If not, then this reveals that support points were indeed eliminated. In this case, the process is repeated with the restored point set and the current solution as the starting point.

Our computational evaluation on problems with 10^6 points in up to 25 dimensions indicates that performance can be improved by up to $70\times$ using the developed techniques.

Chapter 5

Conclusion

In this dissertation, new algorithms for large-scale instances of the MEB and MVEE problems are proposed. In addition, several speed-up techniques that address the main bottlenecks of these and existing algorithms are presented.

There is an abundance of applications in which the fast processing of large multidimensional data sets is essential. For example, in interactive computer graphics and simulation applications, such as virtual reality and video games, there is an ever-increasing demand for heightened realism through refined geometric detail and higher frame rates. Also in non-interactive applications, such as computer-generated imagery using advanced rendering techniques, efficient processing of the involved geometry is important. As was discussed in Sections 2.4 and 3.4, bounding spheres and ellipsoids can be used to accelerate various geometric operations in these applications, such as collision and visibility queries. As the geometric complexity increases, faster algorithms to compute the bounding shapes are necessary, whether the acceleration data structures are constructed as a preprocessing step, or they need to be recomputed dynamically in each time step of an interactive simulation.

More generally, massive amounts of data are generated each day by, e.g., mobile and sensing devices, and vast quantities of multimedia content is uploaded daily to the web. This unprecedented availability of data brings new opportunities to extract knowledge and to build highly accurate models for, e.g., novelty detection and pattern recognition. As discussed in Sections 2.4 and 3.4, MEBs and MVEEs provide powerful tools for such tasks, and the techniques proposed herein should be valuable additions in the effort toward meeting the demands imposed by the ever-growing data sets.

Although these techniques have mainly been presented and evaluated in the context of large data sets, they are likely beneficial also in contexts involving smaller data sets but more expensive operations on the input elements. For example, in the kernel methods discussed in Section 2.4.2, computing the distance between two points in the kernel-induced feature space is much more costly than computing their Euclidean distance as in the basic formulation of the MEB problem. Thus, skipping unnecessary such distance computations, by pruning or other means, should provide significant performance improvements also for less numerous data sets. Similar conditions hold for computing the MEB or MVEE under any expensive distance measure, or if the input contains other geometric primitives than points, as discussed in Sections 2.5 and 3.5, which require costly procedures to calculate the distances. Yet another example is when the numerical calculations need to be carried out using multiple-precision software libraries in order to ensure certain levels of numerical precision [57, 27].

5.1 Further Studies

To conclude this part of the thesis, we outline a few possible directions for further studies:

- In Paper B, the new bounds are used exclusively to accelerate the individual iterations, but they could likely also be used to improve the convergence rate of certain algorithms. For example, in the multiplicative algorithm by Lawson [83] (see (2.25) on page 28 of this thesis), pruning could be used to more quickly redistribute weight from irrelevant (pruned) points. Harman and Pronzato [60] demonstrate a similar use of their bound to speed up an algorithm for the D-optimal design problem by Titterington [127] (see (3.34) on page 53).

- Although the generic strategies of Algorithm 2.1 and 3.1, for which the techniques proposed in this thesis were designed, fits many efficient algorithms for the MEB and MVEE problems, it would be worthwhile to study pruning and/or filtering techniques also for algorithms based on different approaches. For example, in the primal algorithm by Fischer et al. [50], discussed in Section 2.3.1, the main computational hot spot consists of repeatedly scanning the whole point set to find the next point to enter the boundary of the shrinking candidate MEB. Specially designed speed up heuristics would likely be effective in reducing the number of points considered in each such scan.

- More generally, it would be relevant to study acceleration methods for the related problems discussed in Sections 2.5 and 3.5. Since several of these problems are amenable to first-order algorithms similar to those considered here, they could likely benefit from further developments and generalizations of the proposed heuristics.

- In the construction of MEB and MVEE hierarchies for various tasks such as collision detection and similarity searching, there appear to be opportunities to design speed up techniques that reuse information across different levels of the hierarchy when computing the enclosing shapes. For example, when computing the MEB or MVEE for a node, the union of the core-sets computed for the node's children should—even though it may not necessarily satisfy the core-set property—provide a good starting point for an active set strategy.

- Although promising results are reported from the empirical trials of the included papers, case studies applying the proposed methods in real-world computational tasks would be worthwhile, to assess their suitability in various applications. Such studies may identify certain characteristics of typical data sets and scenarios encountered in practice, which could guide the further refinement of our techniques.

Bibliography

[1] Pankaj K. Agarwal, Sariel Har-Peled, and Kasturi R. Varadarajan. Geometric approximation via coresets. *Combinatorial and Computational Geometry*, 52:1–30, 2005.

[2] S. Damla Ahipaşaoğlu, Peng Sun, and Michael J. Todd. Linear convergence of a modified Frank–Wolfe algorithm for computing minimum-volume enclosing ellipsoids. *Optimization Methods and Software*, 23(1):5–19, 2008.

[3] S. Damla Ahipaşaoğlu and Michael J. Todd. A modified Frank–Wolfe algorithm for computing minimum-area enclosing ellipsoidal cylinders: Theory and algorithms. *Computational Geometry*, 46(5):494–519, July 2013.

[4] S. Damla Ahipaşaoğlu and E. Alper Yıldırım. Identification and elimination of interior points for the minimum enclosing ball problem. *SIAM J. Optim.*, 19(3):1392–1396, 2008.

[5] Selin Damla Ahipaşaoğlu. Fast algorithms for the minimum volume estimator. *Journal of Global Optimization*, 62(2):351–370, August 2015.

[6] Dan Albocher and Gershon Elber. On the computation of the minimal ellipse enclosing a set of planar curves. In *2009 IEEE International Conference on Shape Modeling and Applications*. IEEE, June 2009.

[7] Ola Amayri and Nizar Bouguila. A study of spam filtering using support vector machines. *Artificial Intelligence Review*, 34(1):73–108, 2010.

[8] Ulf Assarsson and Tomas Möller. Optimized view frustum culling algorithms. Technical report, Department of Computer Engineering, Chalmers University of Technology, Gothenburg, Sweden, 1999.

[9] Corwin L. Atwood. Sequences converging to D-optimal designs of experiments. *The Annals of Statistics*, 1(2):342–352, 1973.

[10] Mihai Bădoiu and Kenneth L. Clarkson. Smaller core-sets for balls. In *Proceedings of the Fourteenth Annual ACM–SIAM Symposium on Discrete Algorithm*, pages 801–802, 2003.

[11] Mihai Bădoiu, Sariel Har-Peled, and Piotr Indyk. Approximate clustering via core-sets. In *Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing*, STOC '02, pages 250–257. ACM, 2002.

[12] E. R. Barnes. An algorithm for separating patterns by ellipsoids. *IBM Journal of Research and Development*, 26(6):759–764, November 1982.

[13] Felix Behrend. Über einige Affininvarianten konvexer Bereiche. *Mathematische Annalen*, 113(1):713–747, December 1937.

[14] Nathan Bell and Jared Hoberock. Thrust: A productivity-oriented library for CUDA. In *GPU Computing Gems Jade Edition*, pages 359–371. Elsevier, 2012.

[15] J. L. Bentley, H. T. Kung, M. Schkolnick, and C. D. Thompson. On the average number of maxima in a set of vectors and applications. *Journal of the ACM*, 25(4):536–543, October 1978.

[16] U. Betke and M. Henk. Approximating the volume of convex bodies. *Discrete & Computational Geometry*, 10(1):15–21, 1993.

[17] L. M. Blumenthal and G. E. Wahlin. On the spherical surface of smallest radius enclosing a bounded subset of n-dimensional Euclidean space. *Bulletin of the American Mathematical Society*, 47(10):771–777, 1941.

[18] D. Böhning. A vertex-exchange-method in D-optimal design theory. *Metrika*, 33(1):337–347, 1986.

[19] N. D. Botkin and V. L. Turova-Botkina. An algorithm for finding the Chebyshev center of a convex polyhedron. *Applied Mathematics & Optimization*, 29(2):211–222, March 1994.

[20] Christian Bouville. Bounding ellipsoids for ray-fractal intersection. In *Proceedings of SIGGRAPH '85*, pages 45–52. ACM, 1985.

[21] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[22] Yaroslav Bulatov, Sachin Jambawalikar, Piyush Kumar, and Saurabh Sethia. Hand recognition using geometric classifiers. In *Biometric Authentication*, volume 3072 of *Lecture Notes in Computer Science*, pages 753–759. Springer Berlin Heidelberg, 2004.

[23] Gabriele Capannini and Thomas Larsson. Adaptive collision culling for massive simulations by a parallel and context-aware sweep and prune algorithm. *IEEE Transactions on Visualization and Computer Graphics*, 24(7):2064–2077, 2017.

[24] C. Carathéodory. Über den Variabilitätsbereich der Koeffizienten von Potenzreihen, die gegebene Werte nicht annehmen. *Mathematische Annalen*, 64(1):95–115, March 1907.

[25] Jorge Caravantes and Laureano Gonzalez-Vega. On the interference problem for ellipsoids: Experiments and applications. In *Mathematical Software – ICMS 2018*, pages 89–97. Springer International Publishing, 2018.

[26] Lawrence Cayton. Fast nearest neighbor retrieval for Bregman divergences. In *Proceedings of the 25th international conference on Machine learning – ICML '08.* ACM Press, 2008.

[27] CGAL, Computational Geometry Algorithms Library. `https://www.cgal.org`. Accessed: 2019-08-16.

[28] Timothy M. Chan. Improved deterministic algorithms for linear programming in low dimensions. *ACM Transactions on Algorithms*, 14(3):1–10, June 2018.

[29] Timothy M. Chan and Vinayak Pathak. Streaming and dynamic algorithms for minimum enclosing balls in high dimensions. *Computational Geometry*, 47(2):240–247, February 2014.

[30] Jung-Woo Chang, Wenping Wang, and Myung-Soo Kim. Efficient collision detection using a dual OBB-sphere bounding volume hierarchy. *Computer-Aided Design*, 42(1):50–57, 2010.

[31] Bernard Chazelle and Jiří Matoušek. On linear-time deterministic algorithms for optimization problems in fixed dimension. *J. Algorithms*, 21(3):579–597, 1996.

[32] George Chrystal. On the problem to construct the minimum circle enclosing n given points in the plane. *Proceedings of the Edinburgh Mathematical Society*, 3:30–33, 1885.

[33] Kenneth L. Clarkson. Linear programming in $O(n \times 3^{d^2})$ time. *Information Processing Letters*, 22(1):21–24, 1986.

[34] Kenneth L. Clarkson. Las Vegas algorithms for linear and integer programming when the dimension is small. *Journal of the ACM (JACM)*, 42(2):488–499, 1995.

[35] Kenneth L. Clarkson. Coresets, sparse greedy approximation, and the Frank–Wolfe algorithm. In *Proceedings of the Nineteenth Annual ACM–SIAM Symposium on Discrete Algorithms*, pages 922–931. Society for Industrial and Applied Mathematics, 2008.

[36] Wei-jie Cong, Hong-wei Liu, Feng Ye, and Shui-sheng Zhou. Rank-two update algorithms for the minimum volume enclosing ellipsoid problem. *Computational Optimization and Applications*, 51(1):241–257, 2012.

[37] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, September 1995.

[38] Andrew Davidson and John Owens. Toward techniques for auto-tuning GPU algorithms. In *Applied Parallel and Scientific Computing*, pages 110–119. Springer Berlin Heidelberg, 2012.

[39] P. M. Dearing and Christiane R. Zeck. A dual algorithm for the minimum covering ball problem in $\mathbb{R}^n$. *Operations Research Letters*, 37(3):171–175, 2009.

[40] P.M. Dearing, Pietro Belotti, and Andrea M. Smith. A primal algorithm for the weighted minimum covering ball problem in $\mathbb{R}^n$. *TOP*, 24(2):466–492, 2016.

[41] P.M. Dearing and Andrea M. Smith. A dual algorithm for the minimum covering weighted ball problem in $\mathbb{R}^n$. *Journal of Global Optimization*, 55(2):261–278, 2013.

[42] Martin Dyer. A class of convex programs with applications to computational geometry. In *Proceedings of the eighth annual symposium on Computational geometry – SCG '92*. ACM Press, 1992.

[43] Martin E. Dyer. Linear time algorithms for two- and three-variable linear programs. *SIAM Journal on Computing*, 13(1):31–45, 1984.

[44] Martin E. Dyer. On a multidimensional search technique and its application to the Euclidean one-centre problem. *SIAM Journal on Computing*, 15(3):725–738, 1986.

[45] D. Jack Elzinga and Donald W. Hearn. Geometrical solutions for some minimax location problems. *Transportation Science*, 6(4):379–394, nov 1972.

[46] D. Jack Elzinga and Donald W. Hearn. The minimum covering sphere problem. *Management Science*, 19(1):96–104, 1972.

[47] Christer Ericson. *Real-Time Collision Detection.* Morgan Kaufmann, 2005.

[48] Valerii Vadimovich Fedorov. *Theory of Optimal Experiments.* Elsevier, 1972.

[49] Kaspar Fischer and Bernd Gärtner. The smallest enclosing ball of balls: Combinatorial structure and algorithms. *International Journal of Computational Geometry & Applications*, 14(04n05):341–378, October 2004.

[50] Kaspar Fischer, Bernd Gärtner, and Martin Kutz. Fast smallest-enclosing-ball computation in high dimensions. In *Algorithms – ESA '03*, volume 2832 of *Lecture Notes in Computer Science*, pages 630–641. Springer Berlin Heidelberg, 2003.

[51] Philip J. Fleming and John J. Wallace. How not to lie with statistics: The correct way to summarize benchmark results. *Communications of the ACM*, 29(3):218–221, March 1986.

[52] Richard L. Francis. Letter to the editor—Some aspects of a minimax location problem. *Operations Research*, 15(6):1163–1169, December 1967.

[53] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 1956.

[54] Taras Galkovskyi, Bernd Gärtner, and Bogdan Rublev. The domination heuristic for LP-type problems. In *2009 Proceedings of the Eleventh Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 74–84, 2009.

[55] Bernd Gärtner. A subexponential algorithm for abstract optimization problems. *SIAM Journal on Computing*, 24(5):1018–1035, October 1995.

[56] Bernd Gärtner. Fast and robust smallest enclosing balls. In *Algorithms – ESA '99*, volume 1643 of *Lecture Notes in Computer Science*, pages 325–338. Springer Berlin Heidelberg, 1999.

[57] Bernd Gärtner and Sven Schönherr. Exact primitives for smallest enclosing ellipses. *Information Processing Letters*, 68(1):33–38, October 1998.

[58] Stefan Gottschalk, Ming C Lin, and Dinesh Manocha. OBBTree: A hierarchical structure for rapid interference detection. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pages 171–180. ACM, 1996.

[59] Radoslav Harman, Lenka Filová, and Peter Richtárik. A randomized exchange algorithm for computing optimal approximate designs of experiments. *Journal of the American Statistical Association*, pages 1–30, December 2019.

[60] Radoslav Harman and Luc Pronzato. Improvements on removing nonoptimal support points in D-optimum design algorithms. *Statistics & Probability Letters*, 77(1):90–94, 2007.

[61] Donald Hearn. Observations on the minimum sphere problem. Technical Report RR-77-5, Florida University Gainesville Department of Industrial And Systems Engineering, 1977.

[62] Donald W. Hearn and James Vijay. Efficient algorithms for the (weighted) minimum circle problem. *Operations Research*, 30(4):777–795, 1982.

[63] Martin Henk. Löwner–John ellipsoids. *Documenta Math. Extra Volume: Optimization Stories*, pages 95–106, 2012.

[64] Theodore H. Hopp and Charles P. Reeve. An algorithm for computing the minimum covering sphere in any dimension. Technical report, US Department of Commerce, National Institute of Standards and Technology, 1996.

[65] Philip M Hubbard. Approximating polyhedra with spheres for time-critical collision detection. *ACM Transactions on Graphics (TOG)*, 15(3):179–210, 1996.

[66] Sachin Jambawalikar and Piyush Kumar. A note on approximate minimum volume enclosing ellipsoid of ellipsoids. In *2008 International Conference on Computational Sciences and Its Applications*. IEEE, June 2008.

[67] Doug L. James and Dinesh K. Pai. BD-tree: Output-sensitive collision detection for reduced deformable models. *ACM Transactions on Graphics*, 23(3):393–398, 2004.

[68] Wei jie Cong and Hong wei Liu. Modified algorithms for the minimum volume enclosing axis-aligned ellipsoid problem. *Discrete Applied Mathematics*, 158(6):627–635, March 2010.

[69] Fritz John. Extremum problems with inequalities as subsidiary conditions. In *Studies and Essays, Presented to R. Courant on His 60th Birthday*, pages 187–204. Wiley Interscience, 1984.

[70] Linus Källberg and Thomas Larsson. Ray tracing using hierarchies of slab cut balls. In *Eurographics conference 2010 – short papers*, pages 69–72. Eurographics Association, 2010.

[71] Leonid G. Khachiyan. Rounding of polytopes in the real number model of computation. *Mathematics of Operations Research*, 21(2):307–320, 1996.

[72] Leonid G. Khachiyan and Michael J. Todd. On the complexity of approximating the maximal inscribed ellipsoid for a polytope. *Mathematical Programming*, 61(1-3):137–159, August 1993.

[73] J. Kiefer and J. Wolfowitz. The equivalence of two extremum problems. *Canadian Journal of Mathematics*, 12:363–366, 1960.

[74] J.T. Klosowski, M. Held, J.S.B. Mitchell, H. Sowizral, and K. Zikan. Efficient collision detection using bounding volume hierarchies of k-DOPs. *IEEE Transactions on Visualization and Computer Graphics*, 4(1):21–36, 1998.

[75] P. Kumar and E. A. Yıldırım. Minimum-volume enclosing ellipsoids and core sets. *Journal of Optimization Theory and Applications*, 126(1):1–21, 2005.

[76] P. Kumar and E. A. Yıldırım. Computing minimum-volume enclosing axis-aligned ellipsoids. *Journal of Optimization Theory and Applications*, 136(2):211–228, November 2008.

[77] Piyush Kumar, Joseph S. B. Mitchell, and E. Alper Yıldırım. Approximate minimum enclosing balls in high dimensions using core-sets. *Journal of Experimental Algorithmics*, 8, 2003.

[78] Piyush Kumar and Emre Alper Yıldı rım. An algorithm and a core set result for the weighted Euclidean one-center problem. *INFORMS Journal on Computing*, 21(4):614–629, 2009.

[79] Ruth Kurniawati, Jesse S. Jin, and John A. Shepard. SS^{+}-tree: An improved index structure for similarity searches in a high-dimensional feature space. In Ishwar K. Sethi and Ramesh C. Jain, editors, *Storage and Retrieval for Image and Video Databases V*. SPIE, January 1997.

[80] Thomas Larsson. An efficient ellipsoid-OBB intersection test. *Journal of Graphics Tools*, 13(1):31–43, January 2008.

[81] Thomas Larsson and Tomas Akenine-Möller. Bounding volume hierarchies of slab cut balls. *Computer Graphics Forum*, 28(8):2379–2395, 2009.

[82] Patrick Laube, Marc van Kreveld, and Stephan Imfeld. Finding REMO – Detecting relative motion patterns in geospatial lifelines. In *Developments in Spatial Data Handling*, pages 201–215. Springer Berlin Heidelberg, 2005.

[83] C. L. Lawson. The smallest covering cone or sphere (C. Groenewod and L. Eusanio). *SIAM Review*, 7(3):415–2, 07 1965.

[84] Shengjun Liu, Charlie C. L. Wang, Kin-Chuen Hui, Xiaogang Jin, and Hanli Zhao. Ellipsoid-tree construction for solid objects. In *Proceedings of the 2007 ACM Symposium on Solid and Physical Modeling*, pages 303–308. ACM, 2007.

[85] Ying Liu, Dengsheng Zhang, Guojun Lu, and Wei-Ying Ma. A survey of content-based image retrieval with high-level semantics. *Pattern Recognition*, 40(1):262–282, January 2007.

[86] S. Mandal and B. Torsney. Construction of optimal designs using a clustering approach. *Journal of Statistical Planning and Inference*, 136(3):1120–1134, March 2006.

[87] J. Matoušek, M. Sharir, and E. Welzl. A subexponential bound for linear programming. *Algorithmica*, 16(4-5):498–516, October 1996.

[88] Nimrod Megiddo. Linear-time algorithms for linear programming in $\mathbb{R}^3$ and related problems. *SIAM J. Comput.*, 12(4):759–776, 1983.

[89] Nimrod Megiddo. Linear programming in linear time when the dimension is fixed. *Journal of the ACM (JACM)*, 31(1):114–127, 1984.

[90] Nimrod Megiddo. On the ball spanned by balls. *Discrete & Computational Geometry*, 4(6):605–610, 1989.

[91] Janaina Mourão-Miranda, David R. Hardoon, Tim Hahn, Andre F. Marquand, Steve C.R. Williams, John Shawe-Taylor, and Michael Brammer. Patient classification as an outlier detection problem: An application of the one-class support vector machine. *NeuroImage*, 58(3):793–804, 2011.

[92] Ramanathan Muthuganapathy, Gershon Elber, Gill Barequet, and Myung-Soo Kim. Computing the minimum enclosing sphere of free-form hypersurfaces in arbitrary dimensions. *Computer-Aided Design*, 43(3):247–257, March 2011.

[93] Yurii Nesterov and Arkadii Nemirovskii. *Interior-point polynomial algorithms in convex programming*, volume 13. Siam, 1994.

[94] Frank Nielsen and Richard Nock. Approximating smallest enclosing balls with applications to machine learning. *Int. J. Comput. Geom. Appl.*, 19(5):389–414, 2009.

[95] Richard Nock and Frank Nielsen. Fitting the smallest enclosing Bregman ball. In *Machine Learning: ECML 2005*, pages 649–656. Springer Berlin Heidelberg, 2005.

[96] Rina Panigrahy. Minimum enclosing polytope in high dimensions. *arXiv:cs/0407020*, 2004.

[97] Mark J. Post. Minimum spanning ellipsoids. In *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*, STOC '84, pages 108–116. ACM, 1984.

[98] Luc Pronzato. On the elimination of inessential points in the smallest enclosing ball problem. *Optimization Methods and Software*, pages 1–23, 2017.

[99] S. Quinlan. Efficient distance computation between non-convex objects. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*. IEEE Comput. Soc. Press, 1994.

[100] Anant Raj, Jakob Olbrich, Bernd Gärtner, Bernhard Schölkopf, and Martin Jaggi. Screening rules for convex problems. *arXiv preprint arXiv:1609.07478*, 2016.

[101] Elon Rimon and Stephen P. Boyd. Obstacle collision detection using best ellipsoid fit. *Journal of Intelligent and Robotic Systems*, 18(2):105–126, 1997.

[102] Jack Ritter. An efficient bounding sphere. In Andrew S. Glassner, editor, *Graphics Gems*, pages 301–303. Academic Press Professional, Inc., 1990.

[103] Judah Ben Rosen. Pattern separation by convex programming. *Journal of Mathematical Analysis and Applications*, 10(1):123–134, 1965.

[104] Diego C Ruspini, Krasimir Kolarov, and Oussama Khatib. The haptic display of complex graphical environments. In *Proceedings of SIGGRAPH '97*, volume 97, pages 345–352, 1997.

[105] Jason Sanders and Edward Kandrot. *CUDA by example: An introduction to general-purpose GPU programming.* Addison-Wesley Professional, 2010.

[106] Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

[107] Ulrich Schwesinger, Roland Siegwart, and Paul Furgale. Fast collision detection through bounding volume hierarchies in workspace-time space for sampling-based motion planners. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 63–68. IEEE, 2015.

[108] Raimund Seidel. Small-dimensional linear programming and convex hulls made easy. *Discrete & Computational Geometry*, 6(3):423–434, September 1991.

[109] Michael Ian Shamos and Dan Hoey. Closest-point problems. In *16th Annual Symposium on Foundations of Computer Science (SFCS 1975)*, pages 151–162. IEEE, 1975.

[110] Micha Sharir and Emo Welzl. A combinatorial bound for linear programming and related problems. In *STACS 92*, pages 567–579. Springer Berlin Heidelberg, 1992.

[111] Romy Shioda and Levent Tunçel. Clustering via minimum volume ellipsoids. *Computational Optimization and Applications*, 37(3):247–295, April 2007.

[112] Robin Sibson. Discussion of Dr Wynn's and of Dr Laycock's papers. *Journal of the Royal Statistical Society: Series B*, 34(2):181–183, jan 1972.

[113] B. W. Silverman and D. M. Titterington. Minimum covering ellipses. *SIAM Journal on Scientific and Statistical Computing*, 1(4):401–409, December 1980.

[114] S. D. Silvey. Discussion of Dr Wynn's and of Dr Laycock's papers. *Journal of the Royal Statistical Society: Series B*, 34(2):174–175, jan 1972.

[115] S. D. Silvey and D. M. Titterington. A geometric approach to optimal design theory. *Biometrika*, 60(1):21–32, 1973.

[116] S.D. Silvey, D.H. Titterington, and B. Torsney. An algorithm for optimal designs on a design space. *Communications in Statistics – Theory and Methods*, 7(14):1379–1389, January 1978.

[117] J. E. Smith. Characterizing computer performance with a single number. *Communications of the ACM*, 31(10):1202–1206, October 1988.

[118] Peng Sun and Robert M. Freund. Computation of minimum-volume covering ellipsoids. *Operations Research*, 52(5):690–706, 2004.

[119] J. J. Sylvester. A question in the geometry of situation. *Quarterly Journal of Pure and Applied Mathematics*, 1, 1857.

[120] J. J. Sylvester. On Poncelet's approximate linear valuation of surd forms. *Philosophical Magazine*, 20(132):203–222, 1860.

[121] D. Tax, A. Ypma, and R. Duin. Support vector data description applied to machine vibration analysis. In *Proceedings of the 5th Annual Conference of the Advanced School for Computing and Imaging*, pages 398–405, 1999.

[122] David M.J. Tax and Robert P.W. Duin. Support vector data description. *Machine Learning*, 54(1):45–66, 2004.

[123] Kasemsit Teeyapan, Nipon Theera-Umpon, and Sansanee Auephanwiriyakul. Ellipsoidal support vector data description. *Neural Computing and Applications*, 28(S1):337–347, May 2017.

[124] Kasemsit Teeyapan, Nipon Theera-Umpon, and Sansanee Auephanwiriyakul. A twin-hyperellipsoidal support vector classifier. *Journal of Intelligent & Fuzzy Systems*, 35(6):6141–6152, December 2018.

[125] Bo Tian. Bouncing bubble: A fast algorithm for minimal enclosing ball problem. Self-published through GRIN Publishing, 2012. www.grin.com.

[126] D. M. Titterington. Optimal design: Some geometrical aspects of D-optimality. *Biometrika*, 62(2):313–320, 1975.

[127] D. M. Titterington. Algorithms for computing D-optimal designs on a finite design space. In *Proc. of the 1976 Conf. on Information Science and Systems, John Hopkins University*, volume 3, pages 213–216, 1976.

[128] D. M. Titterington. Estimation of correlation coefficients by ellipsoidal trimming. *Applied Statistics*, 27(3):227–234, 1978.

[129] Michael J. Todd. *Minimum-Volume Ellipsoids: Theory and Algorithms.* Society for Industrial and Applied Mathematics, 2016.

[130] Michael J. Todd and E. Alper Yıldırım. On Khachiyan's algorithm for the computation of minimum-volume enclosing ellipsoids. *Discrete Applied Mathematics*, 155(13):1731–1744, 2007.

[131] B. Torsney and R. Martín-Martín. Multiplicative algorithms for computing optimum designs. *Journal of Statistical Planning and Inference*, 139(12):3947–3961, December 2009.

[132] Ivor Wai Hung Tsang, James T. Kwok, and Pak-Ming Cheung. Core vector machines: Fast SVM training on very large data sets. *Journal of Machine Learning Research*, 6(Apr):363–392, 2005.

[133] Ivor Wai Hung Tsang, James Tin Yau Kwok, and Jacek M. Zurada. Generalized core vector machines. *IEEE Transactions on Neural Networks*, 17(5):1126–1140, 2006.

[134] Charles F. Van Loan and Gene H. Golub. *Matrix Computations.* Johns Hopkins University Press, fourth edition edition, 2013.

[135] Lieven Vandenberghe, Stephen Boyd, and Shao-Po Wu. Determinant maximization with linear matrix inequality constraints. *SIAM Journal on Matrix Analysis and Applications*, 19(2):499–533, April 1998.

[136] Jeffrey Scott Vitter. Algorithms and data structures for external memory. *Foundations and Trends in Theoretical Computer Science*, 2(4):305–474, 2008.

[137] Ingo Wald, Solomon Boulos, and Peter Shirley. Ray tracing deformable scenes using dynamic bounding volume hierarchies. *ACM Transactions on Graphics*, 26(1), 2007.

[138] G. Wang, G. Cao, and T.F. La Porta. Movement-assisted sensor deployment. *IEEE Transactions on Mobile Computing*, 5(6):640–652, June 2006.

[139] Emo Welzl. Smallest enclosing disks (balls and ellipsoids). In *New Results and New Trends in Computer Science*, volume 555 of *Lecture Notes in Computer Science*, pages 359–370. Springer Berlin Heidelberg, 1991.

[140] Emo Welzl. The smallest enclosing circle – A contribution to democracy from switzerland? In *Algorithms Unplugged*, pages 357–360. Springer Berlin Heidelberg, 2011.

[141] Philip Wolfe. Convergence theory in nonlinear programming. *Integer and Nonlinear Programming*, pages 1–36, 1970.

[142] Henry P. Wynn. The sequential generation of D-optimum experimental designs. *Ann. Math. Statist.*, 41(5):1655–1664, 10 1970.

[143] E. Alper Yıldırım. On the minimum volume covering ellipsoid of ellipsoids. *SIAM Journal on Optimization*, 17(3):621–641, January 2006.

[144] E. Alper Yıldırım. Two algorithms for the minimum enclosing ball problem. *SIAM J. Optim.*, 19(3):1368–1391, 2008.

[145] Yaming Yu. D-optimal designs via a cocktail algorithm. *Statistics and Computing*, 21(4):475–481, June 2011.

[146] Hamid Zarrabi-Zadeh and Timothy M. Chan. A simple streaming algorithm for minimum enclosing balls. In *Proceedings of the 18th Annual Canadian Conference on Computational Geometry, CCCG 2006*, pages 139–142, 2006.

[147] Yin Zhang. An interior-point algorithm for the maximum-volume ellipsoid problem. Technical Report TR98-15, Department of Computational and Applied Mathematics, Rice University, Houston, TX, 1998.

[148] Yin Zhang and Liyan Gao. On numerical solution of the maximum volume ellipsoid problem. *SIAM Journal on Optimization*, 14(1):53–76, January 2003.