2012

Geometric Fitting of Quadratic Curves and Surfaces

Hui Ma
University of Alabama at Birmingham

GEOMETRIC FITTING OF QUADRATIC CURVES AND SURFACES

by

HUI MA

NIKOLAI CHERNOV, COMMITTEE CHAIR
WEI-SHEN HSIA
CHARLES KATHOLI
IAN KNOWLES
BORIS KUNIN

A DISSERTATION

Submitted to the graduate faculty of The University of Alabama,
The University of Alabama at Birmingham, and The University of Alabama in
Huntsville in partial fulfillment of the requirements for the degree
of Doctor of Philosophy

BIRMINGHAM, ALABAMA

2011

ABSTRACT

GEOMETRIC FITTING OF QUADRATIC CURVES AND SURFACES

HUI MA

APPLIED MATHEMATICS

Fitting quadratic curves and surfaces to observed images is one of the basic tasks in pattern recognition and computer vision. The most accurate and robust fit is obtained by minimizing geometric (orthogonal) distances, but this problem has no closed form solution. All known algorithms are heuristic and either computationally costly or have drawbacks in accuracy and convergence. We develop a mathematically rigorous approach to the study of the geometric fitting problem. We begin with a thorough investigation of relevant theoretical aspects of the problem and then move on to its practical solution.

We focus on image processing applications, where data points come from a picture, photograph, map, etc. Therefore we adopt standard statistical assumptions that are appropriate for these applications. We investigate the existence of the best fit, describe various parameterization schemes and analyze the behavior of the objective function on the parameter space.

Our goal is to provide a robust, efficient projection method and to develop new fitting schemes, indicating how to combine them to achieve the best performance. Eberly discovered a remarkably fast and totally reliable projection algorithm for ellipses which we generalize to all the other quadratic curves and surfaces and provide proofs of convergence. Ahn has classified various approaches to the fitting problem.

We develop our implicit fitting algorithm based on one of his approaches and demonstrate that it is the most efficient one by comparison to other known algorithms. By combining projection and minimization steps together, we give a complete, reliable and efficient geometric fitting scheme for fitting quadratic curves and surfaces of all kinds.

Keywords: geometric fitting, image processing, quadratic curves, surfaces, projection, minimization.

DEDICATION

TO MY BELOVED PARENTS

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to my advisor Nikolai Chernov. Without his guidance and continuous support throughout the process, I will never successfully finish my dissertation. Also I would like to express a deep gratitude to all my committee members, W. Hsia, C. Katholi, I. Knowles, B. Kunin for their invaluable guidance and remarks.

Many thanks to all faculty, staff and fellow students within the Math department of UAB for their assistance and continuous encouragement, in particular, R.Weikard, Y. Karpeshina, M. Nkashama, J. Mayor, L. Stansell, S. Abdoli and L. Cheryl.

Last but not least, my sincere thanks to my parents Shenghe and Lanfeng. Thank you for your love and always being there to support me and to encourage me to succeed.

Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

The purpose of my research is a general study of an active topic in modern statistics: fitting quadratic curves and surfaces to observed data, in particular, to digitized images. A curve in the xy plane can be described by equation $y = f(x; \Theta)$, where Θ denotes parameters of the curve. Observed data (or a digitized image) is represented by a finite set of points $(x_1, y_1), (x_2, y_2) \ldots, (x_n, y_n)$. Our task is to find the best fitting curve $y = f(x; \Theta)$, i.e., estimate Θ so that the curve passes as close to the given points as possible.

In many applications both coordinates x and y of the observed points are measured imprecisely, i.e., both variables x and y are subject to random errors. The corresponding topic in statistics is known as Errors-In-Variables (EIV) regression. It is quite different and much more complex than the classical regression where the independent variable is assumed to be deterministic (error-free).

My dissertation is focused on image processing applications, where data points come from a picture, photograph, map, etc. In this case both x and y variables are measured in the same units; their errors are independent and have the same magnitude, on average. Thus one assumes that errors Δx and Δy in each point are i.i.d normal random variables with a common variance. Under this assumption, the Maximum Likelihood Estimation (MLE) gives the curve $y = f(x; \Theta)$ that minimizes the sum of squares of the distances to the given points [**16, 19, 18**].

The most common geometric features in applications are lines, circles, ellipses, etc. More complex curved shapes can be approximated by a sequence of arcs "stitched together" (splines); see [**11, 43, 46, 49, 50**]. Fitting straight lines to observed data

when both variables are subject to random errors is an old problem dating back to the 1870s [**2, 3, 36**], and all the major issues were resolved by the late 1990s.

The problem of fitting circles and circular arcs to observed data points started in the 1950s [**12, 28, 57, 58, 59**]. Since about 1980s, it has become an agenda in many application areas. For example, in industry, quality control requires estimation of the radius and the center of the manufactured mechanical parts [**37**]; In medicine, doctors design dental arches from an X-ray [**13**]; In archeology, people determine the size of ancient pottery by analyzing potsherds found in the field expeditions [**22, 30, 31, 60**].

The problem of fitting ellipses and other quadratic curves (conic sections) was first mentioned in the 1970's [**42, 13, 10**]. It became popular and attracted attention of computer vision community in the 1990s [**29, 52, 53, 25**]. As a matter of fact, almost all problems in computer vision, image processing or pattern recognition are related in one form or another to the problem of fitting geometric curves or surfaces to the noisy data. Rapid development of new technologies allows us to obtain large amount of data in short time, thus opening new challenges for us to look for good fitting approaches and to process all the information accurately and efficiently.

The goal of my dissertation is to present the topic of fitting ellipses and general quadratic curves to observed data points in geometrical and computational aspects. All methods and investigations are carried over to 3D space for the problem of fitting quadratic surfaces.

1.1. Scope of the Thesis

Geometric Fitting. What is geometric fitting? Fitting problems have been usually solved through the least squares method. Least Squares fitting minimizes the sum of the squared distances from the given points to the fitted geometric feature. [**47, 45, 44**] provide a good overview of the various distance definitions. In geometric fitting, also known as *best fitting*, the distances are defined as the orthogonal or geometric

distances:

$$\sum_{i=1}^{n} {d_i}^2 = \sum_{i=1}^{n} (x_i - x_i')^2 + (y_i - y_i')^2. \tag{1.1}$$

Here d_i denotes the geometric distance from the observed point (x_i, y_i) to the fitted geometric feature, and (x_i', y_i') is the (orthogonal) projection of (x_i, y_i) onto the feature. Geometric fit has many nice characteristics:

- It is invariant under translations, rotations, and scaling, i.e., the fitted geometric feature does not depend on the choice of the coordinate system.
- It provides the maximum likelihood estimate of the parameters of the fitted feature under standard statistical assumptions.
- Geometric fitting has been prescribed by a recently ratified standard for testing the data processing software for coordinate metrology [**1**].

Objective Function. Geometric fitting is based on minimizing the sum of squares of the geometric distances from the given points to the fitted feature. Given n points $P_1, \ldots, P_n$ and a model object S, the objective function is defined by

$$\mathcal{F}(S) = \sum_{i=1}^{n} \big[\operatorname{dist}(P_i, S)\big]^2. \tag{1.2}$$

The given points are fixed (we cannot change them), so they are not listed as arguments [1] of $\mathcal{F}$.

The given collection of model objects will be denoted by $\mathbb{M}$. We put no restrictions on the collection $\mathbb{M}$ of model objects, other than all $S \in \mathbb{M}$ are assumed to be closed sets. The reason for this requirement will be explained in section 2.1.2. Our goal is to find the best fit $S_{\text{best}} \in \mathbb{M}$ on which the function $\mathcal{F}$ takes its minimum value, i.e.,

$$S_{\text{best}} = \arg\min_{S \in \mathbb{M}} \mathcal{F}(S). \tag{1.3}$$

[1] This is similar to classical statistics: the arguments of a likelihood function are only model parameters, but not observed values of the random variable.

Redundancy Principle. There is a possible redundancy of model objects in the collection $\mathbb{M}$. If an object $S' \in \mathbb{M}$ is a subset of another object $S \in \mathbb{M}$, i.e., $S' \subset S$, then for any point P we have

$$\operatorname{dist}(P, S) \leq \operatorname{dist}(P, S'), \tag{1.4}$$

Here $\operatorname{dist}(P, S)$ denotes the distance from the given point P to the object S. Thus S' cannot fit any set of data points better than S does. So for the purpose of minimizing $\mathcal{F}$, i.e., finding the best fitting object, we may ignore all model objects that are proper subsets of other model objects. This may reduce the collection $\mathbb{M}$ somewhat. Such a reduction is not necessary, it is just a matter of convenience.

Conversely, there is no harm in considering any subset $S' \subset S$ of an object $S \in \mathbb{M}$ as a (smaller) object, too. Indeed, if S' provides a best fit (i.e., minimizes the objective function $\mathcal{F}$), then so does S, because $\mathcal{F}(S) \leq \mathcal{F}(S')$. Hence including S' into the collection $\mathbb{M}$ will not really be an extension of $\mathbb{M}$, its inclusion will not change the best fit.

1.2. Aims and Objectives

The minimization of the squared distances is a nonlinear problem that has no closed form solution. There is no direct algorithm for computing the minimum of $\mathcal{F}$. Various iterative algorithms have been applied to this end. The most popular ones are the Levenberg-Marquardt method and the Trust Region method. Levenberg-Marquardt method is a short name for the classical Gauss-Newton method with Levenberg-Marquardt correction [**38, 39**]. It can effectively solve any least squares problem provided the first derivative of d_i's with respect to the parameters can be computed. Trust Region method is a modification to Levenberg-Marquardt method which was developed in the 1970s [**40**]. Both algorithms are quite stable and reliable, and they usually converge rapidly. For their main ideas, see Appendix B.

The implementation of the geometric fitting of ellipses and other quadratic curves is much more difficult than that of circle fitting. Not only we have more independent

parameters than in the case of circles, but many other geometric and algebraic aspects of the problem lead to complications. The objective function is defined as

$$\mathcal{F}(\Theta) = \sum_{i=1}^{n} d_i(\Theta)^2,$$

where Θ is the vector of unknown parameters of the fitted quadratic curve; d_i's are the geometric distances between the given points and the fitted quadratic curve and they depend on the parameter Θ. To minimize such objective function, the Levenberg-Marquardt method or the Trust Region method requires the computation of the geometric distances d_i's, as well as their derivatives with respect to Θ. In the case of circles, there is an explicit formula for the distances

$$d_i = \sqrt{(x_i - a)^2 + (y_i - b)^2} - R$$

where (x_i, y_i) are given points, (a, b) is the center of the circle and R is its radius. Hence one can easily compute the objective function and also the derivative of d_i with respect to a, b, R. This makes the minimization of $\mathcal{F}$ rather straightforward.

When we consider ellipses or other quadratic curves such as hyperbolas or parabolas, the bad news is that there is no explicit analytic formula for computing the orthogonal distances, which could be used in the Levenberg-Marquardt procedure. Theoretically, the distances can be found by solving a polynomial equation of degree four [**64**], but such a solution is complicated and inconvenient, and it is numerically unstable [**7**]. Alternatively, one can compute the distance from a point to the curve by an iterative procedure, such as Newton's method [**7**], but this is quite expensive and still not absolutely reliable. Hence the very first problem one has to deal with is to find a reliable and efficient method for computing the orthogonal or the shortest distances from the given points to the quadratic curves.

Furthermore, there seem to be no closed form expression for the partial derivatives of the distance d_i with respect to the parameters Θ. It is not so clear how to differentiate d_i. Since one cannot easily compute the objective function or its derivatives, it appears that the Levenberg-Marquardt scheme is impractical (see [**55**] that

mentioned such a point of view). Thus one has to seek other ways or derive formulas for finding those derivatives in order to apply minimization algorithms.

To overcome these difficulties, some researchers enlarge the parameter space. In addition to the parameters defining the quadratic curve itself, they introduce n new parameters defining the position of the projection points (the closest points on the quadratic curve to the given points). This was done by Gander, Golub, and Strebel [**29**] for the ellipse fitting problem. Their procedure avoids the calculation of d_i's, but the minimization in the $(n+5)$-dimensional parameter space is predictably cumbersome and slow. So the authors of [**29**] concluded that the minimization of geometric distances for ellipses was a prohibitively difficult task. Some other known algorithms are also computationally costly or have drawbacks in accuracy and convergence [**52, 53, 51**]. Therefore we arrive at the necessity of developing a better, complete and reliable fitting scheme for fitting quadratic curves (surfaces) of all kinds.

Lastly, before moving on to the computational and practical aspects of the problem, it is also necessary to address some fundamental issues which are essential for understanding of advantages and disadvantages of practical algorithms. We will investigate the existence of the best fit, the right choice of parameters to work with, basic properties of the parameter space and the general behavior of the objective function on the parameter space.

1.3. Organization of the Thesis

The dissertation is organized as follows. It consists of three parts. Chapter 2, 3 and 4 make the first part, which deals with theoretical issues and aspects of fitting quadratic curves. Chapter 2 gives some theoretical analysis and probabilistic approach of the existence of the best fit. Chapter 3 describes various parametrization schemes for quadratic curves and their comparisons. Chapter 3 analyzes the properties of the parameter space for quadratic curves and the behavior of the objective function on the space. Open domains which occupy the whole parameter space and

their volumes will be described. Continuity, differentiability and local minima of the objective function will be discussed and illustrated. Part II which consists of Chapter 5 and 6 is devoted to practical aspects of fitting quadratic curves and surfaces. In Chapter 5, we evaluate several projection algorithms, describe the most efficient method for fitting ellipse and then adapt it to other quadratic curves. In each case, we provide a theoretical proof of convergence. Chapter 6 presents practical solutions to the problem of fitting quadratic curves including ellipse, hyperbola and parabola. They were also compared with other known algorithms to demonstrate a superior performance. Part III is devoted to the problem of fitting quadratic *surfaces* to observed 3D points where certain fundamental issues and practical algorithms are discussed.

Part I

CHAPTER 2

Fitting of Quadratic Curves: Theory

In this chapter, we direct our interest toward some basic theoretical issues of fitting quadratic curves that are rarely discussed in the literature, but are essential for understanding of practical fitting algorithms. Our main problem is finding the best fitting object such as a circle, an ellipse or another quadratic curve for a given set of data points. This is an optimization problem, it consists of minimization of a certain objective function. So it raises a fundamental theoretical question: Does the solution always exist? In other words, we would like to know whether the objective function always has a minimum. We note that our objective function is defined so that it never takes negative values, i.e., its absolute possible minimum is zero.

The question of existence of the best fit is not only of theoretical interest but also practically relevant. For example, knowing under what conditions the problem does not have a solution might help us understand why the computer algorithm keeps diverging, or returns nonsense, or crashes altogether. While the cases where the best fit does not theoretically exist may be exceptional, nearby cases may be practically hard to handle, as the best fitting object may be extremely difficult to find.

Section 2.1 gives theoretical analysis needed for proving the existence of the best fit and the main existence theorem. Section 2.2 analyzes most common fitting models in two-dimensional practical applications. Section 2.3 investigates how frequently best fitting circles or ellipses fail to exist. Section 2.4 introduces sufficient and deficient models.

2.1. Existence of the Best Fit

2.1.1. Definition of Distances. Since geometric fitting problems consists of minimization of geometric distances from the given points to a model object, we first

review the necessary definitions of distances. Given two points $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$, the standard geometric (or Euclidean) distance is computed by

$$\text{dist}(P_1, P_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}. \tag{2.1}$$

Given a point P and a set $S \subset \mathbb{R}^2$, the distance from P to S is defined by

$$\text{dist}(P, S) = \inf_{Q \in S} \text{dist}(P, Q), \tag{2.2}$$

When a minimum in (2.2) exists, then there is a point $Q \in S$ closest to P, i.e., such that $\text{dist}(P, S) = \text{dist}(P, Q)$. This is always possible when S is a closed set. Fortunately, the model objects that are usually fitted to given points - lines, circles, ellipses and other conics - are all closed sets. It follows that there does exist a closest point $Q \in S$ to the point P such that

$$\text{dist}(P, S) = \text{dist}(P, Q) = \min_{Q' \in S} \text{dist}(P, Q'), \tag{2.3}$$

As one needs to use orthogonal projection, the distance from P to S is often called *orthogonal distance*.

Given two sets $S_1, S_2 \subset \mathbb{R}^2$, the distance between S_1 and S_2 is defined by

$$\text{dist}(S_1, S_2) = \inf_{P_1 \in S_1, P_2 \in S_2} \text{dist}(P_1, P_2), \tag{2.4}$$

This is the so called *shortest* distance from S_1 to S_2. The infimum in (2.4) may not be replaced by a minimum even if both sets S_1 and S_2 are closed. However, if one set (say, S_1) is closed and the other (S_2) is compact (a set $S \subset \mathbb{R}^2$ is compact if it is closed and bounded) ,then the infimum in (2.4) can always be replaced by a minimum. Note that circles and ellipses are closed and bounded, i.e., compact. On the other hand, lines and hyperbolas are closed but not bounded.

We also need Hausdorff distance to measure the overall difference or closeness of two sets. Given two sets $S_1, S_2 \subset \mathbb{R}^2$, the Hausdorff distance between S_1 and S_2 is defined by

$$\text{dist}_{\text{H}}(S_1, S_2) = \max\Big\{ \sup_{P_1 \in S_1} \text{dist}(P_1, S_2), \sup_{P_2 \in S_2} \text{dist}(P_2, S_1) \Big\}. \tag{2.5}$$

Basically, the Hausdorff distance is the longest distance you have to travel if you need to move from one set to the other or vice versa.

In real applications, the set of experimental points is finite, hence bounded, so all the data points lie in some rectangle R,

$$R = \{-A \le x \le A, \quad -B \le y \le B\}, \tag{2.6}$$

which we assume to be closed and for the moment will play the role of our "window" through which we look at the plane. Now consider two sets within such finite window R. Let us define the Hausdorff distance between them,

$$\mathrm{dist}_{\mathrm{H}}(S_1, S_2; R) = \max\Big\{ \sup_{P \in S_1 \cap R} \mathrm{dist}(P, S_2), \sup_{Q \in S_2 \cap R} \mathrm{dist}(Q, S_1) \Big\}. \tag{2.7}$$

This is our modification of the classical Hausdorff distance between sets, which means it is the longest distance you have to travel if you need to move from one set to the other or vice versa, provided you start *within* R. The formula (2.7) applies whenever both sets, S_1 and S_2, intersect the window R. If only one set, say S_1, intersects R, we modify (2.7) as follows:

$$\mathrm{dist}_{\mathrm{H}}(S_1, S_2; R) = \sup_{P \in S_1 \cap R} \mathrm{dist}(P, S_2). \tag{2.8}$$

A similar modification is used if only S_2 intersects R. If neither set intersects the window R, we simply set $\mathrm{dist}_{\mathrm{H}}(S_1, S_2; R) = 0$.

2.1.2. Convergence of Sequences of Sets. Now we are ready to introduce an important concept which serves as our basic tool for proving existence: convergence for sequences of sets.

Let $S_n \subset \mathbb{R}^2$ be some sets and $S \subset \mathbb{R}^2$ another set. We say that the sequence S_n converges to S if for any finite window R we have

$$\mathrm{dist}_{\mathrm{H}}(S_n, S; R) \to 0 \quad \text{as} \quad n \to \infty. \tag{2.9}$$

For bounded objects, like circles or ellipses, the above convergence is equivalent to the convergence with respect to the Hausdorff distance. That is, a sequence of

bounded sets S_n converges to a bounded set S if $\mathrm{dist}_{\mathrm{H}}(S_n, S) \to 0$ as $n \to \infty$. However for unbounded objects, such as lines and hyperbolas, we have to use our window-restricted Hausdorff distance and formula (2.9).

Remark: Our definition of convergence is intuitively clear, but some complications may arise if it is used too widely. For example, consider again the sequence of lines $L_n = \{y = x/n\}$ that, as we know, converges to the x axis $L = \{y = 0\}$. Let $L' \subset L$ be a subset of L consisting of points whose x coordinates are rational numbers. Note that L' is a dense subset of L, i.e., the closure of L' is the entire L (this means that if we add to L' all its limit points in $\mathbb{R}^2$, we get the whole line L). It is not hard to see that $L_n \to L'$, as $n \to \infty$. Thus, one sequence, L_n, has two distinct limits, one is L and the other is L'. In mathematics, a convergent sequence having multiple limits may cause various undesirable complications. Such a situation is regarded as a pathology. In this pathological example, the limit set L (a line) is closed, but the other limit set L' is not. From now on, to avoid pathological situations, we will assume that all our sets $S \subset \mathbb{R}^2$ are closed. As we pointed out in the section 2.1.1, all model objects of interest - lines, circles, ellipses and other conics - are closed sets.

2.1.3. Continuity of the Objective Function. Our analysis of the problem of minimization of geometric distances from the given points to a model object is based on the continuity of the objective function. Recall that the function to be minimized is the sum of squares of the distances from the given points to a model object:

$$\mathcal{F}(S) = \sum_{i=1}^{n} \big[\mathrm{dist}(P_i, S)\big]^2, \tag{2.10}$$

where $P_1, \ldots, P_n$ denote the given points and S a model object from the given collection $\mathbb{M}$.

THEOREM 2.1. *For any given points $P_1, \ldots, P_n$ and any collection $\mathbb{M}$ of model objects, the function $\mathcal{F}$ defined by* (2.10) *is continuous on $\mathbb{M}$. This means that if a sequence of objects $S_m \in \mathbb{M}$ converges to another object $S \in \mathbb{M}$ (in the sense defined in last section), then $\mathcal{F}(S_m) \to \mathcal{F}(S)$.*

The mathematical proof will be provided in Qizhuo Huang's dissertation, see [**33**].

2.1.4. Theorem of Existence. Our goal is to choose $S_{\text{best}} \in \mathbb{M}$ on which the function $\mathcal{F}$ takes its minimum value, i.e., such that

$$\mathcal{F}(S_{\text{best}}) \leq \mathcal{F}(S) \quad \text{for all} \quad S \in \mathbb{M}, \qquad \text{or} \qquad S_{\text{best}} = \arg\min_{S \in \mathbb{M}} \mathcal{F}(S). \tag{2.11}$$

The model object S_{best} is called the *best fit* or *the closest object* to the given points. Our fitting problem has a solution if S_{best} exists. Here we are preoccupied with the existence of S_{best}. Does it always exist? If not, what issues can this cause? And how can we resolve them? In fact, one can prove that the best fitting object always exists whenever $\mathbb{M}$ is closed. The theorem is as follows:

THEOREM 2.2. *Suppose the given collection $\mathbb{M}$ of model objects is closed. (meaning that if a sequence of objects $S_m \in \mathbb{M}$ converges, to an object S, then S also belongs to $\mathbb{M}$). Then for any given points $P_1, \ldots, P_n$ there exists the best fitting object $S_{\text{best}} \in \mathbb{M}$, i.e., the objective function $\mathcal{F}$ attains its global minimum on $\mathbb{M}$.*

The key ingredients of the proof will be the continuity of the objective function and the compactness of a restricted domain of that function. Then use the fact that a continuous (real-valued) function on a compact set always takes maximum and minimum values on that set. Detailed proof will be provided in Qizhuo Huang's dissertation, see [**33**].

2.2. Model Objects in 2D

We have described the general theory for the existence of the best fit and we know that to check the existence of the best fitting object, one has to find out if the model collection is closed or not. So in this section, we will look at real fitting models which are common in practical applications. **Note:** This section is part of Qizhuo Huang's dissertation, but it is necessary to briefly discuss it before moving on to my work starting from next section 2.3.

2.2.1. Lines. Before looking at circle fitting and ellipse fitting model, let us start with the simplest case, model collection of lines. Suppose one fits lines to data points, so the model collection $\mathbb{M}_{\mathrm{L}}$ consists of all lines $L \subset \mathbb{R}^2$. Given data points $P_1, \ldots, P_n$, the best fitting line L_{best} minimizes the sum of squares of the distances from $P_1, \ldots, P_n$ to the line. It is easy to see that a sequence of lines L_m can only converge to another line $L_0 \subset \mathbb{R}^2$, hence the collection $\mathbb{M}_{\mathrm{L}}$ of lines is *closed*. This guarantees that the fitting problem always has a solution.

2.2.2. Circles. It is a common task in computer vision (as well as some sciences and industry) to fit circles to data points. Now the model collection $\mathbb{M}_{\mathrm{C}}$ consists of all circles $C \subset \mathbb{R}^2$. Given data points $P_1, \ldots, P_n$, the best fitting circle C_{best} minimizes the sum of squares of the distances from $P_1, \ldots, P_n$ to the circle.

The question is does this circle always exist? Are there any cases where it doesn't exist? Let us look at this example: suppose our model objects are circles, and our given points are $P_1 = (-1, 0)$, $P_2 = (0, 0)$ and $P_3 = (1, 0)$. Then in this collection model, we can find a sequence of circles S_m defined by

$$x^2 + (y - m)^2 = m^2$$

which will fit the points progressively better as m grows and make $\mathcal{F}$ arbitrarily close to zero (see Figure 2.1). But since no circle can interpolate $n \geq 3$ collinear points, we will always have $\mathcal{F} > 0$. For any circle, one can always find another circle that fits the data better. None of them would be the best fit. Obviously the best fit here is given by the straight line through the data points, which yields $\mathcal{F} = 0$. Thus the circle fitting problem has no solution in this case.

In fact, a sequence of circles C_m can converge to an object of three possible types: (see Figure 2.2)

- a circle $C_0 \subset \mathbb{R}^2$;
- a line $L_0 \subset \mathbb{R}^2$;
- a single point (singleton) $P_0 \in \mathbb{R}^2$.

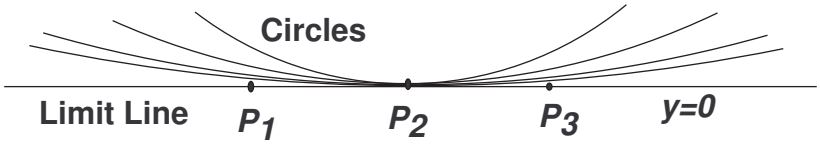

Figure 2.1: Example of non-existence of circle fitting problem

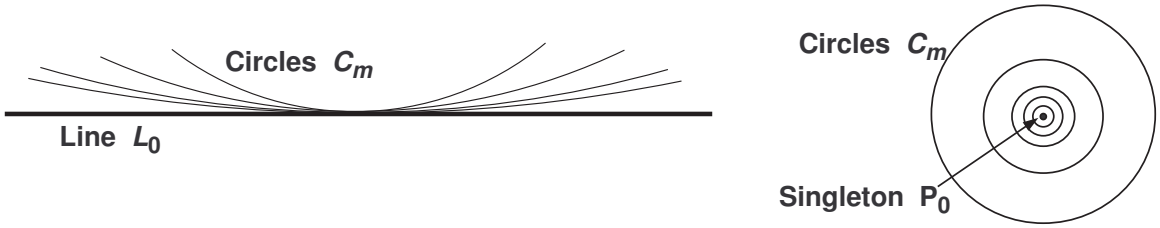

Figure 2.2: Limit objects of collection of circles

Note that singletons can be regarded as degenerate circles whose radius are zero, so they can be regarded as part of the model collection. But lines cannot be dealt with so easily, because they are neither circles nor degenerate circles. So we can see that the collection $\mathbb{M}_{\mathrm{C}}$ of circles is *not* closed (in our sense). As a result, the circle fitting problem does not always have a solution.

In order to guarantee the existence of the best fitting object, one needs to extend the given collection $\mathbb{M}_{\mathrm{C}}$ by adding all its "limit points", in this case - lines. The extended collection will include all circles and all lines, i.e.,

$$\bar{\mathbb{M}}_{\mathrm{C}} = \mathbb{M}_{\mathrm{C}} \cup \mathbb{M}_{\mathrm{L}}. \tag{2.12}$$

In topology, if we add all "limit points" to a set A, we get a larger set called the closure of A, it is denoted by $\bar{A}$. So we denote it by $\bar{\mathbb{M}}_{\mathrm{C}}$ the closure of $\mathbb{M}_{\mathrm{C}}$.

Now the (extended) circle fitting problem always has a solution:

THEOREM 2.3. *For any data points $P_1, \ldots, P_n$, there is a best fitting object $S_{\mathrm{best}} \in \bar{\mathbb{M}}_{\mathrm{C}}$ that minimizes the sum of squares of the distances to $P_1, \ldots, P_n$.*

One has to keep in mind, though, that the best fitting object may be a line, rather than a circle.

2.2.3. Ellipses. It is also a common task in computer vision to fit ellipses to data points. Now the model collection $\mathbb{M}_{\mathrm{E}}$ consists of all ellipses $E \subset \mathbb{R}^2$. Given data points $P_1, \ldots, P_n$, the best fitting ellipse E_{best} minimizes the sum of squares of the distances from $P_1, \ldots, P_n$ to the ellipse.

A sequence of ellipses E_m may converge to an object of many possible types:

- the limit object may be an ellipse $E_0 \subset \mathbb{R}^2$;
- as circles are ellipses, the limit object may be a line or a singleton (see above);
- a limit object can be a parabola, or a pair of parallel lines, or a segment of a line, or a half-line (a ray). See Figure 2.3.

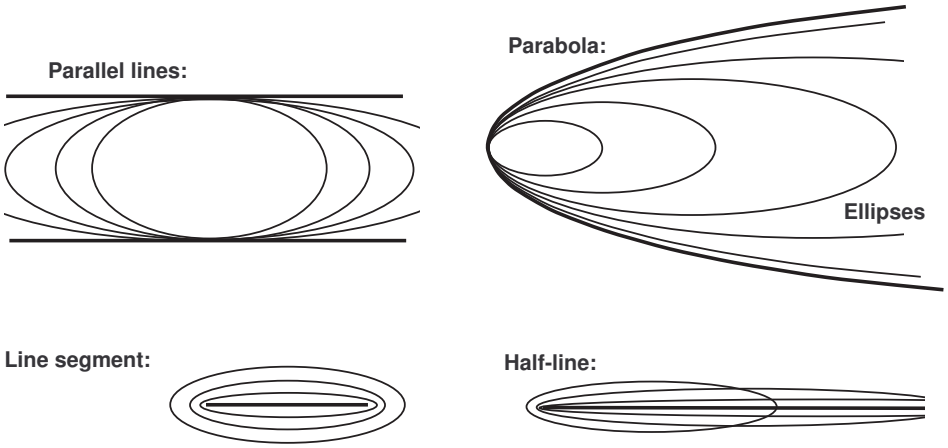

Figure 2.3: Limit objects of collection of ellipses

Well, singletons can be regarded as degenerate circles and line segments can be regarded as degenerate ellipses, whose minor axis is zero. In fact, both singletons and line segments, as well as half-lines, can be ignored based on the redundancy principle, see discussion in Chapter 1. But lines, pairs of parallel lines, and parabolas cannot be dealt with so easily. We see that the collection $\mathbb{M}_{\mathrm{E}}$ of ellipses is not closed. As a result, the ellipse fitting problem does not always have a solution. Examples may

be sets of points that are placed on a line, on two parallel lines, or on a parabola. Actually there are a lot more other examples as well, see 2.4.1.

In order to guarantee the existence of the best fitting object, one needs to extend the collection $\mathbb{M}_{\mathrm{E}}$ of ellipses by adding all its "limit points", in this case - *lines*, *pairs of parallel lines*, and *parabolas*. We denote that extended collection by $\bar{\mathbb{M}}_{\mathrm{E}}$:

$$\bar{\mathbb{M}}_{\mathrm{E}} = \mathbb{M}_{\mathrm{E}} \cup \mathbb{M}_{\mathrm{L}} \cup \mathbb{M}_{\parallel} \cup \mathbb{M}_{\cup}, \tag{2.13}$$

where $\mathbb{M}_{\parallel}$ denotes the collection of pairs of parallel lines, and $\mathbb{M}_{\cup}$ the collection of parabolas.

Now the (extended) ellipse fitting problem always has a solution:

THEOREM 2.4. *For any data points $P_1, \ldots, P_n$, there is a best fitting object $S_{\mathrm{best}} \in \bar{\mathbb{M}}_{\mathrm{E}}$ that minimizes the sum of squares of the distances to $P_1, \ldots, P_n$.*

One has to keep in mind, though, that the best fitting object may be a *line*, or a *pair of parallel lines*, or a *parabola*, rather than an ellipse.

2.2.4. Quadratic Curves. Let us consider a more general task of fitting quadratic curves (also known as *conic sections* or simply *conics*) to data points. Now the model collection $\mathbb{M}_{\mathrm{Q}}$ consists of all quadratic curves, by which we mean all ellipses, parabolas, and hyperbolas. Given data points $P_1, \ldots, P_n$, the best fitting quadratic curve minimizes the sum of squares of the distances from $P_1, \ldots, P_n$ to the curve.

A sequence of quadratic curves may converge to an object of many possible types:

- the limit object may be a quadratic curve;
- as ellipses are quadratic curves, the limit object may be a line, a pair of parallel lines or a singleton;
- hyperbolas may converge to a pair of intersecting lines or two opposite half-lines (rays). See Figure 2.4.

Now every half-line and two opposite half-lines are a part (subset) of a full line, so they can be incorporated into lines. Again we treat singletons as degenerate circles.

Figure 2.4: Limit objects of collection of hyperbolas

But pairs of parallel lines and pairs of intersecting lines cannot be dealt with so easily. We see that the collection $\mathbb{M}_{\mathrm{Q}}$ of quadratic curves (conics) is *not closed* (in our sense). As a result, the conic fitting problem does not always have a solution.

In order to guarantee the existence of the best fitting object, one needs to extend the collection $\mathbb{M}_{\mathrm{Q}}$ of quadratic curves by adding all its "limit points" - *lines*, *pairs of parallel lines*, and *pairs of intersecting lines*. We denote that extended collection by $\bar{\mathbb{M}}_{\mathrm{Q}}$:

$$\bar{\mathbb{M}}_{\mathrm{Q}} = \mathbb{M}_{\mathrm{Q}} \cup \mathbb{M}_{\mathrm{L}} \cup \mathbb{M}_{\parallel} \cup \mathbb{M}_{\times}, \tag{2.14}$$

where $\mathbb{M}_{\parallel}$ was introduced in (2.13) and $\mathbb{M}_{\times}$ denotes the collection of pairs of intersecting lines.

Now the (extended) conic fitting problem always has a solution:

THEOREM 2.5. *For any data points $P_1, \ldots, P_n$, there is a best fitting object $S_{\mathrm{best}} \in \bar{\mathbb{M}}_{\mathrm{Q}}$ that minimizes the sum of squares of the distances to $P_1, \ldots, P_n$.*

One has to keep in mind, though, that the best fitting object may be a *line*, or a *pair of parallel lines*, or a *pair of intersecting lines*, rather than a conic.

2.3. Probabilistic Approach

We showed in Section 2.1 that the collection $\mathbb{M}$ of model objects $\{S\}$ used to fit observed points $P_1, \ldots, P_n$ has to be closed if we want the fitting problem to have a solution. If not, then we have to close it up, i.e., add to $\mathbb{M}$ all objects that are

obtained as limits of existing objects. The new, extended collection is called the topological closure of $\mathbb{M}$ and denoted by $\bar{\mathbb{M}}$. We will use the following terms:

DEFINITION 2.1. *The original objects $S \in \mathbb{M}$ are called primary objects and the objects that have been added, i.e., $S \in \bar{\mathbb{M}} \setminus \mathbb{M}$, are called secondary objects.*

As we have seen in section 2.2, the collection of circles $\mathbb{M}_{\mathrm{C}}$ is not closed. In order to close it up, one has to add lines to it. That is, one really has to operate with the extended collection

$$\bar{\mathbb{M}}_{\mathrm{C}} = \mathbb{M}_{\mathrm{C}} \cup \mathbb{M}_{\mathrm{L}} \tag{2.15}$$

of circles and lines. Circles are primary objects, and lines are secondary objects.

In practical applications, secondary objects may be unwanted, or not welcome. For example, when one fit circles or ellipses, they might not want lines or parabolas or a pair of parallel lines (which are secondary objects). Hence the question is how frequently these unwanted secondary objects occur as the best fits?

2.3.1. How Frequently does the Best Fitting Circle Fail to Exist? In most applications data points are observed with some random noise. Thus the noise has a probability distribution. One usually assumes that the noise added to every point is a 2D normally distributed vector which is added independently to every point.

Let us see how frequently the best fitting circle would fail to exist under these standard assumptions. Let us begin with the simplest case of $n = 3$ points. If they are not collinear, then they can be interpolated by a circle. If they are collinear and distinct, there is no interpolating circle, so the best fit is achieved by a line.

Under the above assumptions of independent normally distributed noise, the collinearity occurs with probability zero. In simple practical terms, it is "impossible", it "never happens". If we generate three data points by using a computer random number generator, we will practically never see collinear points. All our practical experience tells us: the best fitting circle *always* exists. Perhaps for this reason the

circle fitting problem is usually studied without including lines in the collection of model objects, but there are notable exceptions such as [**63**].

In the case of $n > 3$ data points, the best fitting circle fails to exist when the points are collinear, but there are other instances, too. For example, let $n = 4$ points be at $(0, 1)$, $(0, -1)$, $(A, 0)$, and $(-B, 0)$ for some large $A, B \gg 1$. See Figure 2.5. Then it is not hard to check, by direct inspection, that the best fitting circle fails to exist, and the best fit is achieved by the line $y = 0$.

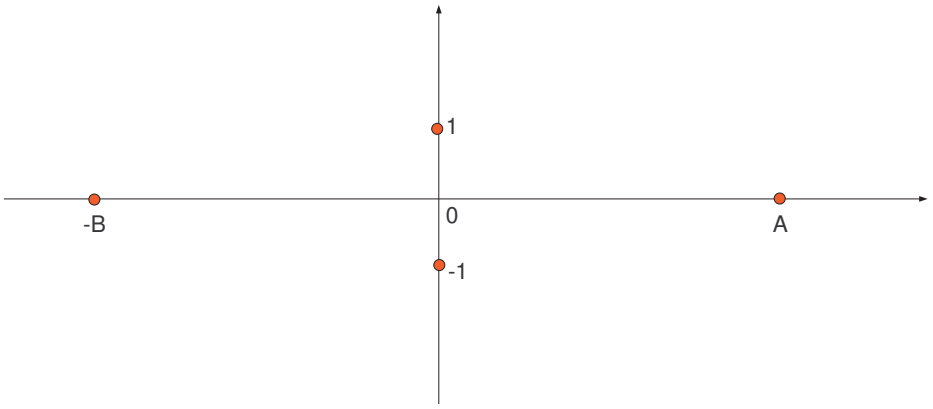

Figure 2.5: Example of four points that are not collinear for which the best fitting circle fails to exist

It is not easy to describe all sets of points for which the best fitting circle fails to exist. Such a description was given in [**18**] (see Theorem 8 on page 68 there). Here we only state the final conclusion: for every $n > 3$, the best fitting circle exists with probability one, i.e., the failure to exist occurs with probability zero.

In other words, no matter how large or small the data set is, the best fitting circle will exist with probability one, so practically we never have to resort to secondary objects (lines), we never have to worry about existence. The model of circles is adequate, it does not really require an extension. This fact explains why lines are often ignored in practice and one works with circles only.

2.3.2. How Frequently does the Best Fitting Ellipse Fail to Exist? Now let us examine the model collection $\mathbb{M}_{\mathrm{E}}$ of ellipses and see how frequently the best fitting ellipse fails to exist. It is particularly easy to deal with the simplest case of $n = 5$ data points. For any set of distinct 5 points in a general linear position (which means that no three points are collinear), there exists a unique quadratic curve (conic) that passes through all of them (i.e., interpolates them); That conic may be an ellipse, or a parabola, or a hyperbola. If 5 points are not in generic linear position, i.e., at least three of them are collinear, then they can be interpolated by a degenerate conic, a pair of lines.

If the interpolating conic is an ellipse, then obviously that ellipse is the best fit, the objective function takes it absolute minimum value - zero. What if it is a parabola or a pair of parallel lines? Then it is a secondary object in the extended model (2.13), and we have an unwanted event: a secondary object provides the best fit. This, however, occurs with probability zero, so it is not a real concern.

But what if the interpolating conic is a hyperbola or a pair of intersecting lines? Then the situation is less clear, as such an interpolating conic does not belong to the extended collection $\bar{\mathbb{M}}_{\mathrm{E}}$; see (2.13). Is it possible now that the best fitting object $S_{\mathrm{best}} \in \bar{\mathbb{M}}_{\mathrm{E}}$, i.e., the best fitting object from the collection (2.13), is an ellipse?

The answer is *No*. Here are our reasons. If there was a best fitting ellipse $E_{\mathrm{best}} \in \bar{\mathbb{M}}_{\mathrm{E}}$, then no other ellipse could provide a better fit, i.e., for any other ellipse $E \in \mathbb{M}_{\mathrm{E}}$ we would have $\mathcal{F}(E) \geq \mathcal{F}(E_{\mathrm{best}})$. And we have proved that this is impossible.

THEOREM 2.6 ([**33**]). *Suppose $n = 5$ data points are given. Then no ellipse E or hyperbola H can provide a local minimum of the objective function $\mathcal{F}$.*

Basically it says that for any ellipse E that does not interpolate our 5 points (i.e., for which $\mathcal{F}(E) > 0$) there exist other ellipses E' arbitrary close to E such that $\mathcal{F}(E') < \mathcal{F}(E)$. Similarly, for any hyperbola H that does not interpolate our 5 points (i.e., for which $\mathcal{F}(H) > 0$) there exist other hyperbolas H' arbitrary close to H such

that $\mathcal{F}(H') < \mathcal{F}(H)$. This theorem is proved by Q.Huang and proof is avaliable in [**33**].

Thus the best fitting object $S_{\text{best}} \in \bar{\mathbb{M}}_{\text{E}}$ is now a secondary one (a parabola, or a line, or a pair of parallel line), i.e., an unwanted event occurs. As a matter of fact, this really happens with a positive probability.

Numerical Experiment for $n = 5$. One can easily estimate the probability of the above unwanted event numerically. For a given probability distribution of the data points (x_i, y_i), $i = 1, \dots, 5$, one can generate random samples of $n = 5$ points (x_i, y_i), $1 \le i \le 5$, and for each sample find the interpolating conic. The latter can be found by solving a system of equations

$$Ax_i^2 + 2Bx_iy_i + Cy_i^2 + 2Dx_i + 2Ey_i + F = 0, \qquad i = 1, \dots, 5$$

for unknown coefficients A, B, C, D, E, F (which only need to be determined up to a scalar multiple), and then finding the type of the respective conic by standard rules of analytic geometry. For example, there are three basic types of them:

- ellipse, characterized by $AC - B^2 > 0$
- hyperbola, corresponding to $AC - B^2 < 0$
- parabola, which occurs when $AC - B^2 = 0$

Every time that conic happens to be other than an ellipse, an unwanted event occurs, i.e., the best fit from the collection (2.13) is provided by a secondary object.

We have run the above experiment with a 2D standard normal distribution. Precisely, each point (x_i, y_i) was selected independently from a 2D normal distribution $\mathcal{N}(\mathbf{u}, \mathbf{V})$ with mean $\mathbf{u} = (0, 0)$ and covariance matrix $\mathbf{V} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. In other words, we selected each coordinate x_i and y_i from a standard normal distribution $\mathcal{N}(0, 1)$.

We found that the random points were interpolated by an ellipse with probability 22% and by a hyperbola with probability 78%. Other conics, including parabolas,

never turned up; They occur with probability zero. This is a striking result: hyperbolas actually dominate over ellipses! Thus the best fitting ellipse *fails to exist* with a probability as high as 78%.

In another experiment we sampled points from the unit square $[0,1] \times [0,1]$ with uniform distribution (i.e., we selected each coordinate x_i and y_i from the unit interval $[0,1]$). In this experiment ellipses turned up with probability 28% and hyperbolas with probability 72%. Again an overwhelming domination of hyperbolas!

Remark on Different Types of Distributions. One might think that different normal distributions might give different results. Surprisingly, the answer is No. The percentages of ellipses (22%) and hyperbolas (78%) are the same when points are sampled from any 2D normal distribution $\mathcal{N}(\mathbf{u}, \mathbf{V})$, with any mean $\mathbf{u} \in \mathbb{R}^2$ and any 2×2 covariance matrix $\mathbf{V}$. Indeed, any 2D normal distribution can be transformed to a standard normal distribution by a linear transformation of the plane $\mathbb{R}^2$. Now under linear transformations conics are transformed to conics, and their types are preserved (i.e., ellipses are transformed to ellipses, hyperbolas to hyperbolas, etc). Therefore the type of the interpolating conic cannot change.

Also, the percentages of ellipses (28%) and hyperbolas (72%) are the same when points are sampled from any rectangular domain $R \subset \mathbb{R}^2$ in the plane. Indeed, any rectangle R can be transformed to the unit square $[0,1] \times [0,1]$ by a linear transformation of the plane $\mathbb{R}^2$, and the same argument as above applies. We also did experiments for randomly selected samples of $n > 5$ points, see Sec. 2.4.

Conclusion. The collection of ellipses is not a sufficient model for fitting purposes. This means that there is a real chance that for a given set of data points no ellipse could be selected as the best fit to the points, i.e., the ellipse fitting problem would have no solution. In other words, whenever the best fitting ellipse fails to exist, the ellipse fitting procedure attempts to move beyond the collection of ellipses, and ends up on the border of that collection and then returns a secondary object: a parabola or a pair of lines. Even more, the probability of such events occurring is not low.

2.4. Sufficient and Deficient Model

DEFINITION 2.2. *We say that a collection* $\mathbb{M}$ *of model objects is* sufficient *(for fitting purposes) if the best fitting object* S_{best} *exists with probability one, assuming that the data points* $P_1, \ldots, P_n$ *are independent normally distributed random variables.*

In other words, with probability one the best fitting object S_{best} belongs to the original collection $\mathbb{M}$, rather than its extension $\bar{\mathbb{M}} \setminus \mathbb{M}$. As we have just seen, the collection of circles is sufficient (adequate) for fitting purposes.

DEFINITION 2.3. *Model collections that are not sufficient, as defined above, will be called* deficient.

Their deficiency indicates that they should be substantially extended for the fitting problem to have a reasonable (adequate) solution with probability one. We have seen that the collection of ellipses is *not sufficient* for fitting purposes.

2.4.1. Numerical Tests for Deficient Models. Here we present similar results for randomly selected samples of $n > 5$ points. The analysis of samples of $n > 5$ points requires more sophisticated numerical tests than those used in section 2.3.2.

Description of Numerical Tests. First let me describe our numerical tests. Recall that our basic task is to see what type of conic (ellipse or hyperbola) provides the best fit for a given set of $n > 5$ points. The best fitting conic is the one minimizing the sum of squares of geometric (orthogonal) distances to the given points.

For $n > 5$ we cannot hope for an interpolating conic, thus we have to employ a standard minimization procedure to find the minimum of the objective function $\mathcal{F}$. The function $\mathcal{F}$ is the sum of squares of geometric distances from a conic to the given points. Its global minimum gives the best fitting conic. We use the Levenberg-Marquardt minimization algorithm (to be described in chapter 6) to find a minimum of the objective function $\mathcal{F}$.

The Levenberg-Marquardt algorithm, as all standard minimization procedures, requires an initial guess. Given a set of $n > 5$ points, we generate 1000 randomly chosen initial guesses (see below) and then we use each one to initialize the Levenberg-Marquardt procedure. We run the latter until it converges to a limit conic. Thus we get 1000 limit conics, which can be regard as local minima of the objective function. From those 1000 conics we select the one that gives the smallest value of $\mathcal{F}$ and regarded it as the global minimum, i.e., the best fitting conic.

This numerical scheme is rather computationally intense, so for each given distribution and each $n > 5$ we only generate 10,000 random samples of n points to determine the percentages of samples for which the best fitting conic is an ellipse or hyperbola.

The Choice of Initial Guesses. Initial guesses are generated as follows. Let $R = [x_1, x_2] \times [y_1, y_2] \in \mathbb{R}^2$ be a rectangle that contains all our $n > 5$ points (the construction of R is described below). We generate 1000 random five-point sets $\{Q_1, \dots Q_5\}$ with uniform distribution in the rectangle R. (In other words, we select five x-coordinates in the interval $[x_1, x_2]$ randomly and five y-coordinates in the interval $[y_1, y_2]$ randomly and combine them to get five pairs of xy coordinates that we treat as five points in R.) For each five-point set $\{Q_1, \dots Q_5\}$ in R we find the unique conic that interpolates those five points by solving the corresponding system of linear equations. Then we use that conic as an initial guess.

Rational for Our Choice of Initial Guesses. Our method of generating random initial guesses has an advantage that all the resulting conics pass through the rectangle R, i.e., conics that are absurdly far from our n data point will never be selected. At the same time, our five points $\{Q_1, \dots Q_5\}$ are selected totally randomly in R and not related to the given data points; this ensures a wide variety of initial approximations.

Construction of the Rectangle R. If the sample of $n > 5$ points is selected randomly from a rectangular domain, then it is natural to use that domain as the rectangle R. If the sample of $n > 5$ points is selected randomly from a normal

distribution, then no natural rectangle can be found. In that case we can construct $R = [x_1, x_2] \times [y_1, y_2]$ by setting x_1 and x_2 to the smallest and largest x-coordinate of our data points, and y_1 and y_2 to the smallest and largest y-coordinate of our data points.

Numerical Results.

Standard Normal Distribution: $n > 5$**.** Table 2.1 shows the percentages of samples of $n > 5$ points for which the best fitting conic is an ellipse or a hyperbola. The samples here are generated randomly with a standard normal distribution (mean zero and the identity covariance matrix).

Percentage(%)	$n = 5$	$n = 6$	$n = 8$	$n = 10$	$n = 20$	$n = 50$
Best Fit is Hyperbola	78	77	77	76	73	79
Best Fit is Ellipse	22	23	23	24	27	21

Table 2.1: Percentages for best fitting conic: samples of $n > 5$ from standard normal distribution

We see that the percentages fluctuate but do not change much as n grows; the domination of hyperbolas persists.

Non-Standard Normal Distribution: $n > 5$**.** We also used a non-standard normal distribution that had covariance matrix $\mathbf{V} = \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix}$ (so that the scattering along one axis is twice as large as that along the other axis). Table 2.2 shows the corresponding percentages. Here too, the percentages fluctuate but do not change much; the domination of hyperbolas persists.

Rectangular Distribution: $n > 5$**.** A more intriguing picture appears when samples are generated from a rectangular distribution, instead of normal distribution. Table 2.3 below represent percentages for a uniform distribution in the unit square $[0, 1] \times [0, 1]$.

Percentage(%)	$n = 5$	$n = 6$	$n = 8$	$n = 10$	$n = 20$	$n = 50$
Best Fit is Hyperbola	78	78	76	74	75	67
Best Fit is Ellipse	22	22	24	26	25	33

Table 2.2: Percentages for best fitting conic: samples of $n > 5$ from non-standard normal distribution

Percentage(%)	$n = 5$	$n = 6$	$n = 10$	$n = 15$	$n = 20$	$n = 30$	$n = 50$	$n = 100$
Best Fit is Hyperbola	72	68	60	54	48	43	40	34
Best Fit is Ellipse	28	32	40	46	52	57	60	66

Table 2.3: Percentages for best fitting conic: samples of $n > 5$ from rectangular distribution

Here the percentage of ellipses noticeably grows from 28% to 66%. It appears that it will keep growing to 100% as n gets larger. We give a reason for this fact in next section 2.4.2. However, when samples are generated from a rectangle $[0, 2] \times [0, 1]$, then the percentage of ellipses does not grow that much. We give an explanation to this fact, too.

Data Points Sampled along An Ellipse. In the above tests the data points were taken from some general distribution , normal or rectangular. This produces completely irregular ("chaotic") samples without any predefined pattern, elliptic or hyperbolic. One may argue that such simulated data are unrealistic. So we conducted more realistic tests. Following traditions in the literature, we positioned n points equally spaced along an elliptic arc and added some small Gaussian noise.

More precisely, we chose the ellipse

$$\frac{x^2}{4} + \frac{y^2}{1} = 1,$$

which has semiaxes 2 and 1. The elliptical arc along which the data points were placed was chosen in three different ways (see Figure 2.6):

- the upper half of the ellipse
- the right half of the ellipse
- the upper right quarter of the ellipse

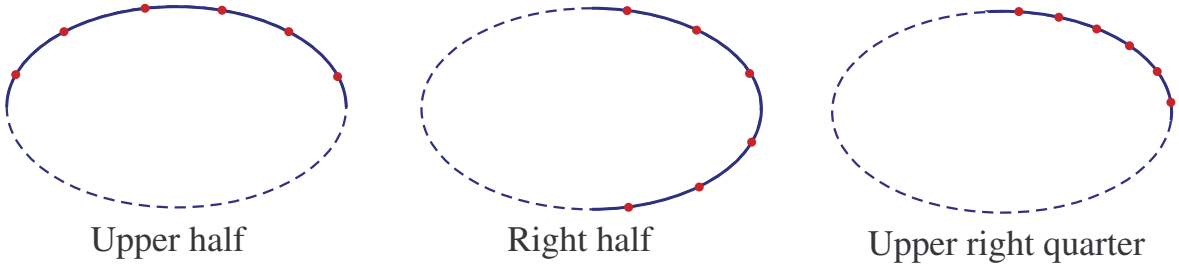

Figure 2.6: Elliptical arcs along which the data points were placed

The Gaussian noise was added at two levels: $\sigma = 0.05$ and $\sigma = 0.1$. Adding Gaussian noise at level σ means that the x and y coordinates of every point are perturbed by a normal random variable with mean zero and standard deviation σ. The noise level $\sigma = 0.1$ is not too big for many computer vision applications, and it is also quite small by common standards in statistical applications.

Tables 2.4 and 2.5 below show the results of our experiments for $n = 6$ and $n = 8$ points:

$n = 6$	Upper half of ellipse		Right half of ellipse		Quarter of ellipse	
Percentage(%)	$\sigma = 0.05$	$\sigma = 0.1$	$\sigma = 0.05$	$\sigma = 0.1$	$\sigma = 0.05$	$\sigma = 0.1$
Best Fit is Hyperbola	4	26	3	18	62	75
Best Fit is Ellipse	96	74	97	82	38	25

Table 2.4: Percentages of samples for which best fit is ellipse or hyperbola: data points sampled along an ellipse, $n = 6$

$n = 8$	Upper half of ellipse		Right half of ellipse		Quarter of ellipse	
Percentage(%)	$\sigma = 0.05$	$\sigma = 0.1$	$\sigma = 0.05$	$\sigma = 0.1$	$\sigma = 0.05$	$\sigma = 0.1$
Best Fit is Hyperbola	1	17	1	13	57	75
Best Fit is Ellipse	99	83	99	87	43	25

Table 2.5: Percentages of samples for which best fit is ellipse or hyperbola: data points sampled along an ellipse, $n = 8$

Conclusions. We see that even in realistic numerical tests where the data points form a clear elliptic pattern, the best fitting conic may be a hyperbola with positive probability (which is sometimes quite high!). Whenever this happens, the ellipse fitting problem has no solution, i.e., the best fitting ellipse fails to exist. For longer elliptical arcs (half ellipse or more) this happens occasionally - just a few percents of the time - so one may simply dismiss it as a "bad luck". But for shorter elliptical arcs (quarter of ellipse or less) this happens *most of the time*. The existence of the best fitting ellipse is rather an exception than a rule. In any case one cannot dismiss samples for which the best fit is hyperbola.

2.4.2. Large Sample Limit. As the number of points n grows, according to the Law of Large Numbers in Probability Theory the normalized objective function $\frac{1}{n}\mathcal{F}$ converges to the integral

$$\mathcal{F}_\infty(S) = \iint d^2(x,y)\, dP(x,y), \qquad\qquad d(x,y) = \text{dist}[(x,y), S]$$

where $P(x,y)$ denotes the probability distribution from which the points are sampled, and S is our standard notation for the fitting model object (in the present case - a conic). Then the global minimum of the function $\mathcal{F}_\infty(S)$ would give us the best fitting object corresponding to the given probability distribution P. This way we find a conic which is the best fit (or the best approximation) to a *probability distribution*, rather than a finite set of data points.

2.4.2.1. *Limit as $n \to \infty$: Rectangular Distribution.* For a uniform distribution in a rectangle $[0, a] \times [0, 1]$ this integral becomes

$$\mathcal{F}_\infty(S) = a^{-1} \int_0^a \int_0^1 d^2(x, y)\, dy\, dx, \qquad d(x, y) = \text{dist}[(x, y), S] \tag{2.16}$$

We have integrated it numerically in order to find the best fitting conic S. The results are given below.

2.4.2.2. *Perfect Square $R = [0, 1] \times [0, 1]$.* What conic would you expect to be the best fit to a mass uniformly distributed in a square? Due to the perfect shape of the square and its many symmetries, it is natural to expect a symmetric conic, i.e., either a circle or a pair of lines, say, the diagonals of the square.

What we found was totally unexpected: the best fitting conic is an *ellipse*! It has different axes: 0.9489 and 0.6445 (assuming that the square has unit side). Its axes are aligned with the sides of the square, and its center coincides with the center of the square. Therefore, there are exactly *two* such ellipses, one is oriented horizontally and the other - vertically; see Figure 2.7. These two ellipses beat any circle and any pair of lines; they provide the global minimum of the objective function (2.16). Table 2.6 above Figure 2.7 compares the above two ellipses to the best circle (whose radius is 0.38265) and the best pair of lines (diagonals of the square).

Fitting object	Value of $\mathcal{F}$
Best ellipse (either of the two)	0.02001
Best circle (of radius 0.38265)	0.02025
Two diagonals of the square	0.02083

Table 2.6: Comparison of different fits for a uniform distribution

The fact that the best fitting conic to the mass distributed in a square is an ellipse explains why for large samples generated from the uniform distribution in a square ellipses dominate over hyperbolas.

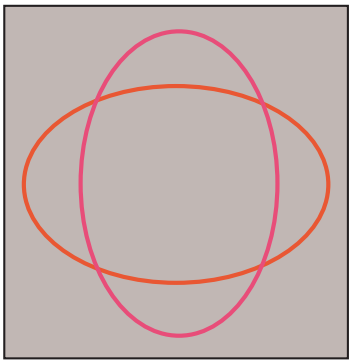

Figure 2.7: Best fit to a uniform distribution in a square

2.4.2.3. *Other Rectangles* $R = [0, a] \times [0, 1]$*, where* $a > 1$. As the rectangle R gets extended horizontally, i.e., as $a > 1$ increases, the best fitting conic changes. We found that for $a < a_1 \approx 1.2$, the best fitting conic is still an ellipse and it is unique (though it gets elongated compared to the one found for the square, when $a = 1$). But for $a > a_1$, the best fitting conic abruptly changes from an ellipse to a pair of parallel lines. Those two lines are running through the rectangle horizontally; their equations are $y = 0.25$ and $y = 0.75$. See Figure 2.8.

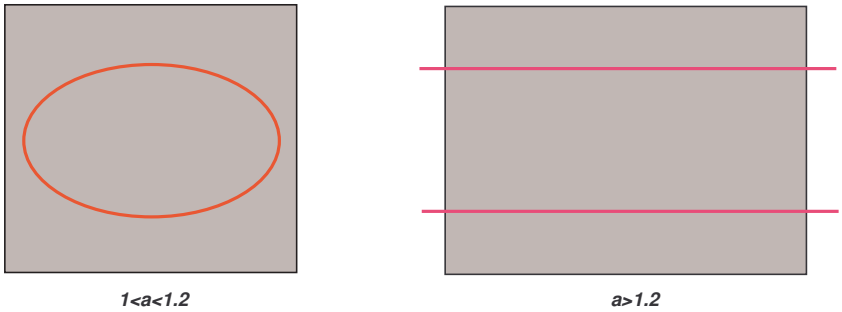

Figure 2.8: Best fit to a uniform distribution in a rectangle

This fact explains that for longer rectangles, such as $R = [0, 2] \times [0, 1]$ mentioned above, the percentage of ellipses does not grow much as n grows. It remains unclear

which type of conic dominates (ellipses or hyperbolas) for large samples generated from longer rectangles $[0, a] \times [0, 1]$, where $a > a_1$.

2.4.3. Dilemma: Fitting Ellipses versus Fitting Conics. In many applications, one specifically seeks an elliptical fit rather than hyperbolic or parabolic. In this case, however, one should be aware of the fact that there is a positive probability (sometimes quite high!)that the best fitting ellipse cannot be found. When the objective function $\mathcal{F}$ does not take a minimum on the space of ellipses (in which case, the best fitting conic is hyperbola), the minimum of $\mathcal{F}$ will be attained on parabola or a pair of parallel lines in the extended collection of ellipses. When this happens, the final choice of a good (or best) fit will fully depend on the seekers. There are three options:

(1) One can choose any fitted ellipse as a reasonable fit, as we discussed previously that no ellipse is optimal in this case. For any ellipse we found, we can always find another ellipse who fits the data points better. When iteration goes on, we can decide when we want to stop as long as the value of $\mathcal{F}$ is acceptably small.

(2) If one wants to find the actual best fitting object where the objective function $\mathcal{F}$ attains minimum, we have to appeal to the secondary objects in the extended collection of ellipses, which is usually a parabola. So in addition to ellipses, one needs to consider parabolas in the fitting progress to find a best fitting curve.

(3) On the other hand, if one can accept a parabolic fit, it would not be a bad idea to go for a hyperbolic fit since it is the real best fitting conic in the whole collection of all quadratic curves. In the larger framework of fitting all quadratic curves, the best fitting object will always be found, either an ellipse or a hyperbola.

As we can see, this is not an easy choice. If one only needs a reasonably good fit, then the first option is the fastest because one fits ellipses only; If one has a higher

expectation to the accuracy, then the second option is good, though not optimal. If one wants the best fitting object and doesn't specify the shape of the fit, then the third option will be the best approach. And from now on, we proceed with the third choice. We treat the problem of fitting ellipse to data points as a part of a more general problem of fitting conics. In fact, with this understanding, we will be able to find more efficient practical algorithms for fitting ellipses to data, which we will show later.

CHAPTER 3

Parameters for Quadratic Curves

In this chapter we will describe the most common parametrization schemes used in quadratic curve (conic) fitting applications. Conics can be described by their natural geometric parameters or by the general algebraic parameters, which both have advantages and disadvantages. Geometric parameters are more standard and useful for us to find the distances between points and conics. Algebraic parameters allow us to reach all real conics and even "imaginary" conics. Later in our work, we will see that algebraic parameters should be chosen over geometric parameters because of some compelling reasons which will be explained here.

In section 3.1 geometric parameters and its drawbacks are described. In section 3.2 algebraic parameters are discussed and a complete classification of conics is given. In section 3.3 we will compare these two parameter choices and explain why algebraic parameters is a more elegant choice for us.

3.1. Geometric Parameters of Conics

3.1.1. Ellipses: Standard Geometric Parameters. Ellipses can be described by five geometric parameters. The most popular choice is the coordinates of the center (x_c, y_c), the semi-axes a and b and the angle of tilt, α, of the major axis. There are natural restrictions on the semi-axes, $a \geq b > 0$. The angle α has period π and one often requires $\alpha \in [0, \pi)$. The full ellipse equation in these parameters is rather complicated:

$$\frac{\big[(x - x_c)\cos\alpha + (y - y_c)\sin\alpha\big]^2}{a^2} + \frac{\big[-(x - x_c)\sin\alpha + (y - y_c)\cos\alpha\big]^2}{b^2} = 1.$$

Alternatively one can define the ellipse in parametric form:

$$x = x_c + a\cos t\cos\alpha - b\sin t\sin\alpha \qquad \text{and} \qquad y = y_c + a\cos t\sin\alpha + b\sin t\cos\alpha,$$

where $0 \le t \le 2\pi$ is an inner parameter on the ellipse. The inner parameter t plays an important role in some fitting algorithms; see [**29, 54, 55**]. The five parameters $(x_c, y_c, a, b, \alpha)^T$ were used by many authors. In earlier publications they were used in conjunction with the inner parameter t on the ellipse; see [**29, 54**]. In later publications, they were used without the inner parameter; see [**7, 5, 4**].

Geometric parameters generally work well, but a problem arises when the ellipse turns into a circle. In this case the angle α becomes a redundant parameter that cannot be defined. Indeed, changing the angle α amounts to rotating the circle around its center, which does not affect it at all. As a result, the Jacobian matrix becomes singular, one of its columns is filled with zeros (see also page 2290 in [**7**]). We provide a proof of this fact in section 3.3.

The singularity of the Jacobian matrix was first noted in [**29**], and for this reason they avoided circles as initial guesses for their iterative ellipse fitting algorithms. On the other hand, the singularity should not really be troublesome here - the Levenberg-Marquardt algorithm works for singular matrices, even the Gauss-Newton step can go through if one applies SVD (see again page 2290 in [**7**]). Perhaps, avoiding circles is just an attempt to "stay on the safe side". Some other authors also impose the restriction $a \neq b$ to avoid the same singularity; see [**52, 53**] and page 2284 in [**7**]. Still others note that the statistical accuracy of the corresponding parameter estimates deteriorates (their variances grow) when the ellipse is close to a circle [**6**]. We also noticed that the angle α tends to change erratically, which may destabilize the performance of the fitting procedure.

3.1.2. Ellipses: Kepler's Parameters. Ellipse can be described by another set of geometric parameters, which involve its foci (x_1, y_1), (x_2, y_2) and the major

semi-axis a. Now the equation of the ellipse is

$$\sqrt{(x-x_1)^2+(y-y_1)^2}+\sqrt{(x-x_2)^2+(y-y_2)^2}=2a, \tag{3.1}$$

based on the fact that the sum of the distances from any point on the ellipse to its foci is constant. There is a natural restriction on the semi-axis: $2a \geq \sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$.

These parameters were used in [**41**] who called (6.11) Kepler's definition of ellipse, so $(x_1, y_1, x_2, y_2, a)^T$ can be called Kepler's parameters.

These parameters also work well, but the peculiar singularity noted above gets even worse now. Let the ellipse be a circle, i.e., let $x_1 = x_2$ and $y_1 = y_2$. Then the Jacobian matrix again becomes singular, and furthermore its rank drops from 5 to 3. Indeed, consider a family of ellipses with foci $(x_1 + s, y_1 + r)$ and $(x_2 - s, y_2 - r)$ for which the fifth parameter a is fixed. When $s = r = 0$, we have the original circle. By elementary geometry, when s and r are small, the ellipse is at distance $\mathcal{O}(s^2 + r^2)$ from the original circle (in the Hausdorff metric). Therefore the derivatives (with respect to s and r) of the distances d_i from the data points to the ellipse turn zero at $s = r = 0$, causing a double-singularity for the Jacobian matrix (the loss of two dimensions in its rank). See a more formal argument in section 3.3.

A particularly bad situation occurs when the ellipse coincides with best fitting circle. Then the Jacobian turns zero completely. In other words, the best fitting circle is a stationary point for the objective function in Kepler's parameters. We provide a proof of this fact in section 3.3. If the iterative procedure starts at the best fitting circle or arrives at it by chance, it will stall instantly and will not progress anywhere.

Actually, many authors use the best fitting circle to initialize their iterative ellipse fitting algorithms; see [**29, 7**]. Some claim that the best circle is the most robust choice for the initial ellipse; see Section 1.1.3 in [**4**]. In Kepler's parameters, this choice would lead to an immediate failure as described above.

3.1.3. Hyperbola: Geometric Parameters. Hyperbolas can be described by five geometric parameters, similar to ellipses, the center coordinates (x_c, y_c), semi-axes a, b and the angle of tilt, α, of the major axis. The hyperbola equation is

$$\frac{\big[(x-x_c)\cos\alpha+(y-y_c)\sin\alpha\big]^2}{a^2}-\frac{\big[-(x-x_c)\sin\alpha+(y-y_c)\cos\alpha\big]^2}{b^2}=1.$$

It is possible to define the hyperbola in parametric form:

$$x = x_c + a\cosh t\cos\alpha - b\sinh t\sin\alpha \qquad \text{and} \qquad y = y_c + a\cosh t\sin\alpha + b\sinh t\cos\alpha,$$

where $-\infty < t < \infty$ is an inner parameter on the hyperbola. These parameters were used in [**7**].

Alternatively, hyperbola can be described by geometric parameters involving its foci: coordinates of both foci (x_1, y_1),(x_2, y_2) and the major semi-axis a. Now the equation of the hyperbola is

$$\left|\sqrt{(x-x_1)^2+(y-y_1)^2}-\sqrt{(x-x_2)^2+(y-y_2)^2}\right|=2a,$$

based on the fact that the difference of the distances from any point on the hyperbola to its foci is constant. We are not aware of any published work using these parameters. In fact very few authors studied the problem of hyperbola fitting, so advantages and disadvantages of the above parametrization schemes are yet to be investigated.

3.1.4. Parabola: Geometric Parameters. Parabolas can be described by four geometric parameters: coordinates of the vertex (x_c, y_c), focus distance p to the directrix and the angle of tilt, α. The parabola equation is

$$\big[-(x-x_c)\sin\alpha+(y-y_c)\cos\alpha\big]^2=2p\big[(x-x_c)\cos\alpha+(y-y_c)\sin\alpha\big].$$

These parameters were used in [**7**].

3.2. Algebraic Parameters of Conics

3.2.1. Quadratic Equation. A *quadratic curve(or conic)* is a figure in the xy plane defined by quadratic equation

$$Ax^2 + 2Bxy + Cy^2 + 2Dx + 2Ey + F = 0, \tag{3.2}$$

where A, B, C, D, E, F are real numbers (*parameters* of the conic). In a more formal way: a conic is a set of points $(x, y) \in \mathbb{R}^2$ satisfying the equation (3.2), with one exclusion which will be discussed later.

Since A, B, C, D, E, F are the coefficients of a quadratic polynomial (i.e., an algebraic expression), they are often called algebraic parameters of the conic.

3.2.2. Parameter Vectors. The six numbers A, B, C, D, E, F can be regarded as components of a vector, $\mathbf{A} = (A, B, C, D, E, F)^T$, which will be called the *vector of parameters*. As usual, we treat all vectors as column vectors, this is why we have to put the transposition sign T.

The parameter space thus will be $\mathbb{R}^6$, the space of all real six-dimensional vectors, with one exclusion (see next). We note that there are many redundancies in this space, so it will be conveniently reduced later.

If all the parameters happen to be equal to zero, i.e., $A = B = C = D = E = F = 0$, then every point (x, y) satisfies (3.2), hence the respective "conic" would be the entire plane $\mathbb{R}^2$. This unwanted "conic" will be excluded, so we will always assume that at least one parameter is different from zero, i.e., we always assume $A^2 + B^2 + C^2 + D^2 + E^2 + F^2 > 0$. This means that the origin $(0, 0, 0, 0, 0, 0)$ must be excluded from the parameter space $\mathbb{R}^6$.

The same conic can be represented by many different quadratic equations (3.2). In particular, if we multiply (3.2) by a non-zero scalar $\alpha \neq 0$, then we get another quadratic equation

$$\alpha Ax^2 + 2\alpha Bxy + \alpha Cy^2 + 2\alpha Dx + 2\alpha Ey + \alpha F = 0 \tag{3.3}$$

that obviously represents the same conic as (3.2). The new equation (3.3) has parameters $\alpha A, \alpha B, \alpha C, \alpha D, \alpha E, \alpha F$, i.e., its parameter vector is $\alpha\mathbf{A}$.

Thus, multiplying the parameter vector $\mathbf{A}$ by a non-zero scalar would not affect the conic, it will remain the same. Given a conic with parameter vector $\mathbf{A} \neq 0$, the entire line in $\mathbb{R}^6$ spanned by $\mathbf{A}$ would represent the same conic. In other words, the magnitude of the parameter vector $\mathbf{A}$ is irrelevant, only its direction matters.

Due to the above considerations, we can assume that every parameter vector has length one:

$$A^2 + B^2 + C^2 + D^2 + E^2 + F^2 = 1. \tag{3.4}$$

In other words, the parameter vector is always assumed to be normalized (its norm is set to one). Thus the reduced parameter space will be the unit sphere in $\mathbb{R}^6$. We will denote it by $\mathbb{S}^5$. Here 5 represents the number of dimensions on it, as the unit sphere in the n-dimensional space $\mathbb{R}^n$ is an $(n-1)$-dimensional hypersurface.

Parameter vectors $\mathbf{A} \in \mathbb{S}^5$ of the unit sphere represent conics *almost* uniquely, i.e., different vectors *usually* correspond to different conics. But there is still a duplicity here, as for any vector $\mathbf{A} \in \mathbb{S}^5$ the opposite vector $-\mathbf{A} \in \mathbb{S}^5$ represents the same conic. So one may further reduce the parameter space $\mathbb{S}^5$ to a half-sphere (a hemisphere), e.g., by requiring $F \geq 0$. Such a further reduction has little advantage but causes unpleasant technical complications, so we will not do that. We will work with the sphere $\mathbb{S}^5$ and simply keep in mind that diametrically opposite points always represent the same conic.

3.2.3. Classification of Quadratic Curves. There are two special points on the unit sphere $\mathbb{S}^5$: the point $(0, 0, 0, 0, 0, 1)$ and its diametrically opposite counterpart $(0, 0, 0, 0, 0, -1)$. They correspond to "quadratic equations" $1 = 0$ and $-1 = 0$, which have no solutions, either real or complex. Some authors call these two parameter vectors "impossible". We call them *poles* (like North Pole and South Pole of the unit sphere). They will play a special role in our considerations.

For every vector $\mathbf{A} \in \mathbb{S}^5$ other than the above two poles, the quadratic equation (3.2) has solutions, either real or complex. If it has real solutions, those make a figure in $\mathbb{R}^2$ that we call a conic. If it has only complex (non-real) solutions, then the corresponding figure in $\mathbb{R}^2$ does not exist (is an empty set), but it has a certain meaning and name in complex coordinates, and we will call it accordingly: *imaginary ellipse*, *imaginary lines*, etc.

A complete classification of quadratic equations and the respective conic types is given below: quadratic equation (3.2) can be written in a vector-matrix form as follows:

$$\begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} A & B & D \\ B & C & E \\ D & E & F \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0,$$

and its quadratic part $Ax^2 + 2Bxy + Cy^2$ can be written in a vector-matrix form as $\begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} A & B \\ B & C \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$. We denote the determinants of the above 3×3 and 2×2 matrices by Δ and J, respectively:

$$\Delta = \det \begin{bmatrix} A & B & D \\ B & C & E \\ D & E & F \end{bmatrix}, \qquad J = \det \begin{bmatrix} A & B \\ B & C \end{bmatrix}.$$

In standard textbooks, Δ is called the "determinant" and J the "discriminant" of the corresponding quadratic form. A few other useful quantities are

$$I = A + C, \qquad Q = A^2 + B^2 + C^2$$

$$K = \det \begin{bmatrix} A & D \\ D & F \end{bmatrix} + \det \begin{bmatrix} C & E \\ E & F \end{bmatrix}$$

Then the types of conics are classified in terms of the above quantities in the following table (see pages 200-201 in [**15**] and Internet article [**62**]):

Remark: circles are special type of ellipses, hence they are included into the "Ellipse" category. Also there are two types of quadratic equations representing

Q	Δ	J	K	$\Delta \cdot I$	Type of conic
> 0	$\neq 0$	> 0		< 0	**Ellipse**
> 0	$\neq 0$	< 0			**Hyperbola**
> 0	$\neq 0$	0			**Parabola**
> 0	0	< 0			**Intersecting lines**
> 0	0	0	< 0		**Parallel lines**
> 0	0	0	0		**Coincident lines**
0			< 0		**Single line**
> 0	0	> 0			**Single point**
> 0	$\neq 0$	> 0		> 0	**Imaginary ellipse**
> 0	0	0	> 0		**Imaginary parallel lines**
0			0		**Poles**

Table 3.1: Types of quadratic curves

single lines: ones are equations with a non-zero quadratic part, such as $x^2 = 0$. We call them *pairs of coincident lines*; others are equations with zero quadratic part, such as $x = 0$. We call them just *single lines*.

3.3. Geometric Parameters versus Algebraic Parameters

Now we have introduced two different parametrization schemes, geometric and algebraic parameters. How do we choose between them in practice and what are their advantages and disadvantages?

First of all, geometric parameters of a conic are standard, simple and have natural geometric meanings. They have been used by many authors in various literatures. Compared to geometric parameters, algebraic parameters do not have any clear geometric significance. We do not know which conic they stand for, what the orientation of the conic is or what the length of the axes are and etc. by just looking at them.

However, when dealing with algebraic parameters, we are not limited to only one type of conic. In other words, we are able to easily switch between different types of conics during the fitting process, which is a huge advantage in practice.

Secondly, geometric parameters such as axes of an ellipse a, b can potentially take arbitrarily large values during the fitting process. If we imagine the space of geometric parameters, we can think of it as a flattened surface of the Earth. When searching for a minimum in such unbounded space, it may result in divergence of the fitting algorithm. On the other hand, algebraic parameters under our normalization (3.4) will never take arbitrary large values. So the space of algebraic parameters under normalization is like a surface of the Earth which is spherical. In particular, when searching for a minimum in this compact parameter space $\mathbb{S}^5$, it can never diverge. Even if we move toward the wrong direction at the beginning, we can still go around the sphere and come back to where we want it to be. So this boundedness enforces the convergence of the iterative fitting algorithm, which makes algebraic parameters a preferred choice for us in practice.

Another interesting characteristic of algebraic parameters is that if we move from one point to another point in the compact algebraic parameter space, then it will cause a very big change in geometric parameters. In other words, a very small change in algebraic parameters will result in a totally different conic.

Lastly, geometric parameters have one unpleasant technical problem when the ellipse turns into a circle, the rotation angle is undefined in this case. This makes the Jacobian matrix singular which is inconvenient in some sense. Recall that if one minimizes $\mathcal{F}(\Theta) = \sum_{i=1}^{n} {d_i}^2$, then the first derivatives of d_i with respect to Θ, i.e., the Jacobian matrix will be needed. We mentioned this fact in section 3.1.1 and 3.1.2. However, algebraic parameters do not have a similar problem. Proofs are provided below.

3.3.1. Proofs of the Singularity of Jacobian Matrix.

THEOREM 3.1. *Let* $\Theta = (x_c, y_c, a, b, \alpha)^T$ *be the vector of standard geometric parameters for an ellipse* $P(x, y; \Theta) = 0$. *If* $a = b$, *i.e., the ellipse is a circle, then the Jacobian matrix becomes singular.*

PROOF. In [GGS1994] Gander et al. proposed a geometric ellipse fitting algorithm in parametric form.

If $\Theta = (x_c, y_c, a, b, \alpha)^T$, an ellipse in parametric form is then given by:

$$\begin{cases} x = x_c + a\cos t\cos\alpha - b\sin t\sin\alpha \\ y = y_c + a\cos t\sin\alpha + b\sin t\cos\alpha \end{cases}$$

or it can be written in matrix form

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_c \\ y_c \end{pmatrix} + Q(\alpha) \begin{pmatrix} a\cos t \\ b\sin t \end{pmatrix}$$

where

$$Q(\alpha) = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix}$$

Given points $(x_i, y_i), i = 1 \cdots n$, one minimizes the sum of the squares of the distances of these points to the fitted ellipse. It is equivalent to solving:

$$P_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix} - \begin{pmatrix} x_c \\ y_c \end{pmatrix} - Q(\alpha) \begin{pmatrix} a\cos t_i \\ b\sin t_i \end{pmatrix} = 0, \quad i = 1, \cdots, n$$

Thus we have $2n$ nonlinear equations for $n{+}5$ unknowns $(t_1, t_2, \cdots, t_n, x_c, y_c, a, b, \alpha)$.

Differentiating P_i with respect to these $n{+}5$ unknowns, one obtains a $(2n){\times}(n{+}5)$ Jacobian matrix:

$$\mathbf{J} = \begin{pmatrix} -as_1 & 0 & \dots & 0 & c_1 & 0 & c & s & -bs_1 \\ 0 & -as_2 & \dots & 0 & c_2 & 0 & c & s & -bs_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & -as_n & c_n & 0 & c & s & -bs_n \\ bc_1 & 0 & \dots & 0 & 0 & s_1 & -s & c & ac_1 \\ 0 & bc_2 & \dots & 0 & 0 & s_2 & -s & c & ac_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & bc_n & 0 & s_m & -s & c & ac_n \end{pmatrix}$$

where $s = \sin\alpha, c = \cos\alpha, s_i = \sin t_i, c_i = \cos t_i, i = 1, \cdots, n$.

If a circle is used to be an initial guess to the iterative fitting algorithm, then the parameters a and b are both equal to the radius of the circle r, and the angle α is not defined. Applying this to the Jacobian matrix, when $a = b$, the last column in Jacobian which denotes the partial derivatives with respect to parameter α becomes linearly dependent on the first n columns. This means that the parameter α is redundant for a circle. Consequently, the Jacobian matrix turns to singular. If we use the standard minimization method such as Gauss-Newton method, then the fitting algorithm will not be able to find a "downhill" direction where the objective function decreases so the iteration will fail. In that case, the standard Gauss-Newton method has to be modified to apply Marquardt steps. □

THEOREM 3.2. *Let $\Theta = (x_1, y_1, x_2, y_2, a)^T$ be the vector of Kepler parameters an ellipse $P(x, y; \Theta) = 0$.*

(i) If $x_1 = x_2$ and $y_1 = y_2$, i.e., the ellipse is a circle, then the Jacobian matrix turns singular.

(ii) Furthermore, if the ellipse coincides with the "best fitting circle", then the Jacobian matrix turns completely zero.

PROOF. (i) Let $\Theta = (x_1, y_1, x_2, y_2, a)^T$ be a Kepler parameter vector, then an ellipse is defined by equation

$$P(x, y; \Theta) = \sqrt{(x - x_1)^2 + (y - y_1)^2} + \sqrt{(x - x_2)^2 + (y - y_2)^2} - 2a = 0 \tag{3.5}$$

It follows that

$$\mathbf{J}(\Theta) = (d_i)_\Theta = ((d_i)_{x_1}, (d_i)_{y_1}, (d_i)_{x_2}(d_i)_{y_2}, (d_i)_a), i = 1 \cdots n \tag{3.6}$$

is the $n \times 5$ Jacobian matrix.

We have proved in [**20**] that

$$(d)_\Theta = \frac{P_\Theta}{\sqrt{P_x^2 + P_y^2}}$$

where P_Θ, P_x, P_y denote the first order partial derivatives of P with respect to Θ, x, y, respectively. Thus to obtain the Jacobian matrix, one needs P_Θ, P_x, P_y.

Differentiating the identity $P(x, y; \Theta) = 0$ gives

$$P_\Theta = (P_{x_1}, P_{y_1}, P_{x_2}, P_{y_2}, P_a)^T = \begin{pmatrix} -\frac{x-x_1}{\sqrt{(x-x_1)^2+(y-y_1)^2}} \\ -\frac{y-y_1}{\sqrt{(x-x_1)^2+(y-y_1)^2}} \\ -\frac{x-x_2}{\sqrt{(x-x_2)^2+(y-y_2)^2}} \\ -\frac{y-y_2}{\sqrt{(x-x_2)^2+(y-y_2)^2}} \\ -2 \end{pmatrix} \tag{3.7}$$

$$P_x = \frac{x - x_1}{\sqrt{(x - x_1)^2 + (y - y_1)^2}} + \frac{x - x_2}{\sqrt{(x - x_2)^2 + (y - y_2)^2}} \tag{3.8}$$

$$P_y = \frac{y - y_1}{\sqrt{(x - x_1)^2 + (y - y_1)^2}} + \frac{y - y_2}{\sqrt{(x - x_2)^2 + (y - y_2)^2}} \tag{3.9}$$

If one uses a circle as the initial guess to the iterative algorithm, then two foci become the same point. In other words, parameters $x_1 = x_2, y_1 = y_2$ and they are

both equal to the center (x_c, y_c) of the circle. Consequently, partial derivatives of P with respect to x_1, y_1 will be equal to partial derivatives of P with respect to x_2, y_2, i.e. $P_{x_1} = P_{x_2}$ and $P_{y_1} = P_{y_2}$. Then in the Jacobian matrix, the first column $(d_i)_{x_1}$ and the third column $(d_i)_{x_2}$ will be the same, the second column $(d_i)_{y_1}$ and the fourth column $(d_i)_{y_2}$ will be the same too. It follows that the Jacobian matrix is again singular; In fact its rank is 3 instead of 5.

(ii) If one uses the best fitting circle as an initial guess, then we have $x_1 = x_2 = x_c, y_1 = y_2 = y_c$ and $a = R$, where(x_c, y_c) is the center of the circle and R is its radius. Since it is the best fitting circle, the objective function $\mathcal{F}(x_c, y_c, R) = \sum_{i=1}^{n} d_i^2$ is minimized, where $d_i = \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} - R$. It follows that

$$\frac{\partial \mathcal{F}}{\partial x_c} = 2\sum_{i=1}^{n} d_i \frac{\partial d_i}{\partial x_c} = 0 \tag{3.10}$$

$$\frac{\partial \mathcal{F}}{\partial y_c} = 2\sum_{i=1}^{n} d_i \frac{\partial d_i}{\partial y_c} = 0 \tag{3.11}$$

$$\frac{\partial \mathcal{F}}{\partial R} = 2\sum_{i=1}^{n} d_i \frac{\partial d_i}{\partial R} = 0 \tag{3.12}$$

(3.13)

$$\mathbf{J}(\Theta) = (d_i)_\Theta = ((d_i)_{x_1}, (d_i)_{y_1}, (d_i)_{x_2}(d_i)_{y_2}, (d_i)_a) = ((d_i)_{x_c}, (d_i)_{y_c}, (d_i)_{x_c}, (d_i)_{y_c}, (d_i)_R)$$

The step of standard minimization method is given by

$$h = -(\mathbf{J}^T\mathbf{J})^{-}\mathbf{J}^T d$$

the vector $\mathbf{J}^T d$ is zero followed by equations (3.10) to (3.13). Thus the step h is zero, which means the iterative algorithm gets stuck and can't move or go anywhere.

□

Recall that the algebraic parameters $\Theta = (A, B, C, D, E, F)^T$ are determined up to a scalar factor, this leaves the Jacobian matrix automatically singular, i.e., rank($\mathbf{J}$) =

5. However, unlike standard geometric and kepler parameters, there is no other singularity occurring for algebraic parameters.

THEOREM 3.3. *Let* $\Theta = (A, B, C, D, E, F)^T$ *be the vector of standard algebraic parameters for a conic* $P(x, y; \Theta) = 0$, (x_i, y_i) , $i = 1, \ldots, n(\geq 5)$ *be the given distinct points. Then for any* Θ, *rank*$(\mathbf{J}) = 5$ *for all* (x_i, y_i)*'s.*

PROOF. We have proved in [**20**] that

$$(d)_\Theta = \frac{P_\Theta}{\sqrt{P_x^2 + P_y^2}}$$

where P_Θ, P_x, P_y denote the first order partial derivatives of P with respect to Θ, x, y and all the derivatives are taken at the projection points.

Differentiating the identity $P(x, y; \Theta) = 0$ gives

$$(P_\Theta) = (P_A, P_B, P_C, P_D, P_E, P_F) \tag{3.14}$$

$$= (x^2, 2xy, y^2, 2x, 2y, 1) \tag{3.15}$$

The denominators $\sqrt{P_x^2 + P_y^2}$ are the same for each point (each row) so that they don't really matter in this case.

By looking at the columns $x^2, 2xy, y^2, 2x, 2y, 1$ in our Jacobian matrix, we found that there is a linear relationship between them. All the x, y's here are the projection points, so they sit on a conic with parameters A, B, C, D, E, F, which means $Ax^2 + 2Bxy + Cy^2 + 2Dx + 2Ey + F = 0$. What's more, this is the only linear relationship they have. For a set of n projection points and $n \geq 5$, there is a unique conic that interpolates them, meaning the projection points will not satisfy any other conics. Hence there are no other linear dependency between these columns. Therefore the rank of the Jacobian matrix is always five. □

3.3.2. Conversion between Algebraic and Geometric Parameters. The conversion formulas from geometric parameters $(x_c, y_c, a, b, \alpha)^T$ for ellipses and hyperbolas to algebraic parameter $(A, B, C, D, E, F)^T$ are straightforward.

$$A = (c/a)^2 + (s/b)^2, \qquad B = cs(1/a^2 - 1/b^2), \qquad C = (s/a)^2 + (c/b)^2,$$

$$D = -Ax_c - By_c, \qquad E = -Cy_c - Bx_c, \qquad F = Ax_c^2 + Cy_c^2 + 2Bx_cy_c - 1.$$

where $s = \sin(\alpha)$ and $c = \cos(\alpha)$.

Conversely, we can also transform algebraic parameters to geometric parameters by translation of the origin and rotation of the coordinate axes. The rotation angle is easy to find

$$\alpha = \frac{\arctan\frac{2B}{A-C}}{2},$$

After rotating the conic, the center in the new coordinate system becomes

$$\begin{pmatrix} {x_c}' \\ {y_c}' \end{pmatrix} = \begin{pmatrix} -\frac{N_1}{M_{11}} \\ -\frac{N_2}{M_{22}} \end{pmatrix}$$

where

$$Q = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix}, \qquad M = Q \begin{pmatrix} A & B \\ B & C \end{pmatrix} Q^T, \qquad N = Q \begin{pmatrix} D \\ E \end{pmatrix}$$

Then two axes are

$$a = \sqrt{\left|\frac{O}{M_{11}}\right|}, \qquad b = \sqrt{\left|\frac{O}{M_{22}}\right|}$$

where

$$O = F - {x'_c}^2 M_{11} - {y'_c}^2 M_{22}.$$

Lastly, we need to rotate the center back to the original coordinate system

$$(x_c, y_c) = Q^T(x'_c, y'_c).$$

CHAPTER 4

Objective Function for Quadratic Curves

In this chapter, we discuss some basic properties of unit sphere $\mathbb{S}^5$, which plays the role of the parameter space for quadratic curves and investigate the general behavior of the objective function on the sphere.

4.1. Open Domains in Parameter Space

We begin with the description of the unit sphere $\mathbb{S}^5$. First let us recall the main types of conics here, grouped according to the dimensionality of the corresponding regions in $\mathbb{S}^5$.

Dimension = 5	Dimension = 4	Dimension = 3 or 2
Ellipse (E)	**Parabola** (P)	**Parallel lines**(PL)
Hyperbola (H)	**Single point**(SP)	**Imaginary parallel lines** (IPL)
Imaginary ellipse (IE)	**Intersecting lines** (IL)	**Coincident lines** (CL)
		Single line (SL)

Table 4.1: Main types of quadratic curves grouped according to the dimensionality of the corresponding regions in $\mathbb{S}^5$

4.1.1. Partition into Domains. Each parameter vector $\mathbf{A} \in \mathbb{S}^5$ corresponds to a conic of a certain type. Accordingly, the unit sphere $\mathbb{S}^5$ (the parameter space) is divided into 10 domains corresponding to each conic types, plus two extra points, the poles:

$$\mathbb{S}^5 = \mathbb{D}_{\mathrm{E}} \cup \mathbb{D}_{\mathrm{H}} \cup \mathbb{D}_{\mathrm{IE}} \cup \mathbb{D}_{\mathrm{P}} \cup \mathbb{D}_{\mathrm{SP}} \cup \mathbb{D}_{\mathrm{IL}} \cup \mathbb{D}_{\mathrm{PL}} \cup \mathbb{D}_{\mathrm{IPL}} \cup \mathbb{D}_{\mathrm{CL}} \cup \mathbb{D}_{\mathrm{SL}} \cup \{\mathbf{P}_1\} \cup \{\mathbf{P}_{-1}\},$$

where the domains are coded by the names of the conic types, as shown in the above table.

As we know, larger and more complex domains have higher dimensionality. The maximal number of dimensions here is five, as the entire sphere $\mathbb{S}^5$ is five-dimensional. Three five-dimensional domains $\mathbb{D}_{\mathrm{E}}$, $\mathbb{D}_{\mathrm{H}}$, and $\mathbb{D}_{\mathrm{IE}}$ are most important; they are "massive", they occupy a substantial part of the sphere $\mathbb{S}^5$. In fact, they are *open domains*, in the topological sense. Their openness follows from the classification table given in section 3.2.3: they are all defined by inequality constraints, such as $Q > 0$, $\Delta \neq 0$, $J > 0$, etc. We note that if an inequality holds at any point $\mathbf{A} \in \mathbb{S}^5$, it will still hold in a small vicinity of that point, as our functions Q, Δ, J, etc., change continuously on the sphere $\mathbb{S}^5$. This simple observation proves the openness of the domains $\mathbb{D}_{\mathrm{E}}$, $\mathbb{D}_{\mathrm{H}}$, and $\mathbb{D}_{\mathrm{IE}}$.

4.1.2. Volumes of Open Domains. The "size" or "mass" of any domain in the sphere $\mathbb{S}^5$ can be measured by its volume (more precisely, five-dimensional volume, or Lebesgue measure). Five-dimensional domain have positive ("appreciable") volume. All the other domains in our partition of $\mathbb{S}^5$ have volume zero: they are too tiny, "invisible", the volume does not recognize their existence.

The volume of our domains can be estimated by an easy Monte Carlo experiment. We just generate random points on the unit sphere $\mathbb{S}^5$ and estimate the volume of each domain by the percentage of points falling into it. Below is our experimental estimates:

Domain	$\mathbb{D}_{\mathrm{E}}$	$\mathbb{D}_{\mathrm{H}}$	$\mathbb{D}_{\mathrm{IE}}$
Volume (%)	20.7	76.8	2.5

Table 4.2: Volumes of open domains on the unit sphere $\mathbb{S}^5$

The volume of each domain is given as a percentage of the total volume of $\mathbb{S}^5$. The volume of all the other domains is zero. We see that hyperbolas occupy more

than 3/4 of the parameter space. The domination of hyperbolas over ellipses is also noted in sections 2.3 and 2.4. We also note that"imaginary ellipse" occupy just a small fraction of the sphere $\mathbb{S}^5$, only 2.5.

Remark: Some authors write the quadratic equation (A.2) in a slightly different form:

$$Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0, \tag{4.1}$$

And under this parameter scheme, the estimate volume for each open domain is the following

Domain	$\mathbb{D}_{\mathrm{E}}$	$\mathbb{D}_{\mathrm{H}}$	$\mathbb{D}_{\mathrm{IE}}$
Volume (%)	26.5	65	8.5

As we can see, we have a much higher volume (8.5%) for the open domain of imaginary ellipse which in practice we consider as empty set or undefined. By adding coefficient 2 in our parameter scheme, we reduce the domain of points that represents invalid solutions for our model to 2.5%. When running an iterative numerical algorithm searching for the minimum of the objective, every time iterations run into such domain of empty set and return a empty solution, they have to retreat and readjust their step. Due to this, the smaller the size of $\mathbb{D}_{\mathrm{IE}}$, the better. That's why the algebraic parameter scheme we adopted is better than this scheme.

4.1.3. Connected Components of Open Domains. In topological terms, open domains may or may not be connected. If the domain is not connected, it consists of connected components. Interestingly, our open domains $\mathbb{D}_{\mathrm{E}}$, $\mathbb{D}_{\mathrm{H}}$, and $\mathbb{D}_{\mathrm{IE}}$ are <u>not</u> connected, each of them consists of exactly two connected components. This is related to the choice of sign, see below.

Indeed, each parameter vector $\mathbf{A} = (A, B, C, D, E, F)^T \in \mathbb{S}^5$ not only specifies a conic, but defines a quadratic function

$$Q(x, y) = Ax^2 + 2Bxy + Cy^2 + 2Dx + 2Ey + F$$

on the xy plane. Note that $Q(x,y)=0$ *on the conic*, but $Q(x,y)>0$ or $Q(x,y)<0$ elsewhere. If the conic is an ellipse, for example, we may have $Q>0$ inside the ellipse and $Q<0$ outside of it, or vice versa. Thus we have two types of parameter vectors $\mathbf{A}\in\mathbb{D}_{\mathrm{E}}$: for one Q is positive inside the ellipse and for the other Q is negative inside the ellipse. This dichotomy causes $\mathbb{D}_{\mathrm{E}}$ to consist of two pieces, we denote them by $\mathbb{D}_{\mathrm{E}}^{+}$ and $\mathbb{D}_{\mathrm{E}}^{-}$, depending on whether $Q>0$ or $Q<0$ inside the ellipse (i.e., at its center). Similarly, we have the respective partition $\mathbb{D}_{\mathrm{H}}=\mathbb{D}_{\mathrm{H}}^{+}\cup\mathbb{D}_{\mathrm{H}}^{-}$ for hyperbola too. Lastly, if $\mathbf{A}\in\mathbb{D}_{\mathrm{IE}}$, then the corresponding quadratic function $Q(x,y)$ cannot take zero values, so it is either entirely positive or entirely negative. Again this causes a natural partition of $\mathbb{D}_{\mathrm{IE}}$ into two pieces, $\mathbb{D}_{\mathrm{IE}}^{+}$ and $\mathbb{D}_{\mathrm{IE}}^{-}$.

Note that if $\mathbf{A}\in\mathbb{D}_{\mathrm{E}}^{+}$, then $-\mathbf{A}\in\mathbb{D}_{\mathrm{E}}^{-}$ and vice versa. Thus the subdomains $\mathbb{D}_{\mathrm{E}}^{+}$ and $\mathbb{D}_{\mathrm{E}}^{-}$ are diametrically opposite to each other on the sphere $\mathbb{S}^5$. In a sense, they are "mirror images" of each other; they have identical shapes and equal volumes. The same is true for the two parts of $\mathbb{D}_{\mathrm{H}}$ and the two parts of $\mathbb{D}_{\mathrm{IE}}$.

4.1.4. Boundaries of Open Domains. We've discussed three principal open domains $\mathbb{D}_{\mathrm{E}}$, $\mathbb{D}_{\mathrm{H}}$, and $\mathbb{D}_{\mathrm{IE}}$. And we know that they are five dimensional and they cover 100% of the sphere $\mathbb{S}^5$, in terms of volume. How about other domains? The other domains have lower dimensionality and zero volume. They, in a sense, make pieces of the boundaries of the principal domains $\mathbb{D}_{\mathrm{E}}$, $\mathbb{D}_{\mathrm{H}}$, and $\mathbb{D}_{\mathrm{IE}}$.

Most important of those "boundary pieces" are the four-dimensional domains: $\mathbb{D}_{\mathrm{P}}$ (parabolas), $\mathbb{D}_{\mathrm{SP}}$ (single points), and $\mathbb{D}_{\mathrm{IL}}$ (intersecting lines). They separate our open domains or their components from each other. As a matter of fact, the hypersurface $\mathbb{D}_{\mathrm{SP}}$ separates the open domain $\mathbb{D}_{\mathrm{E}}$ of ellipses from the open domain $\mathbb{D}_{\mathrm{IE}}$ of imaginary ellipses, the hypersurface $\mathbb{D}_{\mathrm{IL}}$ separates the two components $\mathbb{D}_{\mathrm{H}}^{+}$ and $\mathbb{D}_{\mathrm{H}}^{-}$ of the open domain $\mathbb{D}_{\mathrm{H}}$ of hyperbolas from each other and the hypersurface $\mathbb{D}_{\mathrm{P}}$ separates the domain $\mathbb{D}_{\mathrm{H}}$ of hyperbolas from the domain $\mathbb{D}_{\mathrm{E}}$ of ellipses. The following schematic diagram illustrats the structure of the parameter space, with all principal subdomains and the respective separating hypersurfaces.

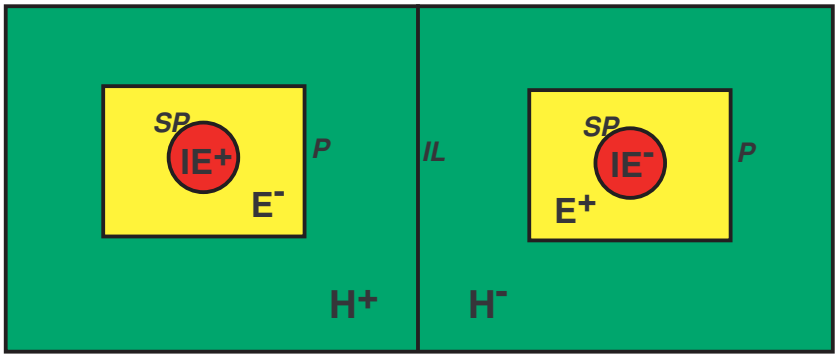

Figure 4.1: Principal domains and separating hypersurfaces

Note: The labels correspond to our notation in the text: H^+ means $\mathbb{D}^+_{\mathrm{H}}$, etc.

4.2. Objective Function on the Sphere

Now we move on to the study of the objective function on the parameter space of quadratic curves, i.e.,on the sphere $\mathbb{S}^5$. Recall that given some points $P_1, \ldots, P_n \in \mathbb{R}^2$, the objective function is the sum of squares of the distances to a model object (in our case, conic) S:

$$\mathcal{F}(S) = \sum_{i=1}^{n} \big[\mathrm{dist}(P_i, S)\big]^2, \tag{4.2}$$

The objective function depends on the points $P_1, \ldots, P_n$, but in practical settings those are fixed, so the only variable is S, which is regarded as the sole argument of $\mathcal{F}$.

If $\mathbf{A}$ is a vector of algebraic parameters describing the conic S, then $\mathcal{F}$ naturally becomes a function of $\mathbf{A}$, i.e., $\mathcal{F}$ becomes a function on the parameter space. More precisely, $\mathcal{F}$ is defined on all parameter vectors $\mathbf{A} \in \mathbb{S}^5$ corresponding to real conics. It is not defined for parameters corresponding to imaginary conics or poles. We denote the domain of the objective function by

$$\mathfrak{D}_{\mathcal{F}} = \mathbb{D}_{\mathrm{E}} \cup \mathbb{D}_{\mathrm{H}} \cup \mathbb{D}_{\mathrm{P}} \cup \mathbb{D}_{\mathrm{SP}} \cup \mathbb{D}_{\mathrm{IL}} \cup \mathbb{D}_{\mathrm{PL}} \cup \mathbb{D}_{\mathrm{CL}} \cup \mathbb{D}_{\mathrm{SL}}.$$

We note that $\mathfrak{D}_{\mathcal{F}}$ does not include regions $\mathbb{D}_{\mathrm{IE}}$ and D_{IPL} corresponding to imaginary conics or the poles $\mathbf{P}_{\pm 1}$.

We will describe some general behaviors of the objective function $\mathcal{F}$ on such domain $\mathfrak{D}_{\mathcal{F}}$ in the following.

4.2.1. Continuity of the Objective Function. One can prove that $\mathcal{F}$ is continuous on the entire domain $\mathfrak{D}_{\mathcal{F}}$ except a tiny subregion $\mathbb{D}_{\mathrm{CL}}$ corresponding to coincident lines. On the latter region $\mathcal{F}$ is badly discontinuous, but it is lower semi-continuous. Theorems will be presented here and detailed proofs are available in [**33**].

THEOREM 4.1. *The objective function $\mathcal{F}$ is continuous everywhere on its domain $\mathfrak{D}_{\mathcal{F}}$ except on the region $\mathbb{D}_{\mathrm{CL}}$ corresponding to coincident lines.*

THEOREM 4.2. *The objective function $\mathcal{F}$ is lower semi-continuous on the region $\mathbb{D}_{\mathrm{CL}}$ corresponding to coincident lines.*

THEOREM 4.3. *The objective function $\mathcal{F}$ grows to infinity near the region $\mathbb{D}_{\mathrm{IPL}}$ and near the poles $\mathbf{P}_{\pm 1}$. More precisely, if $\mathbf{A}_n \to \mathbf{A}$ and either $\mathbf{A} \in \mathbb{D}_{\mathrm{IPL}}$ or $\mathbf{A} = \mathbf{P}_{\pm 1}$, then $\mathcal{F}(\mathbf{A}_n) \to \infty$.*

The above theorems easily imply the existence of a global minimum of $\mathcal{F}$ as follows. The domain $\mathfrak{D}_{\mathcal{F}}$ is not compact, but due to Theorem 4.3 we can cut out and ignore a small vicinity of the region $\mathbb{D}_{\mathrm{IPL}}$ and the poles $\mathbf{P}_{\pm 1}$ where the function is too big. Then the remaining part of the domain $\mathfrak{D}_{\mathcal{F}}$ will be compact. And now the lower semi-continuity of $\mathcal{F}$ guarantees the existence of its global minimum. Indeed, any lower semi-continuous function on a compact domain attains its minimum. The existence of a global minimum is nothing new, however, as we have proved the existence of the best fitting object already in section 2.1.

4.2.2. Differentiability of the Objective Function. Most popular minimization algorithms (such as the steepest descent, Newton-Raphson, Gauss-Newton, or

Levenberg-Marquardt) use derivatives of the given function. Thus it is essential that $\mathcal{F}$ is differentiable. As $\mathcal{F}$ is the sum of squares of the distances, see (2.10), it is really enough to check if $[\mathrm{dist}(P,S)]^2$, i.e., the square of the distance from any given point $P=(x_0,y_0)\in\mathbb{R}^2$ to a conic S, is differentiable with respect to the conic's parameters. Next,

$$[\mathrm{dist}(P,S)]^2=[\mathrm{dist}(P,Q)]^2=(x-x_0)^2+(y-y_0)^2,$$

where $Q=(x,y)$ is the projection of P onto the conic S. Thus again, it is enough to check that the coordinates x,y of the footpoint Q of the projection are differentiable with respect to the conic's parameters.

We note that the point Q on the conic S closest to the given point P may not be uniquely defined. For example, if S is a circle and P is its center, then all the points of S are equally distant from P, hence the point Q cannot be chosen anywhere on the circle. In such exceptional situations one can hardly expect that the coordinates of Q would be differentiable. But other than that, one can prove the differentiability of the objective function.

THEOREM 4.4. [**33**] *If the point Q on the conic S closest to the given point P is unique and if P is not at the center of curvature of the conic S at the point Q, then the coordinates x and y of the point Q are differentiable with respect to the conic's parameters.*

The above theorem does not depend on the type of the conic, so the change of the conic type does not affect the differentiability of $\mathcal{F}$. For example, if the given parameter vector $\mathbf{A}$ corresponds to a parabola, so that nearby parameter vectors correspond to ellipses or hyperbolas, then the objective function is still differentiable at $\mathbf{A}$.

In other words, the objective function is not only continuous on the borders between our main subdomains $\mathbb{D}_{\mathrm{E}}^{\pm}$ and $\mathbb{D}_{\mathrm{H}}^{\pm}$, but it changes smoothly from one subdomain to another. If we could "walk" on the graph of $\mathcal{F}$, or "feel" it with a hand, we would

never notice that it consisted of different parts corresponding to ellipses and hyperbolas with positive or negative centers (separated by tiny borders corresponding to parabolas or intersecting lines). The entire graph would appear as one smooth piece.

4.2.3. Illustration. Let us consider a simplified family of conics defined by

$$x^2 + Cy^2 + 2Dx + 1 = 0 \tag{4.3}$$

where only two algebraic parameters, C and D, are variables and all the others are fixed ($A = 1$, $B = E = 0$, and $F = 1$). We easily see that

$$\Delta = \begin{vmatrix} 1 & 0 & D \\ 0 & C & 0 \\ D & 0 & 1 \end{vmatrix} = C(1 - D^2), \qquad J = \begin{vmatrix} 1 & 0 \\ 0 & C \end{vmatrix} = C, \qquad I = 1 + C, \qquad Q = 1 + C^2$$

$$K = \begin{vmatrix} 1 & D \\ D & 1 \end{vmatrix} + \begin{vmatrix} C & 0 \\ 0 & 1 \end{vmatrix} = 1 - D^2 + C.$$

According to standard rules, the conic (4.3) may be of the following types:

- **Hyperbola**, whenever $C < 0$ and $|D| \neq 1$.
- **Ellipse**, whenever $C > 0$ and $|D| > 1$.
- **Imaginary ellipse**, whenever $C > 0$ and $|D| < 1$.
- **Single point**, whenever $C > 0$ and $D = \pm 1$.
- **Intersecting lines**, whenever $C < 0$ and $D = \pm 1$.
- **Parallel lines**, whenever $C = 0$ and $|D| > 1$.
- **Imaginary parallel lines**, whenever $C = 0$ and $|D| < 1$.
- **Coincident lines**, whenever $C = 0$ and $D = \pm 1$.

The diagram 4.2 shows the types of the conic (4.3) on the CD plane:

Next we chose five data points $P_1, \ldots, P_5$ lying on the ellipse $x^2 + 6y^2 - 12x + 1 = 0$ and computed the objective function $\mathcal{F}$ for all $-10 \leq C, D \leq 10$. Fig. 4.3 is the graph of $\mathcal{F}$, as a function of C and D, plotted by MATLAB.

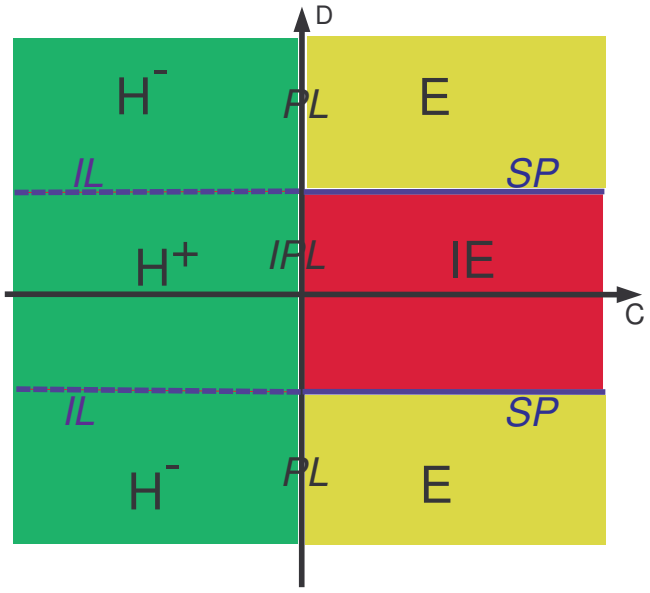

Figure 4.2: Illustration diagram

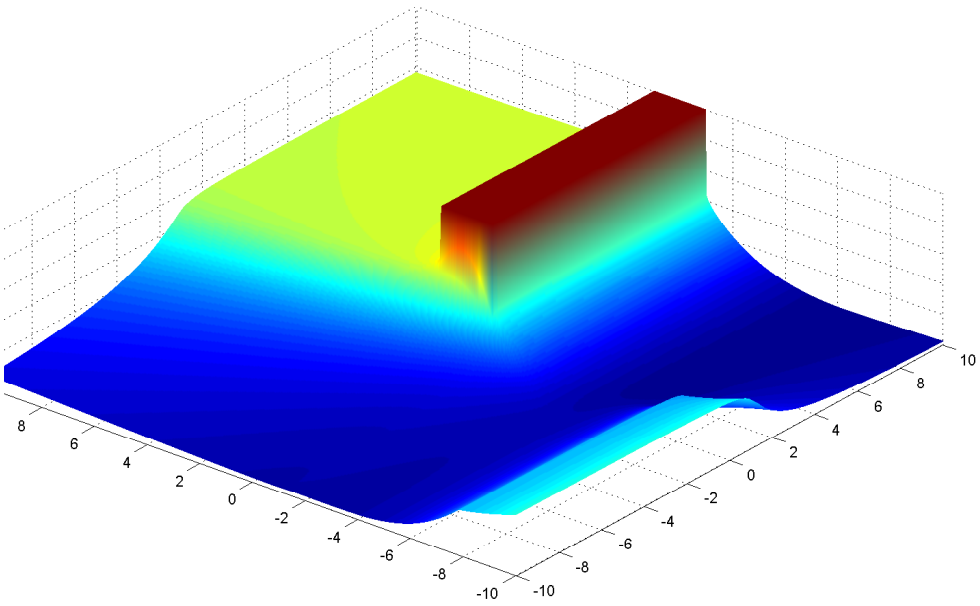

Figure 4.3: Illustration of differentiability of the objective function

The dark red part of the graph lies above the domain of imaginary ellipses $\mathbb{D}_{\mathrm{IE}}$ where the objective function cannot be defined. Other than that, the entire graph appears to be one smooth and "glassy" surface. In particular, we clearly see that the objective function is not broken or even wrinkled at places where the conic type changes.

The lowest (darkest) point of the graph is at $C = 6,\ D = -6$, where the objective function achieves its global minimum $\mathcal{F} = 0$ (corresponding to the ellipse $x^2 + 6y^2 - 12x + 1 = 0$ that passes through all our five data points).

4.3. Objective Function near Boundaries

In the previous section we described some general properties of the objective function $\mathcal{F}$ on its natural domain $\mathfrak{D}_{\mathcal{F}} \subset \mathbb{S}^5$ focusing on its continuity and differentiability.

Now one might ask how does the objective function behave near other boundaries?

4.3.1. Objective Function near the Hypersurface of Single Points. The objective function is not defined for imaginary conics, in particular it is not defined on the red disks $\mathbb{D}^{\pm}_{\mathrm{IE}}$ (shown as $\mathrm{IE}^{\pm}$). The objective function is defined on the subdomains $\mathbb{D}^{\pm}_{\mathrm{E}}$ and the bordering hypersurface $\mathbb{D}_{\mathrm{SP}}$, but then it stops; it does not extend into $\mathbb{D}^{\pm}_{\mathrm{IE}}$. The graph of $\mathcal{F}$ is smooth over $\mathbb{D}^{\pm}_{\mathrm{E}}$ but then it abruptly terminates. Could this cause trouble for our main purpose - the minimization of $\mathcal{F}$?

The answer is No. First of all, the objective function $\mathcal{F}$ actually grows near the hypersurface $\mathbb{D}_{\mathrm{SP}}$. Recall that the parameter vectors $\mathbf{A} \in \mathbb{D}_{\mathrm{E}}$ that are near $\mathbb{D}_{\mathrm{SP}}$ correspond to very small ellipses. As $\mathbf{A}$ gets closer and closer to $\mathbb{D}_{\mathrm{SP}}$, the corresponding ellipse shrinks and collapses to a single points. Obviously, this does not help to fit the data points any better. Making the ellipse smaller and smaller only increases the distances from that ellipse to all the data points located outside the ellipse, thus increasing the value of the objective function.

Secondly, because of such growth, the minimization procedure will not move in the direction of $\mathbb{D}_{\mathrm{SP}}$. All decent minimization algorithms (Levenberg-Marquartd,

Trust Region, etc.) keep moving only as long as the value of the objective function decreases at each iteration. If the objective function does not decrease, then the algorithm retreats, its step is recalculated, and this recalculation is repeated until the algorithm finds a place where the objective function decreases. If the algorithm comes close to the hypersurface $\mathbb{D}_{\mathrm{SP}}$, then it will have to turn around and move away from $\mathbb{D}_{\mathrm{SP}}$ just in order to find smaller values of the objective function.

4.3.2. Objective Function near the Domain of Coincident Lines. The objective function $\mathcal{F}$ becomes highly irregular and badly discontinuous in the vicinity of the domain $\mathbb{D}_{\mathrm{CL}}$ corresponding to coincident lines.

Again the same argument as above shows that the objective function tends to grow near the domain $\mathbb{D}_{\mathrm{CL}}$, which discourages minimization algorithms from approaching this domain. Indeed, recall that if a parameter vector $\mathbf{A}$ is close to the domain $\mathbb{D}_{\mathrm{CL}}$, i.e., $\mathbf{A} \approx \mathbf{A}_0 \in \mathbb{D}_{\mathrm{CL}}$, then the conic S corresponding to $\mathbf{A}$ is close to the line L_0 corresponding to $\mathbf{A}_0$. More precisely, S wholly lies in a narrow strip around L_0. As $\mathbf{A} \to \mathbf{A}_0$, the entire conic S gets closer and closer to L_0, though it may not stretch all the way along L_0.

4.3.3. Objective Function near the Domain of Single Lines. The above argument basically shows that a single line or any object stretching along a single line cannot provide a good fit for typical sets of data points. Many other conics achieve a better fit, including parallel lines, intersecting lines, long ellipses that are close to two parallel lines, or similar hyperbolas, etc. Respectively, the objective function $\mathcal{F}$ tends to decrease if the parameter vector moves away from the domain $\mathbb{D}_{\mathrm{SL}}$ corresponding to single lines. Thus the minimization algorithms are not likely to move toward this domain, i.e., it should not bother us.

4.3.4. Objective Function near the Domain of Parallel Lines. On the contrary, a pair of parallel lines may provide quite a good fit for some sets of data points. It is not apparent at all that ellipses or hyperbolas would provide a better

fit. In fact it is possible that the best fit is achieved only by a pair of parallel lines. Respectively, we believe the minimization algorithms are quite likely to approach the domain $\mathbb{D}_{\mathrm{PL}}$ or wander around in its vicinity. Hence this domain is essential.

4.3.5. Summary. To summarize, we list all the domains where the minimization algorithms are likely to maneuver searching for the best fitting conic and where the best fit can be found: Ellipses $\mathbb{D}_{\mathrm{E}}$, hyperbolas $\mathbb{D}_{\mathrm{H}}$, parabolas $\mathbb{D}_{\mathrm{P}}$, intersecting lines $\mathbb{D}_{\mathrm{IL}}$ and parallel lines $\mathbb{D}_{\mathrm{PL}}$. To formalize this idea we prove the following (see [**33**]):

THEOREM 4.5. *For any set of data points $P_1, \dots, P_n$ the global minimum of the objective function $\mathcal{F}$ belongs to the union $\mathbb{D}_{\mathrm{E}} \cup \mathbb{D}_{\mathrm{H}} \cup \mathbb{D}_{\mathrm{P}} \cup \mathbb{D}_{\mathrm{IL}} \cup \mathbb{D}_{\mathrm{PL}}$. If the objective function $\mathcal{F}$ has multiple global minima, then at least one of them belongs to the above union. This union cannot be shortened, i.e., for any conic S in this union of domains there exists a data set for which S provides the unique best fit.*

This theorem basically says that all the other parts of the parameter space $\mathbb{S}^5$ can be ignored for the purpose of minimization of the objective function. On those parts $\mathcal{F}$ is either not defined or tends to grow.

4.4. Local Minima

In the previous sections we showed that $\mathcal{F}$ is continuous and differentiable (more precisely, has a continuous first derivative) everywhere except certain bad places in $\mathbb{S}^5$ which we carefully identified. At all those places the function $\mathcal{F}$ either has "peak" (local maxima) or somehow tends to grow. Since our main goal is minimization of $\mathcal{F}$, i.e., finding its (local) minima, those bad places should not really harm the minimization procedure. Standard minimization algorithms, such as Gauss-Newton and Levenberg-Marquardt, can only go where the objective function decreases, so they are bound to move away from bad places. Since the parameter space is compact, they cannot move off toward infinity (i.e., diverge). Therefore they are bound to converge to a local minimum of $\mathcal{F}$.

Ideally, the minimization procedures should converge to the *global minimum* of $\mathcal{F}$ and not be distracted by its *local minima*. Here we investigate the local minima of $\mathcal{F}$ and assess their potentially distractive role.

4.4.1. No Local Minima for $n = 5$. In the simplest case of $n = 5$ data points there is always an interpolating conic, i.e., a conic passing through all the five points. If the latter are in general linear position (which means that no three points are collinear), the interpolating conic is unique and non-degenerate (i.e., it is an ellipse, a parabola, or a hyperbola); see (ref).

Thus if $n = 5$, the objective function takes its global minimum $\mathcal{F} = 0$. And, quite surprisingly, it has "no local minima"! This is a mathematical fact that we proved in section 2.3.2. Since there are no local minima, we proceed to $n > 5$.

4.4.2. Numerical Tests for Local Minima: Data Points without Noise. For $n > 5$ data points, local minima are possible. They occur even if the points are observed without noise, i.e., if all the n points lie on a conic. We have investigated local minima for data points placed on an ellipse. And we note that local minima of the objective function $\mathcal{F}$ can only be found numerically. We have run an extensive computer experiment to locate all the local minima of $\mathcal{F}$.

First let me describe our numerical tests whose results were summarized in next section. Our task is to find all local minima of the objective function $\mathcal{F}$ for a given set of $n > 5$ points. We employ the Levenberge-Marquard minimization algorithm to find minima of the objective function $\mathcal{F}$. Given a set of $n > 5$ points, we generate 1000 randomly chosen initial guesses (see below) and then we use each one to initialize the Levenberge-Marquard procedure. We run the latter until it converges to a limit conic. Thus we get 1000 limit conics, which can be potentially local minima of the objective function.

For each computed limit conic we verify that it is indeed a local minimum of $\mathcal{F}$ by checking that its gradient vanishes (i.e., $\nabla\mathcal{F} = 0$) and its Hessian is positive

semidefinite (i.e., $\mathbf{H} = \nabla\nabla\mathcal{F} \geq 0$). The limit conics that fail to satisfy these conditions are discarded. The formulas for $\nabla\mathcal{F}$ and $\nabla\nabla\mathcal{F}$ are given in section [Appendix?].

Then we eliminated repeating limit conics. This is a delicate step as the accuracy of the computed local minimum depends on the eigenvalues of the Hessian matrix: small (positive) eigenvalues indicate that the local minimum is at the bottom of a valley, where the precision of locating the local minimum is poor. We applied the following rule. If $\mathbf{A}$ and $\mathbf{A}'$ are the parameter vectors corresponding to two local minima and $\mathbf{H}$ and $\mathbf{H}'$ are the respective Hessian matrices, then those local minima are treated as coincident (repeating) if $(\mathbf{A} - \mathbf{A}')^T\mathbf{H}(\mathbf{A} - \mathbf{A}') < \varepsilon$ and the same holds for $\mathbf{H}'$ (here ε is a small threshold).

In the end we get a list of distinct local minima of $\mathcal{F}$. The one with the smallest value of $\mathcal{F}$ is its global minimum, all the others are proper local minima.

The Choice of Initial Guesses. Initial guesses are generated as follows. Let $R = [x_1, x_2] \times [y_1, y_2] \in \mathbb{R}^2$ be a rectangle that contains all our $n > 5$ points (the construction of R is described below). We generate 1000 random five-point sets $\{Q_1, \ldots Q_5\}$ with uniform distribution in the rectangle R. (In other words, we select five x-coordinates in the interval $[x_1, x_2]$ randomly and five y-coordinates in the interval $[y_1, y_2]$ randomly and combine them to get five pairs of xy coordinates that we treat as five points in R.) For each five-point set $\{Q_1, \ldots Q_5\}$ in R we find the unique conic that interpolates those five points by solving the corresponding system of linear equations. Then we use that conic as an initial guess.

Rational for our choice of initial guesses and construction of the rectangle R are the same as described in section 2.4.1.

4.4.3. Local Minima for $n > 5$ Points without Noise. In our tests we used the ellipse with semiaxes $a = 2$ and $b = 1$ and placed n points (equally spaced) along the entire ellipse or along a certain arc of the ellipse. This is done by using an internal angular parameter $\varphi \in [-\pi, \pi]$. We choose φ_{ini} and φ_{end} (representing the endpoints

of the arc) and set

$$x_i = a\cos\varphi_i, \qquad y_i = b\sin\varphi_i, \qquad \varphi_i = \varphi_{\text{ini}} + (\varphi_{\text{end}} - \varphi_{\text{ini}})((i-0.5)/n), \quad 1 \le i \le n.$$

We observed that when the data points are placed along the entire ellipse, i.e., $\varphi_{\text{ini}} = -\pi$ and $\varphi_{\text{end}} = \pi$, the objective function had no local minima. The same is true when the data points are placed along the right half of the ellipse, i.e., $\varphi_{\text{ini}} = -\pi/2$ and $\varphi_{\text{end}} = \pi/2$. But when the data points are placed along the upper half of the ellipse, i.e., $\varphi_{\text{ini}} = 0$ and $\varphi_{\text{end}} = \pi$, the objective function does have local minima, see below.

Examples of Local Minima: Upper Half of Ellipse, $n = 6$. The figure below shows $n = 6$ data points placed along the upper half of the ellipse $x^4/4 + y^2 = 1$. The global minimum of $\mathcal{F}$ is achieved by the interpolating ellipse (red), and two local minima of $\mathcal{F}$ correspond to two hyperbolas (blue and green). The first column of the numerical output gives the values of $\mathcal{F}$ at all these three minima. The two hyperbolas are obviously symmetric to each other with respect to the y axis, this is why $\mathcal{F}$ takes the same value at each. The second column (PER) gives the percentages of random initial guesses from which the minimization routine converged to each minimum.

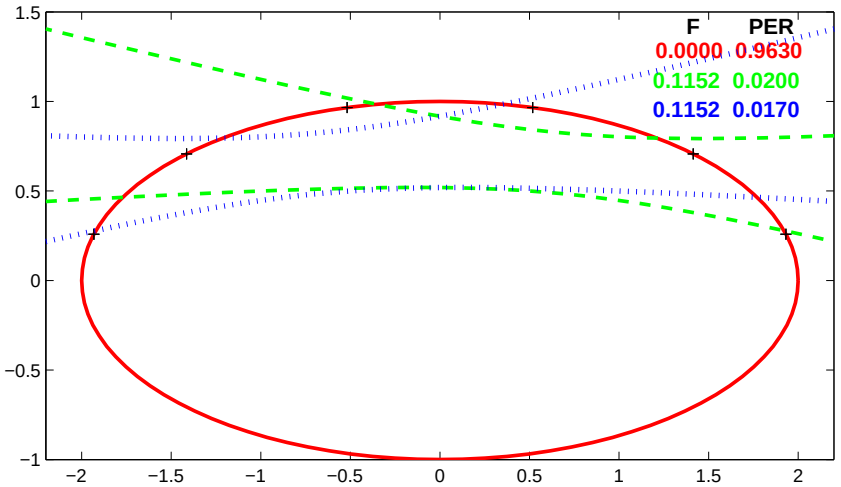

Figure 4.4: Examples of local minima when $n = 6$ points are placed along upper half of an ellipse

Note that from an overwhelming majority (96%) of randomly generated initial conics the minimization procedure converged to the global minimum. We explain this phenomenon below.

Examples of Local Minima: Upper Half of Ellipse, $n = 8$. For $n = 8$ points placed along the upper half of the same ellipse the function $\mathcal{F}$ has one global minimum (the interpolating ellipse) and seven (!) local minima, which include two ellipses and five hyperbolas. They are all shown in the next set of figures. Each figure presents the interpolating ellipse (red) with the data points on it and some conics corresponding to local minima.

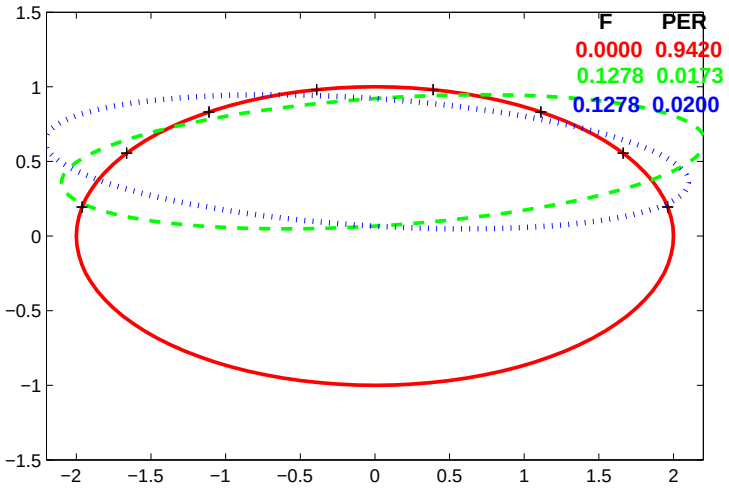

Figure 4.5: Examples of local minima when $n = 8$ points are placed along upper half of an ellipse

We note again that from an overwhelming majority (94%) of randomly generated initial conics the minimization procedure converged to the global minimum. Only 6% of random initial conics fell into the vicinities of local minima.

Examples of Local Minima: Quarter of the Ellipse, $n = 6$. In the last experiment, we placed $n = 6$ data points along a quarter of the same ellipse (we

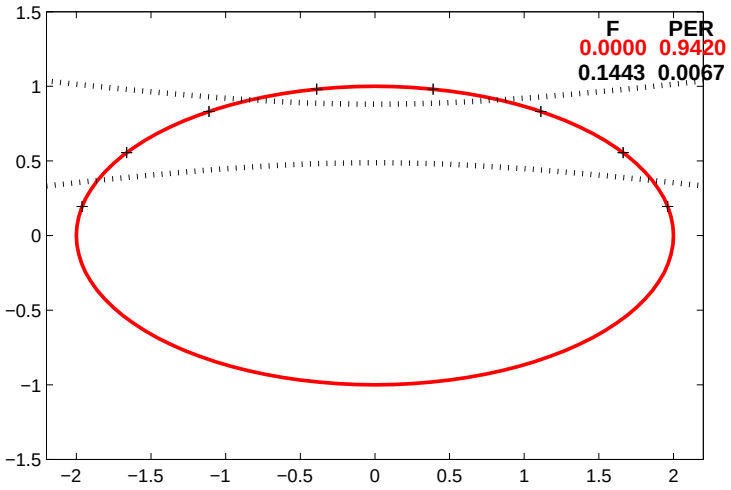

Figure 4.6: Examples of local minima when $n = 8$ points are placed along upper half of an ellipse

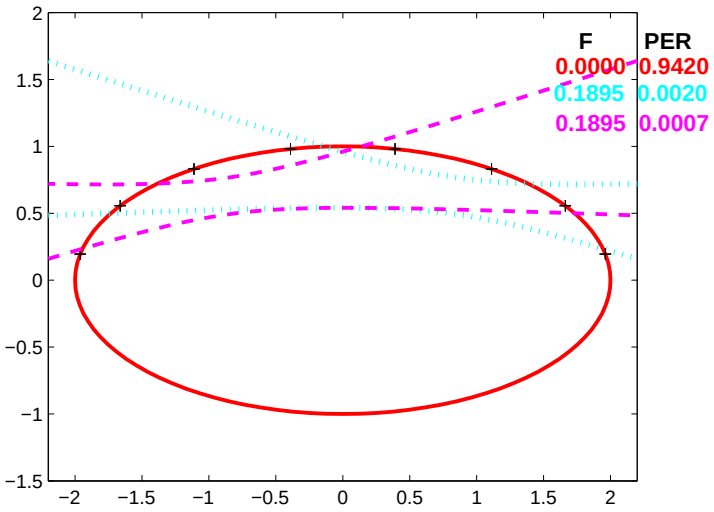

Figure 4.7: Examples of local minima when $n = 8$ points are placed along upper half of an ellipse

chose $\varphi_{\text{ini}} = 0$ and $\varphi_{\text{end}} = \pi/2$). In this example we have one global minimum (the interpolating ellipse, red) and one local minimum (the blue hyperbola).

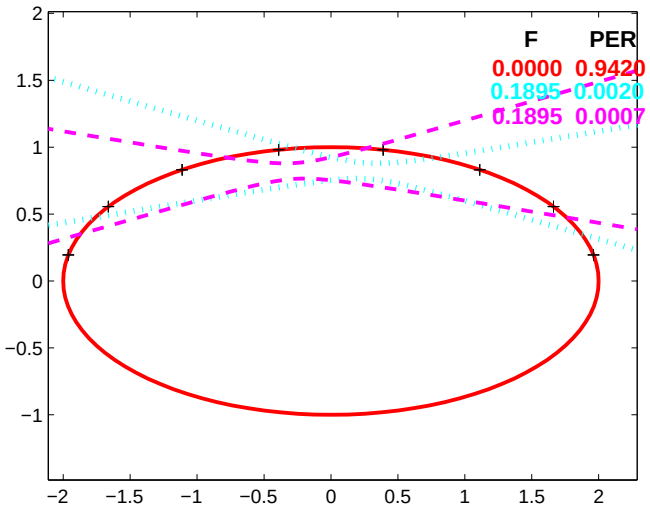

Figure 4.8: Examples of local minima when $n = 8$ points are placed along upper half of an ellipse

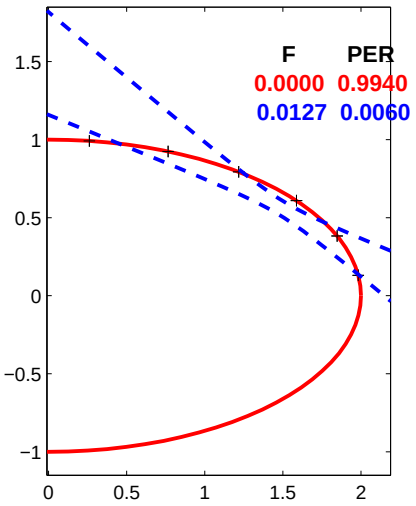

Figure 4.9: Examples of local minima when $n = 6$ points are placed along quarter of an ellipse

Again, from an overwhelming majority (99%) of randomly generated initial conics the minimization procedure converged to the global minimum.

4.4.4. Local Minima for Data with Gaussian Noise. We have also run the above tests for data points with added Gaussian noise (at levels $\sigma = 0.05$ and $\sigma = 0.1$). We observed quite similar picture: one global minimum and several (up to 8) local minima. The largest number of local minima in a single data set we found was 8.

The global minimum is usually an ellipse (close to the "true" ellipse on which the original, unperturbed points are placed) when the noise is small. For larger noise the global minimum is often a branch of a hyperbola that is close to the elliptic arc containing the original points. See section 2.4.1 for percentages of samples for which the global minimum is a hyperbola.

On the contrary, the local minima are mostly hyperbolas whose both branches run through the corridor containing the perturbed points. Less frequently local minima are long narrow ellipses whose both halves run through the corridor containing perturbed points. See the figure labeled "Corridor diagram" below.

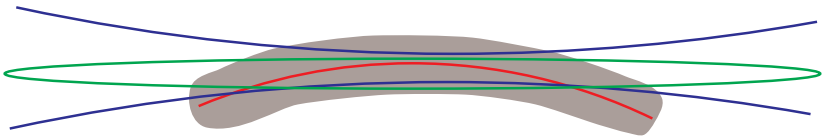

Figure 4.10: Corridor diagram

Note that Grey corridor contains noisy points scattered around red elliptic arc . Two branches of blue hyperbola and two branches of long narrow green ellipse run through the corridor.

4.4.5. Numerical Tests for Local Minima: Data Points with Noise or Random Data Points. In last Section, we have investigated local minima for data points without noise. Here we continue to present more detailed numerical results for the local minima of $\mathcal{F}$ when data points are generated along an ellipse with random Gaussian noise or randomly with a uniform distribution in a unit square.

In typical applications, the points are usually sampled from a predefined curve with some noises. So we investigated local minima for data points placed on an ellipse with Gaussian noise. The general description of our computer experiment is given in section 4.4.2. In this particular test we used the ellipse with semiaxes

$$a = 2 \quad \text{and} \quad b = 1$$

and placed n points (equally spaced) with a random Gaussian noise at level

$$\sigma = 0.05, \quad \sigma = 0.1$$

along the entire ellipse or along a certain arc of the ellipse. In other words, 0.05 or 0.1 multiplied by a normally distributed random number $t \in \mathcal{N}[0, 1]$ were added to the x and y coordinate of true data points on ellipse. Note that we want to investigate the number of local minima and their types (ellipses vs. hyperbolas). So for each sample, starting at 1000 different, randomly selected initial guesses, every point of convergence was recorded as a minimum (local or global) of $\mathcal{F}$. Thus we counted the number of local minima for each sample. Then the experiment was repeated for 2000 samples of n points. Out of 2000 different samples, the percentages of samples having 0 , 1 , 2 or more local minima were recorded. In order to see the types of local minima, the results will be presented in tables of the following form:

Percentage(%)	0	1	≥ 2
0	p_{00}	p_{01}	p_{02}
1	p_{10}	p_{11}	p_{12}
≥ 2	p_{20}	p_{21}	p_{22}

Where p_{00} represents percentage of samples having only global minimum and no local minimum, p_{01} represents percentage of samples having 0 ellipse local minimum and 1 hyperbola local minimum, p_{10} represents percentage of samples having 1 ellipse local minimum and 0 hyperbola local minimum, p_{02} represents percentage of samples having 0 ellipse local minimum and 2 or more hyperbola local minima and etc.

Upper Half of Ellipse, $n = 6$:

Percentage(%)	0	1	≥ 2
0	0	8.05	91.90
1	0.05	0	0
≥ 2	0	0	0

Table 4.3: Upper half of ellipse, $n = 6,\ \sigma = 0.05$

Percentage(%)	0	1	≥ 2
0	0.55	29.35	61.00
1	4.15	4.45	0.15
≥ 2	0.25	0.10	0

Table 4.4: Upper half of ellipse, $n = 6,\ \sigma = 0.1$

We observed that almost all samples have one global minimum and one or two local minima. More precisely, other than the global minimum (mostly an ellipse), the objective function tends to have one or two more hyperbola local minima for over 90% of samples.

Upper Half of Ellipse, $n = 8$:

Percentage(%)	0	1	≥ 2
0	0	0.60	20.40
1	0	1.40	52.00
≥ 2	0	1.70	23.90

Table 4.5: Upper half of ellipse, $n = 8,\ \sigma = 0.05$

Percentage(%)	0	1	≥ 2
0	0	0.70	30.70
1	0	3.00	46.40
≥ 2	0.90	1.80	16.50

Table 4.6: Upper half of ellipse, $n = 8,\ \sigma = 0.1$

For $n = 8$ points placed along the upper half of the same ellipse with random Gaussian noise, the function $\mathcal{F}$ for most data sets has one global minimum and several local minima. The largest number of local minima in a single data set we found was 8.

Right Half of Ellipse, $n = 6$

Percentage(%)	0	1	≥ 2
0	100.00	0	0
1	0	0	0
≥ 2	0	0	0

Table 4.7: Right half of ellipse, $n = 6,\ \sigma = 0.05$

Percentage(%)	0	1	≥ 2
0	96.95	3.05	0
1	0	0	0
≥ 2	0	0	0

Table 4.8: Right half of ellipse, $n = 6,\ \sigma = 0.1$

We found that when $n = 6$ data points are placed along the right half of the ellipse with random Gaussian noise, the objective function has no local minimum for almost

all samples. One hyperbola local minimum occurs only for 3% of samples when a bit large noises added to the true points. It demonstrates the fact that when data points are sampled along an elliptic arc with high curvature, the objective function tends to have one global minimum and no local minima.

Right Half of Ellipse, $n = 8$:

Percentage(%)	0	1	≥ 2
0	99.30	0.70	0
1	0	0	0
≥ 2	0	0	0

Table 4.9: Right half of ellipse, $n = 8,\ \sigma = 0.05$

Percentage(%)	0	1	≥ 2
0	86.40	13.10	0.40
1	0.10	0	0
≥ 2	0	0	0

Table 4.10: Right half of ellipse, $n = 8,\ \sigma = 0.1$

Similarly, the function $\mathcal{F}$ for over 99% of the data sets when $\sigma = 0.05$, 86% of the data sets when $\sigma = 0.1$ has one global minimum and no local minima because points are sampled along an elliptic arc with high curvature.

Quarter of Ellipse, $n = 6$:

In this experiment, we chose the upper right quarter of the same ellipse and placed $n = 6$ points along it with random Gaussian noise. Note the objective function of majority of samples has one global minimum and at least one local minimum.

Quarter of Ellipse, $n = 8$:

Percentage(%)	0	1	≥ 2
0	4.60	54.50	31.80
1	6.50	2.00	0.60
≥ 2	0	0	0

Table 4.11: Quarter of ellipse, $n = 6,\ \sigma = 0.05$

Percentage(%)	0	1	≥ 2
0	14.40	47.20	19.10
1	10.10	7.60	1.20
≥ 2	0.30	0.10	0

Table 4.12: Quarter of ellipse, $n = 6,\ \sigma = 0.1$

Percentage(%)	0	1	≥ 2
0	0	0.90	42.20
1	0.40	3.20	39.90
≥ 2	0.50	2.40	10.50

Table 4.13: Quarter of ellipse, $n = 8,\ \sigma = 0.05$

Percentage(%)	0	1	≥ 2
0	0.90	6.60	41.40
1	2.00	6.40	30.00
≥ 2	0.20	2.80	9.70

Table 4.14: Quarter of ellipse, $n = 8,\ \sigma = 0.1$

When the number of data points increases to 8, the global minimum of $\mathcal{F}$ is usually achieved by a hyperbola (see section 2.4.1) and the objective function tends to have more local minima. Note that in this experiment, the objective function for most data sets has one global minimum and several (up to 8) local minima.

Local Minima for Data with a Uniform Distribution. Now we move on to the complete random samples: generating data points with a uniform distribution in a square. In this experiment, n points were generated randomly with a uniform distribution in the unit square $[0,1]\times[0,1]$. Again, we want to investigate the number of local minima and their types (ellipses vs. hyperbolas).

Percentage(%)	0	1	≥ 2
0	59.60	28.25	4.15
1	4.90	2.40	0.35
≥ 2	0.20	0.10	0.05

Table 4.15: Local minima for data points with a uniform distribution, $n = 6$

Percentage(%)	0	1	≥ 2
0	41.85	29.30	13.05
1	6.95	5.20	2.35
≥ 2	0.45	0.60	0.25

Table 4.16: Local minima for data points with a uniform distribution, $n = 7$

4.4.6. Summary. When data points are sampled along the entire ellipse, or an elliptic arc with high curvature, the objective function tends to have one global minimum and no local minima.

When data points are sampled along an elliptic arc with low curvature, they appear in a corridor around the arc like the one shown in the "Corridor diagram". In

Percentage(%)	0	1	≥ 2
0	33.15	25.00	20.35
1	6.95	5.05	5.75
≥ 2	1.25	1.20	1.30

Table 4.17: Local minima for data points with a uniform distribution, $n = 8$

Percentage(%)	0	1	≥ 2
0	27.65	21.65	24.40
1	6.10	7.10	6.65
≥ 2	1.65	1.85	2.95

Table 4.18: Local minima for data points with a uniform distribution, $n = 9$

Percentage(%)	0	1	≥ 2
0	24.95	18.95	24.15
1	7.75	6.35	9.55
≥ 2	2.05	2.05	4.20

Table 4.19: Local minima for data points with a uniform distribution, $n = 10$

that case the desirable fit is given by an elliptic arc or, occasionally, by a hyperbolic arc stretching through the corridor. At the same time there might be distractive fits (nuisance fits) made by hyperbolas or long narrow ellipses whose both branches run through the corridor; see again the "Corridor diagram". While those undesirable fits may correspond to local minima of the objective function $\mathcal{F}$, those minima tend to be small and narrow, so that the chance of falling into one of them is low. On the contrary, the desirable fit tends to correspond to a wide minimum of $\mathcal{F}$, which

is likely attract the minimization procedures starting even from a randomly chosen initial conic.

Part II

CHAPTER 5

Projection onto Quadratic Curves

Geometric fitting, also known as best fitting, is based on minimizing the geometric distances from the given points to the fitted feature. Thus it is essential for us to know how to locate the nearest point on the conics for a given point so that we can measure the distances between them. In practice, various heuristic projection algorithms are employed [**7, 6, 5, 4, 64**] that are relatively fast, but their convergence is not guaranteed (and occasionally they do fail). On the other hand, certain theoretically reliable methods were proposed [**9, 61**], but most of them are overly complicated and too slow for practical use. In this chapter, we will discuss several projection methods and introduce a remarkable fast and totally reliable algorithm which was first found by D. Eberly in 2004 [**26**, Section 14.13.1] and then adapted and extended by us.

5.1. Introduction

It is sufficient to solve this problem when ellipses, hyperbolas and parabolas are in the standard position, in other words, when they are centered at origin and are axis-aligned with the major axis on the x-axis. All the other ellipses/hyperbolas/parabolas can be rotated and translated to such system and thus the distances can be measured. The coordinate transformation is very common and trivial in conics fitting. The procedure is described as below: Let (X_c, Y_c) be the center of ellipse and hyperbola (or the vertex of parabola), α be the pose angle. If (X,Y) is the original given point, (x, y) is the point after transformation, then

$$\begin{pmatrix} x \\ y \end{pmatrix} = \mathbf{R} \cdot \begin{pmatrix} X - X_c \\ Y - Y_c \end{pmatrix}$$

where $\mathbf{R}$ is the rotation matrix, i.e.,

$$\mathbf{R} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix}$$

with $c = \cos(\alpha)$, $s = \sin(\alpha)$.

Reversibly,

$$\begin{pmatrix} X \\ Y \end{pmatrix} = \mathbf{R}^{-1} \cdot \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} X_c \\ Y_c \end{pmatrix}$$

5.2. Eberly's Method

A remarkable approach to projecting points onto ellipses was found by D. Eberly in 2004 [**26**, Section 14.13.1]. Not only it produces the desired projection faster than anything known previously (including heuristic schemes), but it comes with a mathematical proof of convergence to the correct projection point in all cases, i.e., it is completely reliable. Below we describe Eberly's method for ellipses and then adapt it to other quadratic curves. In each case we provide a proof of convergence. We consider such proofs as an important asset of the proposed methods.

5.2.1. Ellipse. Let (u, v) be the given point. Let the ellipse in standard position be

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1. \tag{5.1}$$

with $a \geq b > 0$.

If the closest point on the ellipse is (x, y), then $(u - x, v - y)$ must be normal to the ellipse. An outward pointing ellipse normal is

$$\frac{1}{2}\nabla(\frac{x^2}{a^2} + \frac{y^2}{b^2} - 1) = (\frac{x}{a^2}, \frac{y}{b^2}) \tag{5.2}$$

Thus

$$(u - x, v - y) = t(\frac{x}{a^2}, \frac{y}{b^2}) \tag{5.3}$$

must be satisfied for some value t. Then (5.1) and (5.3) will be used to determine the value of t.

Due to the obvious symmetry, it is enough to work in the first quadrant $u > 0, v > 0$; then the projection point (x, y) will also be in the first quadrant, i.e., $x > 0, y > 0$. (Other points can be reflected to the first quadrant about the axes, and then the projection point can be reflected back.)

Solving equation (5.3) for x and y, we obtain

$$x = \frac{a^2 u}{t + a^2} \quad \text{and} \quad y = \frac{b^2 v}{t + b^2} \tag{5.4}$$

Fist let us consider the general case when $u > 0, v > 0$ (the degenerate cases when $u = 0$ or $v = 0$ are fairly simple and can be handled separately). Based on the fact that the closest point (x, y) on ellipse should also lie in the same quadrant as the given point, we have $x > 0, y > 0$, thus yielding $t > -a^2, t > -b^2$. Assuming $a \geq b$, the final constraint for t is $t > -b^2$. Substituting (5.4) into (5.1), we obtain a function

$$F(t) = (\frac{au}{t + a^2})^2 + (\frac{bv}{t + b^2})^2 - 1 = 0 \tag{5.5}$$

whose root we need to find (because (x, y) must lie on the ellipse). Once we solve equation (5.5) for t, we can compute the projection point (x, y) by (5.4).

If we differentiate this rational function of t,the first derivative of F is

$$F'(t) = \frac{-2a^2 u^2}{(t + a^2)^3} + \frac{-2b^2 v^2}{(t + b^2)^3} \tag{5.6}$$

and the second derivative is

$$F''(t) = \frac{6a^2 u^2}{(t + a^2)^4} + \frac{6b^2 v^2}{(t + b^2)^4} \tag{5.7}$$

For any point (u, v) with $u > 0, v > 0$, we can observe that $F'(t) < 0$ and $F''(t) > 0$ for $t > -b^2$, showing that $F(t)$ is a monotonically decreasing function and concave for $t \in (-b^2, \infty)$. Also note that

$$\lim_{t \to -b^2+} F(t) = +\infty \quad \text{and} \quad \lim_{t \to \infty} F(t) = -1.$$

It starts with positive values and then becomes negative and goes to -1. See Fig. 5.1. It is clear that this function has a unique root on the specified domain.

Thus standard Newton's method starting at any point t_0 where $F(t_0) > 0$ will converge to the unique root of F. Given an initial guess t_0,

$$t_{n+1} = t_n - \frac{F(t_n)}{F'(t_n)}, \quad n \geq 0 \tag{5.8}$$

Choosing a proper initial guess for which the method converges is important, Eberly suggests to start with $t_0 = bv - b^2$, because $F(t_0) > 0$ is guaranteed by (5.5). We found that it is more beneficial to start with

$$t_0 = \max\{au - a^2, bv - b^2\}. \tag{5.9}$$

Then Newton's method converges in 4-6 iterations in all practical cases and finds the root to within 10-12 significant digits. This is, on average, 2-3 times faster than solving equation of degree four or using general heuristics [**7, 4**]. The MATLAB code for this method is posted on our web page [**32**].

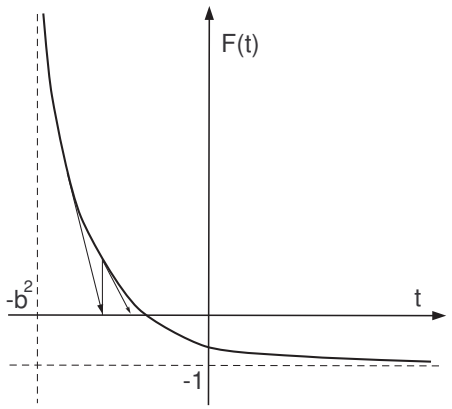

Figure 5.1: A typical graph of $F(t)$ for $t > -b^2$ and the progress of Newton's iterations toward the root.

5.2.2. Hyperbola. Now let us project a point (u, v) onto a hyperbola. Again, the latter can be defined in its canonical coordinates:

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1 \tag{5.10}$$

The orthogonal condition is

$$(u - x, v - y) = t(\frac{x}{a^2}, -\frac{y}{b^2}) \tag{5.11}$$

for some value of t. Due to symmetry, we restrict our method to $u > 0$ and $v > 0$. Then the closest point (x, y) on hyperbola must also be in the first quadrant, thus making $t \in (-a^2, b^2)$. Equation (5.11) implies that

$$x = \frac{a^2 u}{t + a^2} \quad \text{and} \quad y = \frac{b^2 v}{-t + b^2}. \tag{5.12}$$

Substituting (5.12) into (5.10) we obtain a function

$$F(t) = (\frac{au}{t + a^2})^2 - (\frac{bv}{-t + b^2})^2 - 1 = 0, \tag{5.13}$$

whose root we need to find. Note that

$$\lim_{t \to -a^2} F(t) = +\infty \quad \text{and} \quad \lim_{t \to b^2} F(t) = -\infty$$

Taking the derivatives of F we see that

$$F'(t) = \frac{-2a^2u^2}{(t + a^2)^3} + \frac{-2b^2v^2}{(-t + b^2)^3} \tag{5.14}$$

Hence $F'(t) < 0$ for all $t \in (-a^2, b^2)$. Next,

$$F''(t) = \frac{6a^2u^2}{(t + a^2)^4} - \frac{6b^2v^2}{(-t + b^2)^4} \tag{5.15}$$

Now F'' decreases from $+\infty$ (near $-a^2$) to $-\infty$ (near b^2), and it is monotonic (because $F''' < 0$, as one can easily verify). Thus F has a unique inflection point, t_*, within the interval $(-a^2, b^2)$. See Fig. 5.2, where two possible cases are shown: (a) the inflection point lies above the x axis, i.e., $F(t_*) > 0$ and (b) the inflection point lies below the x axis.

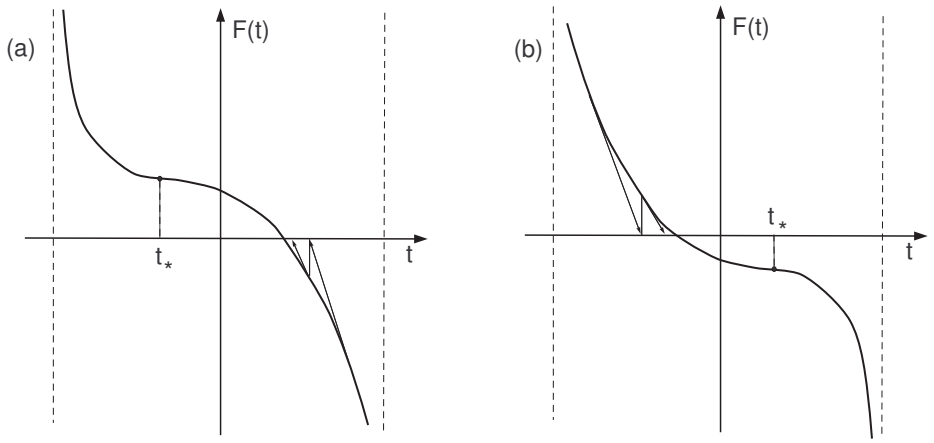

Figure 5.2: Two possible appearances of $F(t)$ on the interval $-a^2 < t < -b^2$. Arrows show the progress of Newton's iterations toward the root.

The inflection point is found by solving $F'' = 0$, hence

$$t_* = \frac{b^2\sqrt{au} - a^2\sqrt{bv}}{\sqrt{au} + \sqrt{bv}}.$$

Now by computing $F(t_*)$ we can determine which case, (a) or (b), we have at hand. Standard Newton's method will converge to the root of $F(t) = 0$, but the starting point t_0 must be selected wisely. Let

$$F_0 = \frac{u^2}{a^2} - \frac{v^2}{b^2} - 1,$$

then we choose our initial guesses according to the following chart 5.1:

t_0	$F_0 > 0$	$F_0 < 0$
$F(t_*) > 0$	$\min\left(\frac{(a^2+b^2)bv}{au+bv}, a^2 + b^2 - au\right)$	a^2
$F(t_*) < 0$	b^2	$\min\left(\frac{(a^2+b^2)au}{au+\sqrt{2}bv}, \frac{au}{\sqrt{2}}\right)$

Table 5.1: Initial choices for projecting points onto a hyperbola

This algorithm is guaranteed to converge, according to our analysis. It converges in 6-8 iterations per point, on average and finds the root to within 10-12 significant digits. The MATLAB code for this method is posted on our web page [**32**].

5.2.3. Parabola. Let the parabola be

$$y^2 = 2px, (p > 0) \tag{5.16}$$

and (u, v) be the given point, (x, y) be the nearest point. The normal to the parabola is

$$\frac{1}{2}\nabla(y^2 - 2px) = (-p, y) \tag{5.17}$$

Then the orthogonal condition follows:

$$(u - x, v - y) = t(-p, y) \tag{5.18}$$

Solving for x and y, we get

$$x = u + tp \quad \text{and} \quad y = \frac{v}{t+1} \tag{5.19}$$

For parabola, we may restrict our attention to $v > 0$ because it is symmetric about x-axis. So the contacting point (x, y) should also lie in the first two quadrants, in other words, y has to be greater than 0. Since $y > 0$, we have constraint $t > -1$. Substituting (5.19) into (5.16), we obtain a function

$$F(t) = (\frac{v}{t+1})^2 - 2p(u + tp) - 1 = 0, \tag{5.20}$$

whose root we need to find. Differentiating $F(t)$, the first derivative is

$$F'(t) = \frac{-2v^2}{(t+1)^3} - 2p^2 \tag{5.21}$$

and the second derivative is

$$F''(t) = \frac{6v^2}{(t+1)^4} \tag{5.22}$$

Note that

$$\lim_{t \to -1} F(t) = +\infty \quad \text{and} \quad \lim_{t \to \infty} F(t) = -\infty$$

Thus on the interval $(-1, \infty)$ we have $F' < 0$ and $F'' > 0$, i.e., the function F is monotonically decreasing and concave. Now standard Newton's method starting at

any point t_0 where $F(t_0) > 0$ will converge to the unique root of F. Initial guesses can be chosen as follows:

- if $u \leq p$, then $t_0 = (\frac{v}{\sqrt{2p}})^{\frac{2}{3}}$;
- if $u > p$, then $t_0 = \min\,(\frac{v}{2\sqrt{(u-p)p}}, (\frac{v}{2p})^{\frac{2}{3}})$;

Then Newton's method converges in 6-8 iterations on average and finds the root to within 10-12 significant digits. The MATLAB code for this method is posted on our web page [**32**].

5.3. Root Finding Method

In this section, we briefly describe the most straightforward method for finding the projections. It is based on finding the roots of a polynomial of degree four. This method does not involve any iteration, however, it is quite impractical and virtually never used since solving cubic or quartic equation is numerically unstable and time-consuming.

5.3.1. Ellipse. An ellipse can be uniquely described by 5 parameters, the center coordinate X_c, Y_c, axis lengths a, b ($a \geq b$), and pose angle α($-\pi/2 < \alpha \leq \pi/2$). After rotating and translating, the 3 of 5 parameters (X_c, Y_c, α) disappear and the ellipse will be described only with axis lengths a, b as below (standard position),

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1. \tag{5.23}$$

Let (u, v) be a given point in the canonical system, (x, y) be the closest point on the ellipse, then the tangent line at (x, y) and the connecting line of two points must be perpendicular to each other:

$$\frac{dy}{dx} \cdot \frac{v-y}{u-x} = -1 \tag{5.24}$$

Rewrite (5.23) and (5.24),we have:

$$f_1 = a^2y^2 + b^2x^2 - a^2b^2 = 0 \tag{5.25}$$

$$f_2 = b^2x(v-y) - a^2y(u-x) = 0 \tag{5.26}$$

The closest point (x, y) on the ellipse must simultaneously satisfy equation (5.25) and (5.26). From equation(5.26), we can express y in terms of x and then replace y by that expression in equation (5.25). Thus we have combined two equations into one quartic equation:

$$Ax^4 + Bx^3 + Cx^2 + Dx + E = 0 \tag{5.27}$$

where

$$A = (a^2 - b^2)^2$$

$$B = -2a^2u(a^2 - b^2)$$

$$C = a^2b^2v^2 + a^4u^2 - a^2(a^2 - b^2)^2$$

$$D = 2a^4u(a^2 - b^2)$$

$$E = -a^6u^2$$

This quartic equation can be solved using a method discovered by Lodovico Ferrari in 1540's. Then one solution (x, y) which has the shortest distance among the maximum 4 real solutions will be the desired closest point on the ellipse.

5.3.2. Hyperbola. A hyperbola in standard position can be described as below:

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1. \tag{5.28}$$

The only difference compared with an ellipse in standard position is the sign of b^2. Thus, we can simply modify the same method into hyperbola case. What we have to do is change the sign of b^2 in all equations from (5.23) to (5.27) and also allow a to be smaller than b. That is,

$$f_3 = a^2y^2 - b^2x^2 + a^2b^2 = 0 \tag{5.29}$$

$$f_4 = -b^2x(v - y) - a^2y(u - x) = 0 \tag{5.30}$$

$$Ax^4 + Bx^3 + Cx^2 + Dx + E = 0 \tag{5.31}$$

where

$$A = (a^2 + b^2)^2$$

$$B = -2a^2u(a^2 + b^2)$$

$$C = -a^2b^2v^2 + a^4u^2 - a^2(a^2 + b^2)^2$$

$$D = 2a^4u(a^2 + b^2)$$

$$E = -a^6u^2$$

Similarly, we can solve this quartic equation by using the same method and choose the solution which makes the distance between the given point and itself the shortest.

5.3.3. Parabola. A parabola can be uniquely described with 4 parameters, the vertex coordinates X_c, Y_c, focus distance $p(> 0)$ from directrix, and pose angle $\alpha(-\pi < \alpha \leq \pi)$. After coordinate transformation, a parabola in canonical system can be described with only one parameter, the focus distance p, as below:

$$y^2 = 2px \tag{5.32}$$

Based on the fact that the tangent line at the nearest point(x, y) on parabola should be perpendicular to the connecting line between $(x, y)$and the given point (u, v), we have the following equation:

$$\frac{p}{y} \cdot \frac{v - y}{u - x} = -1 \tag{5.33}$$

Then, as usual, these two equations may be written as:

$$f_5 = y^2 - 2px = 0 \tag{5.34}$$

$$f_6 = y(u - x) + p(v - y) = 0 \tag{5.35}$$

In equation (5.35), we solve for x:

$$x = u + \frac{p(v - y)}{y} \tag{5.36}$$

Replacing (5.36) in equation (5.34) yields

$$y^3 + 2p(p-u)y - 2p^2v = 0 \tag{5.37}$$

Instead of getting a quartic equation as for ellipse and hyperbola, we obtain a cubic equation for parabola which is even easier to find roots. Among the maximum 3 real roots, the solution with the shortest distance from the given point will be chosen.

5.4. Ahn's Method

Because of the impractical reason of root finding method, Sung Joon Ahn proposed a method in his paper [**7**]. We include Ahn's method here.

5.4.1. Ellipse. The first few steps are the same as root finding method. Once (5.25) and (5.26) were obtained, they were solved simultaneously by using generalized Newton method as following:

$$f_1 = \frac{1}{2}(a^2y^2 + b^2x^2 - a^2b^2) = 0 \tag{5.38}$$

$$f_2 = b^2x(v-y) - a^2y(u-x) = 0 \tag{5.39}$$

$$\begin{aligned} \mathbf{Q} &= \begin{pmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{pmatrix} \\ &= \begin{pmatrix} b^2x & a^2y \\ (a^2-b^2)y + b^2v & (a^2-b^2)x - a^2u \end{pmatrix} \end{aligned} \tag{5.40}$$

$$\mathbf{Q}_k \Delta\mathbf{x} = -\mathbf{f}(x_k) \tag{5.41}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x} \tag{5.42}$$

In this iterative process, Ahn supplies the initial value $\mathbf{x}_0$ to start with. Since the nearest point on the ellipse should lie in the same quadrant as the given point, $\mathbf{x}_0$ is given below:

$$\mathbf{x}_0 = \frac{1}{2}(\mathbf{x}_{k1} + \mathbf{x}_{k2}) \tag{5.43}$$

where

$$\mathbf{x}_{k1} = \begin{pmatrix} u \\ v \end{pmatrix} \frac{ab}{\sqrt{b^2u^2 + a^2v^2}} \tag{5.44}$$

and

$$\mathbf{x}_{k2} = \begin{cases} \begin{pmatrix} u \\ \text{sign}(v)\frac{b}{a}\sqrt{a^2 - u^2} \end{pmatrix} & \text{if} |u| < a, \\ \begin{pmatrix} \text{sign}(u)a \\ 0 \end{pmatrix} & \text{if} |u| \geq a. \end{cases} \tag{5.45}$$

In detail, the connecting line of (u, v) and $(0, 0)$ intersects the ellipse, the intersection point is $\mathbf{x}_{k1}$. If $u < a$, project the given point (u, v) onto the x-axis, the projection line intersects the x-axis with $\mathbf{x}_{k2}$; If $u \geq a$, choose $(\text{sign}(u)a, 0)$ as $\mathbf{x}_{k2}$. Then the initial value is found by choosing the midpoint of $\mathbf{x}_{k1}$ and $\mathbf{x}_{k2}$.

Remark: The Jacobian matrix $\mathbf{Q}$ is singular if the given point (u, v) lies at the ellipse center and if the a and b are equal, which indicates that there is no unique nearest point for the circle center on a circle.

5.4.2. Hyperbola. As we have seen before, the standard form for hyperbola is similar to ellipse in standard position, except the sign of b^2. So Ahn made a simple modification to his method for hyperbola. First, a can be smaller than b. Then, replace each b^2 in equations (5.23) - (5.26) and (5.40) by $-b^2$. Last, give the initial

values for the nearest points

$$\mathbf{x}_0 = \begin{cases} \begin{pmatrix} \text{sign}(u)a \\ 0 \end{pmatrix} & \text{if} |u| \le a, \\ \begin{pmatrix} u \\ \text{sign}(v)\frac{b}{a}\sqrt{u^2 - a^2} \end{pmatrix} & \text{if} |u| > a \quad \text{and} \quad a \ge b, \\ \begin{pmatrix} \text{sign}(u)\frac{a}{b}\sqrt{v^2 + b^2} \\ v \end{pmatrix} & \text{if} |u| > a \quad \text{and} \quad a < b \end{cases} \tag{5.46}$$

Here, $\mathbf{x}_0$ is chosen by intersecting the projection line onto x-axis with the hyperbola when $u > a$. And if $u \le a$, the vertex point in the same quadrant $(\text{sign}(u)a, 0)$ is simply picked.

5.4.3. Parabola. In a similar way, Ahn derived the orthogonal conditions as (5.34) and (5.35) for parabola.

$$f_5 = \frac{1}{2}(y^2 - 2px) = 0 \tag{5.47}$$

$$f_6 = y(u - x) + p(v - y) = 0 \tag{5.48}$$

Then the nearest points are found by using generalized Newton method with Jacobian matrix

$$\begin{aligned} \mathbf{Q} &= \begin{pmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{pmatrix} \\ &= \begin{pmatrix} -p & y \\ -y & u - x - p \end{pmatrix} \end{aligned} \tag{5.49}$$

and the initial starting value for the nearest point is given below:

$$\mathbf{x}_0 = \begin{cases} \begin{pmatrix} 0 \\ 0 \end{pmatrix} & \text{if} |u| < 0, \\ \begin{pmatrix} u \\ \text{sign}(v)\sqrt{2pu} \end{pmatrix} & \text{if} |u| \geq 0. \end{cases} \tag{5.50}$$

5.5. Comparison

Let i be the average number of the iterations that each method takes to converge, per point. Then the comparison of three projection methods are as follows:

Ellipse	Iters	Flops	Time
Eberly's method	4-5	$35 + 16i \approx 107$	*fastest*
Root method	n/a	165	*
Ahn's method	4.3-5.3	$(39 + 49i) \approx 274$	*

Table 5.2: Comparison of three projection methods for ellipse.

From Table. 5.2, 5.3 and 5.4, we can see that our projection method (extended from Eberly's method to all quadratic curves) takes the least number of iterations and runs faster than Ahn's iterative method and the non-iterative root finding method as well.

Hyperbola	Iters	Flops	Time
Eberly's method	6-7	$51 + 16i \approx 155$	*fastest*
Root method	n/a	165	*
Ahn's method	16-17	$30 + 49i \approx 838$	*

Table 5.3: Comparison of three projection methods for hyperbola.

Parabola	Iters	Flops	Time
Eberly's method	4.5	$39 + 15i \approx 106$	*comparable*
Root method	n/a	100	*fastest*
Ahn's method	5.3	$27 + 38i \approx 228$	*

Table 5.4: Comparison of three projection methods for parabola.

CHAPTER 6

Geometric Fits: Minimization of the Objective Function

In this Chapter we discuss practical solutions to the problem of fitting implicit curves. Let $(x_1, y_1), \dots, (x_n, y_n)$ denote the observed points and $P(x, y; \Theta) = 0$ be the equation of the fitted contour, where Θ represents the vector of unknown parameters. For example, ellipses can be defined by

$$\frac{\breve{x}^2}{a^2} + \frac{\breve{y}^2}{b^2} - 1 = 0 \tag{6.1}$$

in the canonical coordinates $\breve{x}, \breve{y}$, which can be related to x, y by translation and rotation, i.e., $\breve{x} = (x - x_c)\cos\alpha + (y - y_c)\sin\alpha$ and $\breve{y} = -(x - x_c)\sin\alpha + (y - y_c)\cos\alpha$; now $\Theta = (x_c, y_c, a, b, \alpha)$, where (x_c, y_c) is the center, a, b are the semiaxes, and α is the angle of tilt.

Our task is to minimize geometric distances from the observed points to the fitting curve:

$$\mathcal{F}(\Theta) = \sum_{i=1}^{n} d_i^2 = \sum_{i=1}^{n} (x_i - x_i')^2 + (y_i - y_i')^2 \quad \to \quad \min . \tag{6.2}$$

Here d_i denotes the geometric distance from the observed point (x_i, y_i) to the fitting contour, and (x_i', y_i') the (orthogonal) projection of (x_i, y_i) onto the contour.

6.1. Introduction

The first thorough investigation of the ellipse fitting problem was done in the middle 1990's by Gander, Golub, and Strebel [**29**]. They developed a roundabout way of minimizing $\mathcal{F}$ by using auxiliary parameters ω_i, $i = 1, \dots, n$, describing the location of the projected points (x_i', y_i') on the ellipse; the latter, in the canonical coordinates (6.1) were expressed by $\breve{x}_i' = a\cos\omega_i$ and $\breve{y}_i' = b\sin\omega_i$. Thus the objective

function becomes

$$\mathcal{F} = \sum_{i=1}^{n} (x_i - x_c - a\cos\omega_i \cos\alpha - b\sin\omega_i \sin\alpha)^2 + (y_i - y_c + a\cos\omega_i \sin\alpha - b\sin\omega_i \cos\alpha)^2.$$

In [**29**], $\mathcal{F}$ was minimized with respect to $x_c, y_c, a, b, \alpha, \omega_1, \ldots, \omega_n$ *simultaneously.* This procedure avoids the calculation of d_i's, but the minimization in the $(n+5)$-dimensional parameter space is predictably cumbersome and slow. So Gander et. al. [**29**] concluded that the minimization of geometric distances for ellipses was a prohibitively difficult task.

Recently Sturm and Gargallo [**55**] modified the above method in several ways. In particular, they used a projective matrix that allowed them to describe conics of all types (ellipses, hyperbolas, parabolas) with the same set of 5 parameters. Thus their method could freely switch between types during iterations. But they still work in an $(n+5)$-dimensional parameter space, in that sense their method is similar to that of [**29**].

In the late 1990s, in computer vision applications various alternative fitting schemes were developed, where so called algebraic distances were minimized; we review them in Appendix A. They produce ellipses that fit less accurately than those minimizing geometric distances (6.2). Some authors say that "the performance gap between algebraic fitting and geometric fitting is wide..." (see [**4**, p. 12]).

In the early 2000's another approach to the minimization of geometric distances (6.2) emerged, due to Ahn et. al. [**7, 5, 4**], that turned out to be very efficient.

6.2. Implicit Fitting Method

Least squares problems are commonly solved by the Gauss-Newton (GN) method or its Levenberg-Marquardt (LM) correction. If one minimizes a sum of squares $\mathcal{F}(\Theta) = \sum f_i^2$, then both GN and LM would use the values of f_i's and their first derivatives with respect to Θ, which we denote by $(f_i)_\Theta$.

Now suppose, as before, that we fit a curve defined by implicit equation $P(x, y; \Theta) = 0$ to observed points (x_i, y_i). Our goal is to minimize the sum of squares (6.2). The GN and LM methods can be applied here in two ways: we either treat $\mathcal{F}$ as a sum of n squares, $\mathcal{F} = \sum d_i^2$, or a sum of $2n$ squares, $\mathcal{F} = \sum g_i^2 + \sum h_i^2$, where $g_i = x_i - x_i'$ and $h_i = y_i - y_i'$. In the former case we will need d_i's and their derivatives $(d_i)_\Theta$. In the latter we need g_i's and h_i's, as well as their derivatives $(g_i)_\Theta$ and $(h_i)_\Theta$. The resulting algorithm is said to be *distance-based* if one uses d_i's, or *coordinate-based* if one uses g_i's and h_i's; see Ahn et al. [**5, 4**].

Obviously, it is enough to know the projections (x_i', y_i') in order to compute d_i's, g_i's, and h_i's. But their derivatives $(d_i)_\Theta, (g_i)_\Theta, (f_i)_\Theta$ present a more challenging problem. Sometimes finite differences are used to approximate these derivatives, which reduces the accuracy of the fit. But there are surprisingly simple formulas for these derivatives:

Proposition. *Let (x, y) be a given a point and (x', y') denote its projection onto the curve $P(x, y; \Theta) = 0$ (then x', y' depend on Θ). Denote $g = x - x'$, $h = y - y'$, and $d^2 = g^2 + h^2$. Then we have*

$$g_\Theta = \frac{P_\Theta P_x^2 - g P_y (P_x P_{y\Theta} - P_y P_{x\Theta})}{P_x (P_x^2 + P_y^2)}, \tag{6.3}$$

$$h_\Theta = \frac{P_\Theta P_y^2 + h P_x (P_x P_{y\Theta} - P_y P_{x\Theta})}{P_y (P_x^2 + P_y^2)}, \tag{6.4}$$

and

$$d_\Theta = \frac{P_\Theta}{\sqrt{P_x^2 + P_y^2}}, \tag{6.5}$$

where P_Θ, P_x, P_y denote the first order partial derivatives of P with respect to Θ, x, y, respectively, and $P_{x\Theta}$ and $P_{y\Theta}$ the corresponding second order partial derivatives; all the derivatives are taken at the projection point (x', y').

Proof. Since the vector $(x - x', y - y')$ is orthogonal to the curve,

$$g = x - x' = t P_x \qquad \text{and} \qquad h = y - y' = t P_y \tag{6.6}$$

for some scalar t. This immediately gives

$$d^2 = g^2 + h^2 = t^2(P_x^2 + P_y^2). \tag{6.7}$$

Next we use differentiation with respect to Θ. Differentiating the identity $P(x', y'; \Theta) = 0$ gives

$$P_\Theta = -P_x x'_\Theta - P_y y'_\Theta = (g g_\Theta + h h_\Theta)/t, \tag{6.8}$$

and differentiating the identity $d^2 = g^2 + h^2$ gives

$$d d_\Theta = g g_\Theta + h h_\Theta = t P_\Theta. \tag{6.9}$$

Now (6.5) follows from (6.9) and (6.7). Differentiation of (6.6) gives

$$g_\Theta = t_\Theta P_x + t P_{x\Theta}, \qquad h_\Theta = t_\Theta P_y + t P_{y\Theta}.$$

Eliminating t_Θ from these two equations yields

$$g_\Theta P_y - h_\Theta P_x = -t(P_x P_{y\Theta} - P_y P_{x\Theta}). \tag{6.10}$$

Solving (6.9) and (6.10) for g_Θ and h_Θ we obtain (6.3) and (6.4). □

The formulas (6.3)–(6.5) were essentially obtained by Ahn et al. [**5, 4**]. Independently the formula (6.5) was derived in [**8**] (see eq. (24) there).

Practically, the calculation of the derivatives $d_\Theta, g_\Theta, h_\Theta$ by (6.3)–(6.5) is easy once the projection point (x', y') is located. The differentiation of $P(x, y; \Theta)$ with respect to Θ is usually straightforward; for example, one can easily find the derivatives of (6.1) with respect to x_c, y_c, a, b, α.

Alternatively, one can try to change the parametrization scheme in order to simplify differentiation. For example, instead of (6.1) one can define an ellipse by equation

$$\sqrt{(x - p_1)^2 + (y - p_2)^2} + \sqrt{(x - q_1)^2 + (y - q_2)^2} - 2a = 0 \tag{6.11}$$

where (p_1, p_2) and (q_1, q_2) denote its foci and $2a$, as before, the major axis. These are sometimes called *Kepler's parameters* of an ellipse; they have certain advantages

[**21**]. In particular, differentiation of (6.11) with respect to p_1, p_2, q_1, q_2, and a is quite straightforward.

We note that the coordinate-based scheme using g_i's and h_i's operates with $2n$ terms and involves the second order derivatives $P_{x\Theta}$ and $P_{y\Theta}$. The distance-based scheme operates only with n terms and does not involve the second order derivatives; cf. (6.5). As a result, the coordinate-based scheme seems to be less efficient, and we will only use the distance-based fit in what follows.

Remark. Since the distance d given must be differentiable, we have to treat it as a *signed* distance - it must be positive on one side of the curve and negative on the other. For ellipses, one can make $d > 0$ for points outside the ellipse and $d < 0$ for points inside.

6.3. Geometric Fitting of Ellipse

In the following four sections, we derive Jacobian matrices needed for our implicit fitting method, based on geometric parameters and algebraic parameters specifically. Let (x, y) be a given a point and (x', y') denote its projection onto the ellipse $P(x, y; \Theta) = 0$, where Θ represents the vector of unknown parameters.

Let $\Theta = (x_c, y_c, a, b, \alpha)$ be the standard geometric parameters. Ellipse can be defined by

(6.12)
$$\frac{((x - x_c)\cos\alpha + (y - y_c)\sin\alpha)^2}{a^2} + \frac{(-(x - x_c)\sin\alpha + (y - y_c)\cos\alpha)^2}{b^2} - 1 = 0, a > b$$

In other words,

$$P(x, y; \Theta) = ((x - x_c)c + (y - y_c)s)^2 b^2 + (-(x - x_c)s + (y - y_c)c)^2 a^2 - a^2 b^2 = 0,$$

where $s = \sin\alpha, c = \cos\alpha$. Taking the first order derivatives of P with respect to the Θ, x, y at the projection point (x', y'), we have

(6.13)
$$(P_\Theta) = (P_{x_c}, P_{y_c}, P_a, P_b, P_\alpha)$$

with

$$P_{x_c} = -(b^2c^2 + a^2s^2)(x - x_c) - cs(b^2 - a^2)(y - y_c) \tag{6.14}$$

$$P_{y_c} = -(b^2s^2 + a^2c^2)(y - y_c) - cs(b^2 - a^2)(x - x_c) \tag{6.15}$$

$$P_a = (x - x_c)^2 s^2 a + (y - y_c)^2 c^2 a - 2cs(x - x_c)(y - y_c)a - b^2 a \tag{6.16}$$

$$P_b = (x - x_c)^2 c^2 b + (y - y_c)^2 s^2 b + 2cs(x - x_c)(y - y_c)b - a^2 b \tag{6.17}$$

$$P_\alpha = (x-x_c)^2(a^2-b^2)cs+(y-y_c)^2(b^2-a^2)cs+(x-x_c)(y-y_c)(b^2-a^2)(c^2-s^2) \tag{6.18}$$

And

$$P_x = -P_{x_c}, \quad P_y = -P_{y_c}. \tag{6.19}$$

As discussed in Section 6.2,

$$(d)_\Theta = \frac{P_\Theta}{\sqrt{P_x^2 + P_y^2}} \quad \text{for each given point } (x, y)$$

Then Jacobian matrix can be derived as

$$\mathbf{J}(\Theta) = (d_i)_\Theta = ((d_i)_{x_c}, (d_i)_{y_c}, (d_i)_a, (d_i)_b, (d_i)_\alpha) \tag{6.20}$$

If we have n observed points, then we obtain an n by 5 Jacobian matrix. With the Jacobian matrix (6.20), and the signed distance (6.7), we construct n linear equations. The linear equations look like

$$\mathbf{J}(\Theta) \begin{pmatrix} \Delta x_c \\ \Delta y_c \\ \Delta a \\ \Delta b \\ \Delta \alpha \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{pmatrix} \tag{6.21}$$

Now Gauss-Newton(GN), Levenberg-Marquardt (LM) or Trust Region(TR) algorithm can be applied to solve these linear equations. The initial guess may be supplied from an algebraic fit, see Appendix A.

6.4. Geometric Fitting of Hyperbola

Let $\Theta = (x_c, y_c, a, b, \alpha)$ be the standard geometric parameters. Hyperbola can be defined by

$$\frac{((x-x_c)\cos\alpha + (y-y_c)\sin\alpha)^2}{a^2} - \frac{(-(x-x_c)\sin\alpha + (y-y_c)\cos\alpha)^2}{b^2} - 1 = 0 \tag{6.22}$$

The only difference compared to an ellipse is the sign of b^2. Thus the Jacobian matrix we derived for ellipses only needs some simple modifications for our hyperbola fitting algorithm. First we allow a to be smaller than b; then in equations (6.14), (6.15), (6.16) and (6.18), we replace b^2 by $-b^2$; also, we replace b in equation (6.17) by $-b$.

6.5. Geometric Fitting of Parabola

Let $\Theta = (x_c, y_c, p, \alpha)$ be the standard geometric parameters. Parabola can be defined by

$$(-(x-x_c)\sin\alpha + (y-y_c)\cos\alpha)^2 = 2p((x-x_c)\cos\alpha + (y-y_c)\sin\alpha), \tag{6.23}$$

where p is the focus distance to the directrix. Thus,

$$P(x, y; \Theta) = (-(x-x_c)s + (y-y_c)c)^2 - 2p((x-x_c)c + (y-y_c)s) = 0,$$

where $s = \sin\alpha, c = \cos\alpha$. Taking the first order derivatives of P with respect to Θ, x, y at the projection point (x', y'), we have

$$(P_\Theta) = (P_{x_c}, P_{y_c}, P_p, P_\alpha) \tag{6.24}$$

with

$$P_{x_c} = -2s^2(x-x_c) + 2cs(y-y_c) + 2pc \tag{6.25}$$

$$P_{y_c} = -2c^2(y - y_c) + 2cs(x - x_c) + 2ps \tag{6.26}$$

$$P_p = -2c(x - x_c) - 2s(y - y_c) \tag{6.27}$$

(6.28)

$$P_\alpha = 2sc(x-x_c)^2 - 2sc(y-y_c)^2 - 2(-s^2+c^2)(x-x_c)(y-y_c) + 2ps(x-x_c) - 2pc(y-y_c)$$

And

$$P_x = -P_{x_c}, \quad P_y = -P_{y_c} \tag{6.29}$$

Now we can write out

$$(d)_\Theta = \frac{P_\Theta}{\sqrt{P_x^2 + P_y^2}} \quad \text{for each given point } (x, y)$$

After obtaining an n by 4 Jacobian matrix for the n given points, the Gauss-Newton(GN), Levenberg-Marquardt (LM) or Trust Region(TR) algorithm is applied for the iterative parabola fitting.

6.6. Geometric Fitting of General Quadratic Curves

In sections 6.3, 6.4 and 6.5, we discussed our implicit fit based on geometric parameters. When using geometric parameters, we can only fit ellipse, hyperbola or parabola separately. In this section, we will describe our implicit fit based on algebraic parameters. Using algebraic parameters, our algorithm is then not specific to any conic type. It can easily switch between ellipse and hyperbola, even when the initial guess is a different type.

Let (x, y) be a given point and (x', y') denote its projection onto the quadratic curve $P(x, y; \Theta) = 0$, where $\Theta = (A, B, C, D, E, F)$ is the vector of algebraic parameters. Then any quadratic curve(conic) can be uniquely defined by

$$P(x, y) = Ax^2 + 2Bxy + Cy^2 + 2Dx + 2Ey + F = 0$$

up to a scale factor, as discussed in 3.2. Differentiating the identity $P(x, y; \Theta) = 0$ gives

$$(P_\Theta) = (P_A, P_B, P_C, P_D, P_E, P_F) \tag{6.30}$$

$$= (x^2, 2xy, y^2, 2x, 2y, 1) \tag{6.31}$$

And

$$P_x = 2Ax + 2By + 2D, \quad P_y = 2Bx + 2Cy + 2E.$$

Thus the derivation of Jacobian matrix is quite straightforward. Keep in mind that all the derivatives are taken at the projection points.

6.7. Benchmark Example

Once we have the Jacobian matrix, the minimization problem (6.2) can be solved by the Gauss-Newton(GN), Levenberg-Marquardt (LM) or Trust Region(TR) algorithm. The rate of converges of these algorithms is nearly quadratic provided one has a good initial guess (which can be found, for example, by a non-iterative algebraic fit, such as the Taubin fit, see Appendix A).

Ahn et al. have applied the above minimization scheme to quadratic curves (ellipses, hyperbolas, and parabolas) and some quadratic surfaces (spheres, ellipsoids, cones); see [**7, 5, 4**]. They compared it with several other minimization algorithms [**6, 5, 4**] and concluded that this one is the fastest and most robust. We have also tested it on quadratic curves and found that it has the following two advantages over other schemes: (i) it converges in fewer iterations, and (ii) it finds a smaller value of the objective function $\mathcal{F}$ more frequently than other methods do (i.e., the other methods tend to end up in a local minimum or diverge more often that this method does).

A benchmark example introduced in [**29**] and used later in [**7**] and other papers is a simple eight point set whose coordinates are shown in Table 6.1. The best fitting

x	1	2	5	7	9	3	6	8
y	7	6	8	7	5	7	2	4

Table 6.1: A benchmark example with eight points [**29**].

ellipse is known to have center $(2.6996, 3.8160)$, axes $a = 6.5187$ and $b = 3.0319$, and angle of tilt $\theta = 0.3596$; see Fig. 6.1.

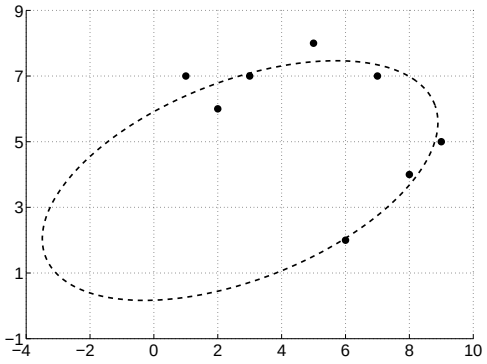

Figure 6.1: A sample of eight points and the best fitting ellipse.

We have run four fitting algorithms – the Gander-Golub-Strebel (GGS) method, which simultaneously optimizes $n+5$ parameters, as mentioned earlier; our implicit fit based on geometric parameters using the Levenberg-Marquardt method (LMG); our implicit fit based on algebraic parameters using the Levenberg-Marquardt method (LMA); our implicit fit based on algebraic parameters using the trust region method (TRA). All methods were initialized by randomly generated ellipses (to get an initial ellipse, we just picked 5 points randomly in the square $0 \leq x, y \leq 10$ and used the ellipse interpolating those 5 points). After running these fitting methods from 10^5 random initial guesses, we found that GGS method failed in 26% of the cases, LMG failed to converged to the best ellipse in 11% of the cases, while LMA and TRA always converge to the best ellipse. In those cases where all algorithms converged,

	Failure rate	Avg. iter.	Cost per iter. (flops)
GGS	26%	60	1710
LMG	11%	20	1640
LMA	0%	16	1880
TRA	0%	16	4480

Table 6.2: Comparison of four ellipse fitting methods.

our implicit fit method LMG took 20 iterations, LMA and TRA took 16 iterations, while the GGS method took 60 iterations, on the average. The cost of one iteration is comparable for the GGS method and LMG. Table 6.2 summarizes the results.

We note that a high number of iterations here is not unusual. The authors of [**29**] used a modification of the best fitting circle to initialize their GGS procedure, and it took 71 iterations (!) to converge. A coordinate-based variant of the implicit fit used in [**7**] took 19 iterations to converge (it was also initialized with the best fitting circle). Our distance-based implicit fit converged in 16 iterations.

In our experiment we used standard geometric parameters of the ellipse,i.e., $\Theta = (X_c, Y_c, a, b, \alpha)$ and also algebraic parameter $\Theta = (A, B, C, D, E, F)$. With Kepler's parameters, things get a little faster - our implicit fit (LMK) converged in 14 iterations. Table 6.3 gives the number of iterations taken by our fitting methods (the implicit method was implemented in geometric parameters (G), algebraic parameters (A) and in Kepler parameters (K)), initialized with the modified best fitting circle (* used in Table 6.3) as in [**29**] and the Taubin fit (see Appendix A).

The performance of each algorithm is illustrated by the following figures: Fig. 6.2, Fig. 6.3 and Fig. 6.4. Fig. 6.2 shows the decreasing path of the value of the objective function for four different methods (GGS, LMG, LMA, TRA) starting at the same initial guess (in this case, it's the best circle). Each dot represents one iteration. As we can see, LMA and TRA performs similarly, which are a little better than LMG.

	Initial ellipse	
	Modified Best Circle (*)	Taubin fit
GGS	50	54
LMG	17	17
LMA	17	15
TRA	16	15
LMK	14	16

Table 6.3: Comparison of five ellipse fitting methods.

But these three methods all head to the global minimum very quickly. In around 5-6 iterations, they almost reach the target. On the other hand, GGS made shorter steps in the first few iterations and moved slower than the other methods. Fig. 6.3 shows the fitted ellipses obtained by four methods after 4 iterations. Fig. 6.4 shows the fitted ellipses obtained by four methods after 8 iterations.

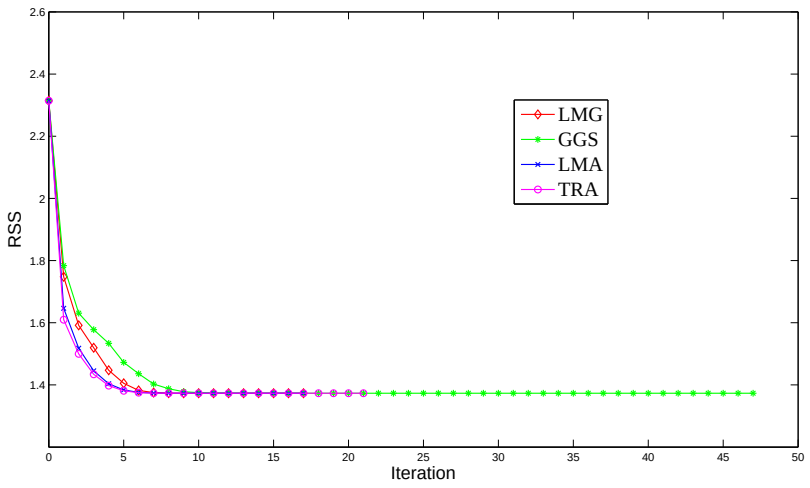

Figure 6.2: Performance of four algorithms.

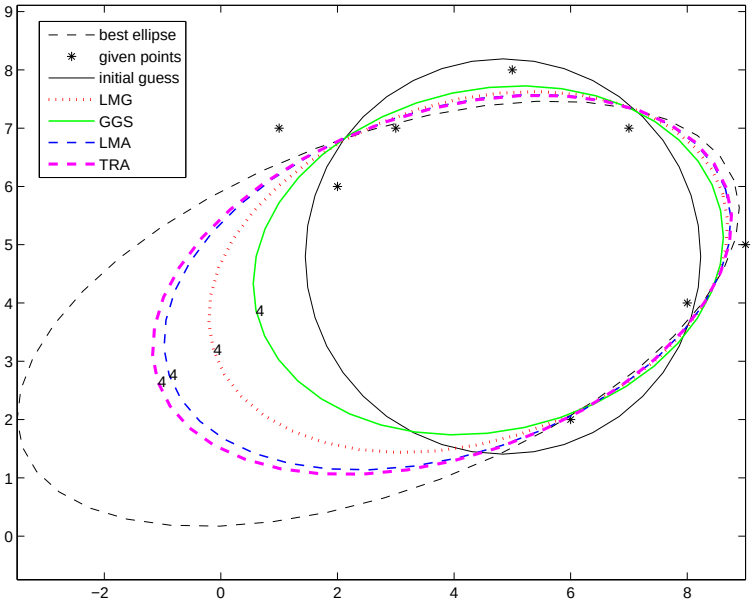

Figure 6.3: The fitted ellipses obtained by four algorithms starting at the same initial guess after 4 iterations.

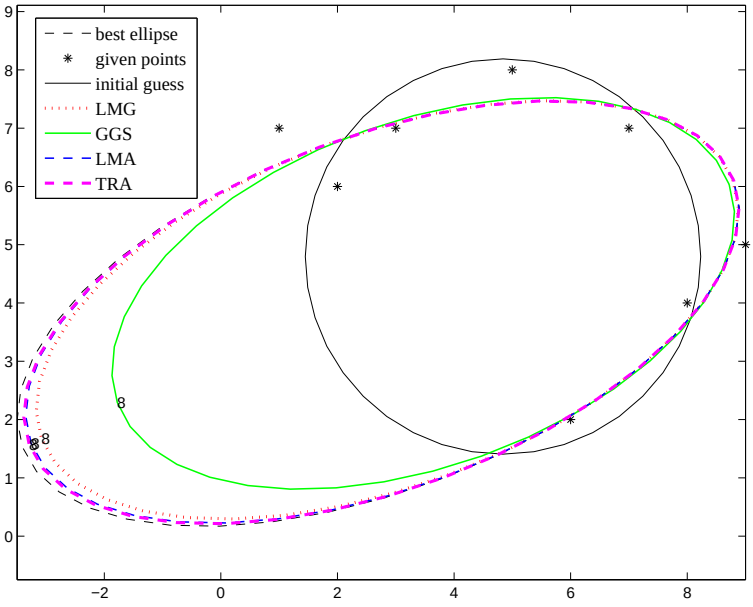

Figure 6.4: The fitted ellipses obtained by four algorithms starting at the same initial guess after 8 iterations.

Part III

CHAPTER 7

Geometric Fitting of Quadratic Surfaces

This is very recent work that is still in progress. Since many steps repeat (with little modification) what we have done in Part I and Part II for quadratic curves, this part presents only the sketch for the problem of fitting quadratic surfaces.

Fitting surfaces to observed 3D points is desired in various fields of science and engineering. In this Chapter, we begin with a theoretical review of quadratic surfaces including the parametrization scheme and classification (Sections 7.1 and 7.2). We investigat the parameter space for quadratic surfaces - the unit sphere $\mathbb{S}^9$ (Sections 7.3 and 7.4). Then we discuss methods for projecting a given set of 3D points onto quadratic surfaces (Section 7.5) and also the implicit fitting method in three-dimensional case (Section 7.6).

7.1. Quadratic Equation and Parameters

Quadratic Equation. A surface defined by an algebraic equation of degree two is called a quadric. It is given by the general equation

$$Ax^2 + By^2 + Cz^2 + 2Fyz + 2Gxz + 2Hxy + 2Px + 2Qy + 2Rz + D = 0 \tag{7.1}$$

where $A, B, C, F, G, H, P, Q, R, D$ are real numbers ("parameters" of the surface).

Algebraic Parameters. Since $A, B, C, F, G, H, P, Q, R, D$ are coefficients of a quadratic polynomial (i.e., an algebraic expression), they are often called "algebraic parameters" of the surface.

The parameter space thus will be $\mathbb{R}^{10}$, the space of all real 10-dimensional vectors with one exclusion $(0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$. Every parameter vector

$$(A, B, C, F, G, H, P, Q, R, D)$$

either represents a quadratic surface (degenerate or non-degenerate) in $\mathbb{R}^3$ or an imaginary surface which can be considered as an empty solution for 3D model. Since the quadratic surfaces are uniquely defined by this parameter scheme up to a scalar multiple, we can impose some constraints. The constraint $A^2 + B^2 + \cdots + D^2 = 1$ eliminates the multiplicity of representations. Then the (reduced) parameter space will be the unit sphere in $\mathbb{R}^{10}$. We will denote it by $\mathbb{S}^9$.

7.2. Classification of Quadratic Surfaces

Various invariants can be used in making classification of quadratic surfaces. See pages 210-211 in [**15**]. Define

$$e = \begin{vmatrix} A & H & G \\ H & B & F \\ G & F & C \end{vmatrix}, \qquad E = \begin{vmatrix} A & H & G & P \\ H & B & F & Q \\ G & F & C & R \\ P & Q & R & D \end{vmatrix}$$

$$\rho_3 = \operatorname{rank}(e), \qquad \rho_4 = \operatorname{rank}(E), \qquad \Delta = \det(E)$$

Also let k_1, k_2, k_3 be the eigenvalues of matrix e and K_1, K_2, K_3 be the eigenvalues of matrix E, then define k (or K) $= 1$ if the sign of nonzero k (or K)'s are all the same and k (or K) $= 0$ otherwise.

Table 7.1 illustrates 19 types of quadratic surfaces and their properties:

Six Nondegenerate Real Quadrics. Among all these types, there are only six non-degenerate real quadratic surfaces:

- ellipsoid (1)
- hyperboloid of one sheet (3)
- hyperboloid of two sheets (4)
- elliptic cone (5)
- elliptic paraboloid (7)
- hyperbolic paraboloid (8)

ρ_3	ρ_4	Δ	k	K	D	Type of Surface	Code
3	4	< 0	1			real ellipsoid	1
3	4	> 0	1			imaginary ellipsoid	2
3	4	> 0	0			hyperboloid of one sheet	3
3	4	< 0	0			hyperboloid of two sheets	4
3	3		0			real elliptic cone	5
3	3		1			imaginary elliptic cone	6
2	4	< 0	1			elliptic paraboloid	7
2	4	> 0	0			hyperbolic paraboloid	8
2	3		1	0		real elliptic cylinder	9
2	3		1	1		imaginary elliptic cylinder	10
2	3		0			hyperbolic cylinder	11
1	3					parabolic cylinder	12
2	2		0			real intersecting planes	13
2	2		1			imaginary intersecting planes	14
1	2			0		real parallel planes	15
1	2			1		imaginary parallel planes	16
1	1					coincident planes	17
0	2					single plane	18
0	1				$\neq 0$	poles	19

Table 7.1: Types of quadratic surfaces

Surfaces $(9), (11), (14)$ are cylinders, which we consider "partially" degenerate. Surfaces $(12), (15), (17), (18)$ are just linear planes which are completely degenerate.

7.3. Open Domains in Parameter Space

Here we begin our description of the unit sphere $\mathbb{S}^9$, which plays the role of the parameter space for surfaces.

7.3.1. Main Types of Surfaces. Each parameter vector $\mathbf{A} \in \mathbb{S}^9$ corresponds to a surface of a certain type. We recall the main types of surfaces here, grouped according to the dimensionality of the corresponding regions in $\mathbb{S}^9$.

Dimension = 9	Dimension = 8
Ellipsoid (E3D)	Elliptic cone (ECo)
Imaginary ellipsoid (IE3D)	Imaginary elliptic cone (IECo)
Hyperboloid of one sheet (1H3D)	Elliptic paraboloid (EP3D)
Hyperboloid of two sheets (2H3D)	Hyperbolic paraboloid (HP3D)
Dimension = 7	Dimension = 6, 5, 4
Elliptic cylinder (ECy)	Intersecting planes (IΠ)
Imaginary elliptic cylinder (IECy)	Imaginary intersecting planes (IIΠ)
Hyperbolic cylinder (HCy)	Parabolic cylinder (PCy)
	Parallel planes (PΠ)
	Imaginary parallel planes (IPΠ)
	Coincident planes (CΠ)
	Single plane (SΠ)

Table 7.2: Main types of surfaces grouped according to the dimensionality of the corresponding regions in $\mathbb{S}^9$

In addition, there are two poles on the sphere ("North Pole" and "South Pole") that are not listed in the above table:

$$\mathbf{P}_1 = (0,0,0,0,0,0,0,0,0,1) \quad \text{and} \quad \mathbf{P}_{-1} = (0,0,0,0,0,0,0,0,0,-1).$$

7.3.2. Partition of $\mathbb{S}^9$ into Domains. Accordingly, the unit sphere $\mathbb{S}^9$ (the parameter space) is divided into 18 domains corresponding to the above surface types, plus two extra points, the poles:

$$\mathbb{S}^9 = \mathbb{D}_{\text{E3D}} \cup \mathbb{D}_{\text{IE3D}} \cup \mathbb{D}_{\text{1H3D}} \cup \mathbb{D}_{\text{2H3D}} \cup \mathbb{D}_{\text{ECo}} \cup \mathbb{D}_{\text{IECo}} \cup \mathbb{D}_{\text{EP3D}} \cup \mathbb{D}_{\text{HP3D}} \cup \mathbb{D}_{\text{ECy}} \cup \mathbb{D}_{\text{IECy}}$$

$$\cup \mathbb{D}_{\text{HCy}} \cup \mathbb{D}_{\text{II}} \cup \mathbb{D}_{\text{III}} \cup \mathbb{D}_{\text{PCy}} \cup \mathbb{D}_{\text{PII}} \cup \mathbb{D}_{\text{IPII}} \cup \mathbb{D}_{\text{CII}} \cup \mathbb{D}_{\text{SII}} \cup \{\mathbf{P}_1\} \cup \{\mathbf{P}_{-1}\},$$

where the domains are coded by the names of the surface types, as shown in Table 7.2.

7.3.3. Open Domains of Dimension 9. Nine-dimensional domains $\mathbb{D}_{\text{E3D}}$, $\mathbb{D}_{\text{IE3D}}$, $\mathbb{D}_{\text{1H3D}}$ and $\mathbb{D}_{\text{2H3D}}$ are the most important domains which occupy a substantial part of the sphere $\mathbb{S}^9$. In fact, they are "open domains", in the topological sense. Their openness follows from the classification table given in section 7.1: they are all defined by inequality constraints. Rank $\rho_3 = 3$ and $\rho_4 = 4$ are equivalent to the conditions that the determinants of matrix e and E are not equal to zero. Then we have $\Delta > 0$ or $\Delta < 0$ which define inequalities too. The other value k is determined by the signs of the nonzero eigenvalues, $k = 1$ if signs of the nonzero eigenvalues are the same and $k = 0$ otherwise. Note that $\mathbb{D}_{\text{E3D}}$, $\mathbb{D}_{\text{IE3D}}$, $\mathbb{D}_{\text{1H3D}}$ and $\mathbb{D}_{\text{2H3D}}$ all have full rank for matrix e, so we have three nonzero eigenvalues λ_1, λ_2 and λ_3. To determine k is equivalent to determine if $\lambda_1 \cdot \lambda_2$ and $\lambda_2 \cdot \lambda_3$ is great than 0 or less than 0, so these are also inequality constraints. If an inequality holds at any point $\mathbf{A} \in \mathbb{S}^9$, it will still hold in a small vicinity of that point, as our functions ρ_3, ρ_4, Δ, k, K, etc., change continuously on the sphere $\mathbb{S}^9$. This simple observation proves the openness of the domains $\mathbb{D}_{\text{E3D}}$, $\mathbb{D}_{\text{IE3D}}$, $\mathbb{D}_{\text{1H3D}}$ and $\mathbb{D}_{\text{2H3D}}$.

7.3.4. Volumes of Open Domains. The "size" of any domain in the sphere $\mathbb{S}^9$ can be measured by its volume (more precisely, nine-dimensional volume, or Lebesgue measure). Nine-dimensional domain have positive volume. All the other domains in our partition of $\mathbb{S}^9$ have volume zero.

The volume of our domains can be estimated by an easy Monte Carlo experiment. We just generate random points on the unit sphere $\mathbb{S}^9$ and estimate the volume of each domain by the percentage of points falling into it. Below is our experimental estimates:

Type of Quadric	Volume (%)
Hyperboloid of one sheet	70.69
Hyperboloid of two sheets	26.86
Real ellipsoid	2.30
Imaginary ellipsoid	0.15
Others	0

Table 7.3: Volumes of open domains in $\mathbb{S}^9$

The volume of each open domain is given as a percentage of the total volume of $\mathbb{S}^9$. The volume of all the other domains is zero. Note that hyperboloid of two sheets and one sheet occupy more than 90% of the parameter space. Also note that "imaginary ellipsoid" occupy just a small fraction of the sphere $\mathbb{S}^9$, only 0.15% of it. This is a good feature, in terms of practical fitting methods. Those usually wander in the parameter space searching for the minimum of the objective function. Every time they accidentally run into a forbidden or an unwanted domain, such as $\mathbb{D}_{\text{IE3D}}$, they have to retreat and readjust the next step. Fortunately, due to the small size of $\mathbb{D}_{\text{IE3D}}$, this does not happen often.

7.3.5. Connected Components of Open Domains. A domain is "connected" if it consists of one piece. If the domain is not connected, it consists of several connected pieces (components). Similar to 2D cases, our open domains $\mathbb{D}_{\text{E3D}}$, $\mathbb{D}_{\text{IE3D}}$, $\mathbb{D}_{\text{1H3D}}$ and $\mathbb{D}_{\text{2H3D}}$ are *not* connected, each of them consists of exactly two connected components. This is related to the choice of sign.

Each parameter vector $\mathbf{A} = (A, B, C, F, G, H, P, Q, R, D)^T \in \mathbb{S}^9$ not only specifies a quadratic surface, but defines a quadratic function

$$Q(x, y, z) = Ax^2 + By^2 + Cz^2 + 2Fyz + 2Gxz + 2Hxy + 2Px + 2Qy + 2Rz + D$$

on the xyz space. Note that $Q(x, y, z) = 0$ "on the surface", but $Q(x, y, z) > 0$ or $Q(x, y, z) < 0$ elsewhere.

If $\mathbf{A} \in \mathbb{D}_{\text{E3D}}$, then we may have $Q > 0$ inside the ellipsoid and $Q < 0$ outside of it, or vice versa. Thus $\mathbb{D}_{\text{E3D}}$ consists of two pieces, we denote them by $\mathbb{D}^+_{\text{E3D}}$ and $\mathbb{D}^-_{\text{E3D}}$, depending on whether $Q > 0$ or $Q < 0$ inside the ellipsoid (i.e., at its center).

Similarly, if $\mathbf{A} \in \mathbb{D}_{\text{1H3D}}$ or $\mathbb{D}_{\text{2H3D}}$, then the corresponding quadratic function $Q(x, y, z)$ may be positive or negative at the center of the hyperboloid. So we have a respective partition $\mathbb{D}_{\text{1H3D}} = \mathbb{D}^+_{\text{1H3D}} \cup \mathbb{D}^-_{\text{1H3D}}$ and $\mathbb{D}_{\text{2H3D}} = \mathbb{D}^+_{\text{2H3D}} \cup \mathbb{D}^-_{\text{2H3D}}$.

Lastly, if $\mathbf{A} \in \mathbb{D}_{\text{IE3D}}$, then the corresponding quadratic function $Q(x, y, z)$ cannot take zero values, so it is either entirely positive or entirely negative. Again this causes a natural partition of $\mathbb{D}_{\text{IE3D}}$ into two pieces, $\mathbb{D}^+_{\text{IE3D}}$ and $\mathbb{D}^-_{\text{IE3D}}$.

Note that if $\mathbf{A} \in \mathbb{D}^+_{\text{E3D}}$, then $-\mathbf{A} \in \mathbb{D}^-_{\text{E3D}}$ and vice versa. Thus the subdomains $+$ and $-$ are diametrically opposite to each other on the sphere $\mathbb{S}^9$. They have identical shapes and equal volumes.

7.4. Boundaries of Open Domains

Recall that the unit sphere $\mathbb{S}^9$ (the parameter space) is divided into 18 domains corresponding to the main surface types, plus two extra points, the poles. As we discussed before, the domains $\mathbb{D}_{\text{E3D}}$, $\mathbb{D}_{\text{IE3D}}$, $\mathbb{D}_{\text{1H3D}}$ and $\mathbb{D}_{\text{2H3D}}$ are nine-dimensional, open, and they cover 100% of the sphere $\mathbb{S}^9$, in terms of volume. The other domains have lower dimensionality and zero volume. They, in a sense, make pieces of the boundaries of the principal domains $\mathbb{D}_{\text{E3D}}$, $\mathbb{D}_{\text{IE3D}}$, $\mathbb{D}_{\text{1H3D}}$ and $\mathbb{D}_{\text{2H3D}}$.

Eight-Dimensional Domains(Hypersurfaces). Most important of those "boundary pieces" are the eight-dimensional domains: $\mathbb{D}_{\text{ECo}}$ (elliptic cone), $\mathbb{D}_{\text{IECo}}$ (imaginary

elliptic cone), $\mathbb{D}_{\text{EP3D}}$ (elliptic paraboloid), $\mathbb{D}_{\text{HP3D}}$ (hyperbolic paraboloid). They separate our open domains from each other in a nine-dimensional space, see below.

7.4.1. Elliptic Cone $\mathbb{D}_{\text{ECo}}$. The hypersurface $\mathbb{D}_{\text{ECo}}$ separates the open domain hyperboloid of one sheet $\mathbb{D}_{\text{1H3D}}$ from the open domain hyperboloid of two sheets $\mathbb{D}_{\text{2H3D}}$. To illustrate this fact, consider the parameter vector

$$\mathbf{A}_c = (1, 1, -1, 0, 0, 0, 0, 0, 0, c),$$

where c is a small variable (we will not normalize $\mathbf{A}_c$ to keep our formulas simple). This parameter vector corresponds to the quadratic function

$$Q(x, y, z) = x^2 + y^2 - z^2 + c.$$

For $c < 0$, the equation $Q(x, y, z) = 0$ defines a hyperboloid of one sheet (more precisely, the surface intersects the xy-plane at a small circle of radius $\sqrt{-c}$), i.e., $\mathbf{A}_c \in \mathbb{D}_{\text{1H3D}}$. For $c = 0$, $Q(x, y, z) = x^2 + y^2 - z^2 = 0$ defines a elliptic cone, i.e., $\mathbf{A}_0 \in \mathbb{D}_{\text{ECo}}$. For $c < 0$ it is a hyperboloid of two sheets, i.e., $\mathbf{A}_c \in \mathbb{D}_{\text{2H3D}}$. As c changes from small negative values to zero and then on to small positive values, the cross-section in xy plane (circle with radius $\sqrt{-c}$) shrinks and collapses to a single point (0,0,0), and then disappears, thus making a hyperboloid of one sheet transform to a cone and then become a hyperboloid of two sheets. In the parameter space $\mathbb{S}^9$, this process corresponds to a continuous motion from the domain $\mathbb{D}_{\text{1H3D}}$ to the domain $\mathbb{D}_{\text{2H3D}}$, across the hypersurface $\mathbb{D}_{\text{ECo}}$.

7.4.2. Imaginary Elliptic Cone $\mathbb{D}_{\text{IECo}}$. The hypersurface $\mathbb{D}_{\text{IECo}}$ separates the open domain hyperboloid of one sheet $\mathbb{D}_{\text{E3D}}$ from the open domain hyperboloid of two sheets $\mathbb{D}_{\text{IE3D}}$. To illustrate this fact, consider the parameter vector

$$\mathbf{A}_c = (1, 1, 1, 0, 0, 0, 0, 0, 0, c),$$

where c is a small variable. This parameter vector corresponds to the quadratic function

$$Q(x, y, z) = x^2 + y^2 + z^2 + c.$$

For $c < 0$, the equation $Q(x, y, z) = 0$ defines a ellipsoid (more precisely, a sphere of radius $\sqrt{-c}$), i.e., $\mathbf{A}_c \in \mathbb{D}_{\text{1H3D}}$. For $c = 0$, it defines an imaginary elliptic cone $x^2 + y^2 + z^2 = 0$,i.e., $\mathbf{A}_0 \in \mathbb{D}_{\text{IECo}}$. For $c < 0$ it defines an imaginary ellipsoid, i.e., $\mathbf{A}_c \in \mathbb{D}_{\text{IE3D}}$. As c changes from small negative values to zero and then on to small positive values, the ellipsoid shrinks and collapses to a single point(imaginary elliptic cone), and then disappears altogether (transforms into an imaginary ellipsoid). In the parameter space $\mathbb{S}^9$, this process corresponds to a continuous motion from the domain $\mathbb{D}_{\text{E3D}}$ to the domain $\mathbb{D}_{\text{IE3D}}$, across the hypersurface $\mathbb{D}_{\text{IECo}}$.

7.4.3. Elliptic Paraboloids $\mathbb{D}_{\text{EP3D}}$. The hypersurface $\mathbb{D}_{\text{EP3D}}$ separates the domain hyperboloid of two sheets $\mathbb{D}_{\text{2H3D}}$ from the domain ellipsoid $\mathbb{D}_{\text{E3D}}$. To illustrate this fact, consider the parameter vector

$$\mathbf{A}_c = (1, 1, c, 0, 0, 0, 0, 0, 1, 0),$$

where c will again play the role of a small variable. This parameter vector corresponds to the quadratic function

$$Q(x, y, z) = x^2 + y^2 + cz^2 + z = x^2 + y^2 + c\Big(z + \frac{1}{2c}\Big)^2 - \frac{1}{4c}.$$

For $c < 0$, $Q(x, y, z) = 0$ defines a hyperboloid of two sheets, i.e., $\mathbf{A}_c \in \mathbb{D}_{\text{2H3D}}$. For $c = 0$, $Q(x, y, z) = x^2 + y^2 + z = 0$ is elliptic paraboloid(which opens along z-axis), i.e., $\mathbf{A}_0 \in \mathbb{D}_{\text{EP3D}}$. For $c > 0$, it is an ellipsoid, i.e., $\mathbf{A}_c \in \mathbb{D}_{\text{E3D}}$. As c changes from small negative values to zero and then on to small positive values, the hyperboloid of two sheets transforms into a elliptic paraboloid, and then into an ellipsoid. In the parameter space, this process corresponds to a continuous motion from the domain $\mathbb{D}_{\text{2H3D}}$ to the domain $\mathbb{D}_{\text{E3D}}$, across the hypersurface $\mathbb{D}_{\text{EP3D}}$.

7.4.4. Hyperbolic Paraboloids $\mathbb{D}_{\text{HP3D}}$. The hypersurface $\mathbb{D}_{\text{HP3D}}$ separates the two components $\mathbb{D}^+_{\text{1H3D}}$ and $\mathbb{D}^-_{\text{1H3D}}$ of the open domain hyperboloid of one sheet $\mathbb{D}_{\text{1H3D}}$ from each other. To illustrate this fact, consider the parameter vector

$$\mathbf{A}_c = (1, -1, c, 0, 0, 0, 0, 0, 1, 0),$$

where c will be a small variable. This parameter vector corresponds to the quadratic function

$$Q(x,y,z) = x^2 - y^2 + cz^2 + z = x^2 - y^2 + c\Big(z + \frac{1}{2c}\Big)^2 - \frac{1}{4c}.$$

The equation $Q(x,y,z) = 0$ defines a hyperboloid of one sheet with center $(0,0,-\frac{1}{2c})$, unless $c = 0$. When $c = 0$, $Q(x,y,z) = x^2 - y^2 + z$ defines a hyperbolic paraboloid. More precisely, for $c < 0$ it is a hyperboloid of one sheet with a "positive center", because $Q(0,0,-\frac{1}{2c}) > 0$, i.e., $\mathbf{A}_c \in \mathbb{D}^+_{1\mathrm{H3D}}$, in this case, the hyperboloid opens along x-axis. For $c > 0$, it is a hyperboloid with a "negative center", because $Q(0,0,-\frac{1}{2c}) < 0$, i.e., $\mathbf{A}_c \in \mathbb{D}^-_{1\mathrm{H3D}}$, in this case, the hyperboloid opens along y-axis. For $c = 0$, it is a a hyperbolic paraboloid, i.e., $\mathbf{A}_0 \in \mathbb{D}_{\mathrm{HP3D}}$. As c changes from small negative values to zero and then on to small positive values, the hyperboloid of one sheet with a positive center transforms into saddle shaped hyperbolic paraboloid and then into a hyperboloid of one sheet with a negative center. In the parameter space, this process corresponds to a continuous motion from the subdomain $\mathbb{D}^+_{1\mathrm{H3D}}$ to the subdomain $\mathbb{D}^-_{1\mathrm{H3D}}$, across the hypersurface $\mathbb{D}_{\mathrm{HP3D}}$.

7.4.5. A simplistic diagram. The above analysis can be summarized in the following schematic diagram illustrating the structure of the parameter space, with all principal subdomains and the respective separating hypersurfaces.

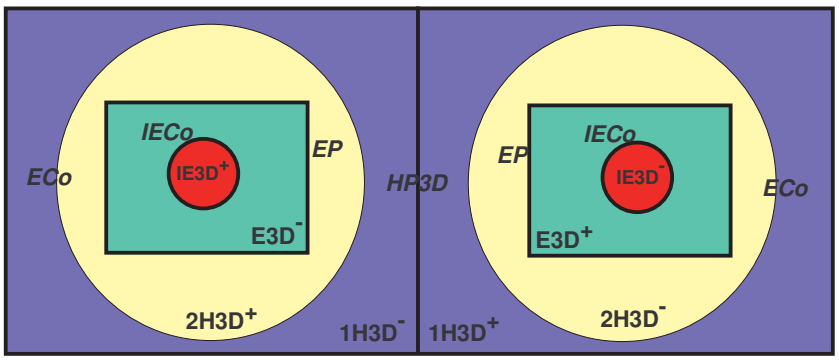

Figure 7.1: Principal domains and separating hypersurfaces in $\mathbb{S}^9$

Note: The labels correspond to our notation in the text: 1H3D$^+$ means $\mathbb{D}^+_{\text{1H3D}}$, etc.

Remark: *Continuity, singularities and differentiability of the objective function on* $\mathbb{S}^9$ *remain open problems for the future work. Next, we will proceed to the practical solutions to the problem of fitting quadratic surfaces.*

7.5. Projection onto Quadrics

Here we describe the projection of a spacial point (u, v, w) onto quadratic surfaces of various kinds.

Ellipsoids. An ellipsoid is defined its canonical coordinates as follows:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} - 1 = 0, \tag{7.2}$$

where $a \geq b \geq c > 0$ are its semiaxes. (Other ellipsoids case be translated and rotated to the canonical form (7.2), and then the projection point can be translated and rotated back to the original ellipsoid.) Due to symmetry, we restrict the method to $u > 0, v > 0, w > 0$, then we also have $x > 0, y > 0, z > 0$. The orthogonality conditions now give

$$u - x = tx/a^2, \qquad v - y = ty/b^2, \qquad w - z = tz/c^2 \tag{7.3}$$

for some scalar t, from which

$$x = \frac{a^2 u}{t + a^2}, \qquad y = \frac{b^2 v}{t + b^2} \qquad z = \frac{c^2 w}{t + c^2} \tag{7.4}$$

Since $x, y, z > 0$, we have constraint $t > \max\{-a^2, -b^2 - c^2\} = -c^2$. Substituting (7.4) into (7.2) we obtain a function

$$F(t) = \frac{a^2 u^2}{(t + a^2)^2} + \frac{b^2 v^2}{(t + b^2)^2} + \frac{c^2 w^2}{(t + c^2)^2} - 1, \tag{7.5}$$

whose root we need to find. Note that

$$\lim_{t \to -c^2+} F(t) = +\infty \qquad \text{and} \qquad \lim_{t \to \infty} F(t) = -1.$$

Taking the derivatives of F we see that

$$F'(t) = -\frac{2a^2u^2}{(t+a^2)^3} - \frac{2b^2v^2}{(t+b^2)^3} - \frac{2c^2w^2}{(t+c^2)^3} \tag{7.6}$$

and

$$F''(t) = \frac{6a^2u^2}{(t+a^2)^4} + \frac{6b^2v^2}{(t+b^2)^4} + \frac{6c^2w^2}{(t+c^2)^4}. \tag{7.7}$$

Thus on the interval $(-c^2, \infty)$ we have $F' < 0$ and $F'' > 0$, i.e., the function F is monotonically decreasing and concave, just as in Fig. 5.1. Thus standard Newton's method starting at any point t_0 where $F(t_0) > 0$ will converge to the unique root of F, and we choose (see (5.9))

$$t_0 = \max\{au - a^2, bv - b^2, cw - c^2\}.$$

Hyperbolic Paraboloids. Now let us project a point (u, v, w) onto a hyperbolic paraboloid ("saddle") defined in its canonical coordinates as

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} - z = 0. \tag{7.8}$$

Due to symmetry, we restrict the method to $u > 0, v > 0$, then we also have $x > 0, y > 0$. The orthogonality conditions now give

$$u - x = tx/a^2, \qquad v - y = -ty/b^2, \qquad w - z = -t/2 \tag{7.9}$$

for some scalar t, from which

$$x = \frac{a^2u}{t+a^2}, \qquad y = \frac{b^2v}{-t+b^2}, \qquad z = w + \frac{t}{2}. \tag{7.10}$$

Since $x, y > 0$, we have constraints $-a^2 < t < b^2$. Substituting (7.10) into (7.8) we obtain a function

$$F(t) = \frac{a^2u^2}{(t+a^2)^2} - \frac{b^2v^2}{(-t+b^2)^2} - w - \frac{t}{2}, \tag{7.11}$$

whose root we need to find. Note that

$$\lim_{t \to -a^2+} F(t) = +\infty \qquad \text{and} \qquad \lim_{t \to b^2-} F(t) = -\infty.$$

Taking the derivatives of F we see that

$$F'(t) = -\frac{2a^2u^2}{(t+a^2)^3} - \frac{2b^2v^2}{(-t+b^2)^3} - \frac{1}{2} \tag{7.12}$$

hence $F' < 0$ for all $t \in (-a^2, b^2)$. Next,

$$F''(t) = \frac{6a^2u^2}{(t+a^2)^4} - \frac{6b^2v^2}{(-t+b^2)^4}. \tag{7.13}$$

Now F'' decreases from $+\infty$ (near $-a^2$) to $-\infty$ (near b^2), and it is monotonic (because $F''' < 0$, as one can easily verify). Thus F has a unique inflection point, t_*, within the interval $(-a^2, b^2)$. Our further analysis repeats that done for hyperbolas in the previous section.

Hyperboloids. Now let us project a point (u, v, w) onto a hyperboloid (one sheet) defined in its canonical coordinates as

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} - \frac{z^2}{c^2} - 1 = 0, \tag{7.14}$$

where we can assume $a \geq b$. Due to symmetry, we restrict the method to $u > 0, v > 0, w > 0$, then we also have $x > 0, y > 0, z > 0$. The orthogonality conditions now give

$$u - x = tx/a^2, \qquad v - y = ty/b^2, \qquad w - z = -tz/c^2 \tag{7.15}$$

for some scalar t, from which

$$x = \frac{a^2u}{t+a^2}, \qquad y = \frac{b^2v}{t+b^2}, \qquad z = \frac{c^2w}{-t+c^2}. \tag{7.16}$$

Since $x, y, z > 0$, we have constraints $-b^2 < t < c^2$. Substituting (7.16) into (7.14) we obtain a function

$$F(t) = \frac{a^2u^2}{(t+a^2)^2} + \frac{b^2v^2}{(t+b^2)^2} - \frac{c^2w^2}{(-t+c^2)^2} - 1, \tag{7.17}$$

whose root we need to find. Note that

$$\lim_{t \to -b^2+} F(t) = +\infty \qquad \text{and} \qquad \lim_{t \to c^2-} F(t) = -\infty.$$

Taking the derivatives of F we see that

$$F'(t) = -\frac{2a^2u^2}{(t+a^2)^3} - \frac{2b^2v^2}{(t+b^2)^3} - \frac{2c^2w^2}{(-t+c^2)^3} \tag{7.18}$$

hence $F' < 0$ for all $t \in (-b^2, c^2)$. Next,

$$F''(t) = \frac{6a^2u^2}{(t+a^2)^4} + \frac{6b^2v^2}{(t+b^2)^4} - \frac{6c^2w^2}{(-t+c^2)^4}. \tag{7.19}$$

Again, as before, F'' decreases from $+\infty$ (near $-b^2$) to $-\infty$ (near c^2), and it is monotonic (because $F''' < 0$, as one can easily verify). Thus F has a unique inflection point, t_*, within the interval $(-b^2, c^2)$. Its graph looks like one of those shown in Fig. 5.2.

But now it is not easy to determine which case we have at hand – the one shown in part (a) or in part (b) of Fig. 5.2 (because we cannot solve equation $F'' = 0$). However, one of the two iterative procedures described in the case of hyperbolas (i.e., Newton's method working from the left and another one – from the right), must work.

Thus we simply can choose one of these two procedures at random and follow it hoping that it converges. But if it fails, i.e., if an iteration lands outside the interval $(-b^2, c^2)$, then we switch to the other procedure, and it will surely converge. We note that even if we start on the wrong side, Newton's iteration may land on the right side and then converge.

As there is a 50% chance of choosing one of the two sides correctly at random, the current projection method is perhaps about 1.5 times slower, on average, than the previous one (a moderate price to pay for extra complications). We emphasize that our analysis still guarantees that the method converges to the correct projection point in all cases.

Other Quadrics. We have covered three major types of quadratic surfaces in 3D. There are two others – elliptic paraboloid and hyperboloid of two sheets, which are treated very similarly, with small variations in each case that we leave out.

7.6. Geometric Surfaces Fit

The implicit fit method we derived in Chapter 6 works for surfaces in the 3D space too, when they are defined by the implicit equations $P(x, y, z; \Theta) = 0$. In this case, the formulas we derived acquire an extra term corresponding to the z variable.

Proposition. *Let (x, y, z) be a given a point and (x', y', z') denote its projection onto the surface $P(x, y, z; \Theta) = 0$ (then x', y', z' depend on Θ). Denote $g = x - x'$, $h = y - y'$, $k = z - z'$ and $d^2 = g^2 + h^2 + k^2$. Then we have*

$$g_\Theta = \frac{P_\Theta P_x - h(P_x P_{y\Theta} - P_y P_{x\Theta}) - k(P_x P_{z\Theta} - P_z P_{x\Theta})}{P_x^2 + P_y^2 + P_z^2}, \tag{7.20}$$

$$h_\Theta = \frac{P_\Theta P_y - g(P_y P_{x\Theta} - P_x P_{y\Theta}) - k(P_y P_{z\Theta} - P_z P_{y\Theta})}{P_x^2 + P_y^2 + P_z^2}, \tag{7.21}$$

$$k_\Theta = \frac{P_\Theta P_z - g(P_z P_{x\Theta} - P_x P_{z\Theta}) - h(P_z P_{y\Theta} - P_y P_{z\Theta})}{P_x^2 + P_y^2 + P_z^2}, \tag{7.22}$$

and

$$d_\Theta = \frac{P_\Theta}{\sqrt{P_x^2 + P_y^2 + P_z^2}}, \tag{7.23}$$

where P_Θ, P_x, P_y, P_z denote the first order partial derivatives of P with respect to Θ, x, y, z, respectively, and $P_{x\Theta}$, $P_{y\Theta}$ and $P_{z\Theta}$ the corresponding second order partial derivatives; all the derivatives are taken at the projection point (x', y', z').

Proof. Since the vector $(x - x', y - y', z - z')$ is orthogonal to the curve,

$$g = x - x' = tP_x \qquad h = y - y' = tP_y \qquad \text{and} \qquad k = z - z' = tP_z \tag{7.24}$$

for some scalar t. This immediately gives

$$d^2 = g^2 + h^2 + k^2 = t^2(P_x^2 + P_y^2 + P_z^2). \tag{7.25}$$

Next we use differentiation with respect to Θ. Differentiating the identity $P(x', y', z'; \Theta) = 0$ gives

$$P_\Theta = -P_x x'_\Theta - P_y y'_\Theta - P_z z'_\Theta = (g g_\Theta + h h_\Theta + k k_\Theta)/t, \tag{7.26}$$

and differentiating the identity $d^2 = g^2 + h^2 + k^2$ gives

$$dd_\Theta = gg_\Theta + hh_\Theta + kk_\Theta = tP_\Theta. \tag{7.27}$$

Now (7.23) follows from (7.27) and (7.25). Differentiation of (7.24) gives

$$g_\Theta = t_\Theta P_x + tP_{x\Theta}, \qquad h_\Theta = t_\Theta P_y + tP_{y\Theta}, \qquad k_\Theta = t_\Theta P_z + tP_{z\Theta}.$$

Eliminating t_Θ from these three equations yields

$$\frac{g_\Theta - tP_x\Theta}{P_x} = \frac{h_\Theta - tP_y\Theta}{P_y} = \frac{k_\Theta - tP_z\Theta}{P_z}. \tag{7.28}$$

Solving (7.27) and (7.28) for g_Θ, h_Θ and k_Θ we obtain (7.20), (7.21) and (7.22). □

Formula (7.23) gives the Jacobian matrix similar to (6.20). Then standard algorithms such as Gauss-Newton(GN), Levenberg-Marquardt (LM) or Trust Region(TR) can be applied for the iterative fitting process.

CHAPTER 8

Thesis Contributions and Conclusions

Fitting simple contours (primitives) such as circles, ellipses or other quadratic curves to observed image data in the plane is a problem that arises in many application areas. In 3D space, one often fits planes, spheres, or more complex surfaces (such as ellipsoids) to point clouds. In the past, people consider the problem of fitting ellipses and other quadratic curves/surfaces by minimizing geometric distances to be a prohibitively difficult task. The rigorous analysis and investigation presented in this thesis provide a strong argument against this customary presumption. In this section, the thesis's main contributions and conclusions are presented:

- The collection of ellipses is not a sufficient model for fitting purposes. This means that there is a real chance that for a given set of data points no ellipse can be selected as the best fit to the points, i.e., the ellipse fitting problem would have no solution. In the larger framework of fitting all quadratic curves, the best fitting object will always be found, either an ellipse or a hyperbola. Therefore we treat the problem of fitting an ellipse to data points as a part of a more general problem of fitting quadratic curves. In fact, with this understanding, we were able to find more efficient practical algorithms for fitting ellipses to data.
- Algebraic parametrization scheme should be chosen over geometric parametrization. Algebraic parameters allow us to reach all types of quadratic curves which could be switched back and forth during the fitting process. In addition, the boundedness of their parameter space enforces the convergence of the iterative fitting algorithm.

- Ellipses $\mathbb{D}_{\mathrm{E}}$, hyperbolas $\mathbb{D}_{\mathrm{H}}$, parabolas $\mathbb{D}_{\mathrm{P}}$, intersecting lines $\mathbb{D}_{\mathrm{IL}}$ and parallel lines $\mathbb{D}_{\mathrm{PL}}$ are the domains where the minimization algorithms are likely to maneuver searching for the best fitting conic and where the best fit can be found. All the other parts of the parameter space $\mathbb{S}^5$ can be ignored for the purpose of minimization of the objective function. On those parts the objective function $\mathcal{F}$ is either not defined or tends to grow.
- We investigated the local minima of the objective function $\mathcal{F}$. For more than five data points, local minima are possible. When data points are sampled along the entire ellipse, or an elliptic arc with high curvature, the objective function tends to have one global minimum and no local minima. When data points are sampled along an elliptic arc with low curvature, the desirable fit is given by an elliptic arc or, occasionally, by a hyperbolic arc. Local minima of $\mathcal{F}$ which correspond to distractive fits tend to be small and narrow, so that the chance of falling into one of them is low. On the contrary, the desirable fit tends to correspond to a wide minimum of $\mathcal{F}$, which is likely attract the minimization procedures starting even from a randomly chosen initial quadratic curve.
- Eberly discovered a remarkably fast and totally reliable projection algorithm for ellipses which we generalized to all the other quadratic curves and surfaces and provided a theoretical proof of convergence in each case. Ahn et al. have designed a general fitting scheme for implicit curves and surfaces that is surprisingly simple and fast. Our implicit fitting algorithms are based on his distance-based scheme where one treats $\mathcal{F}$ as a sum of n squares. By combining projection and minimization steps together, we gave a complete, reliable and efficient geometric fitting scheme for fitting quadratic curves and surfaces of all kinds.

Bibliography

[1] Geometric Product Specification (GPS) Acceptance and Switzerland (2001) representation test for coordinate measuring machines (CMM) Part 6: Estimation of errors in computing Gaussian associated features. Int'l Standard ISO 10360-6. IS, Geneva, 2001.

[2] R. J. Adcock. Note on the method of least squares. *Analyst, London*, 4:183–184, 1877.

[3] R. J. Adcock. A problem in least squares. *Analyst, London*, 5:53–54, 1878.

[4] S. J. Ahn. *Least Squares Orthogonal Distance Fitting of Curves and Surfaces in Space*. Springer, 2004.

[5] S. J. Ahn, W. Rauh, H. Suck Cho, and H. J. Warnecke. Orthogonal distance fitting of implicit curves and surfaces. *IEEE Trans.*, 24:620–638, 2002.

[6] S. J. Ahn, W. Rauh, and Hand M. Recknagel. Least squares orthogonal distances fitting of implicit curves and surfaces. *Pattern Recognition*, 2191:398–405, 2001.

[7] S. J. Ahn, W. Rauh, and H. J. Warnecke. Least-squares orthogonal distances fitting of circle, sphere, ellipse, hyperbola, and parabola. *Pattern Recog.*, 34:2283–2303, 2001.

[8] M. Aigner and B. Jüttler. Robust computation of foot points on implicitly defined curves. In *in: Mathematical Methods for Curves and Surfaces: Troms 2004*, pages 1–10. Nashboro Press, 2005.

[9] M. Aigner and B. Jüttler. Gauss-newton type techniques for robustly fitting implicitly defined curves and surfaces to unorganized data points. In *in: Shape Modeling International*, pages 121–130, 2008.

[10] A. Albano. Representation of digitized contours in terms of conic arcs and straight-line segments. *Computer Graphics and Image Processing*, 3:23–33, 1974.

[11] A. Atieg and G. A. Watson. Fitting circular arcs by orthogonal distance regression. *Appl. Numer. Anal. Comput. Math.*, 1:66–76, 2004.

[12] M. Berman. Estimating the parameters of a circle when angular differences are known. *Appl. Statist.*, 32:1–6, 1983.

[13] R. H. Biggerstaff. Three variations in dental arch form estimated by a quadratic equation. *J. Dental Res.*, 51:1509, 1972.

[14] F. L. Bookstein. Fitting conic sections to scattered data. *Computer Graphics and Image Processing*, 9:56–71, 1979.

[15] W. H. Breyer. *CRC Standard Mathematical Tables and Formulas*. CRC Press, 1987.

[16] N. N. Chan. On circular functional relationships. *J. R. Statist. Soc. B*, 27:45–56, 1965.

[17] N. Chernov. On the convergence of fitting algorithms in computer vision. *J. Math. Imag. Vision*, 27:231–239, 2007.

[18] N. Chernov. *Circular and Linear Regression: Fitting Circles and Lines by Least Squares*. Chapman and Hall, 2010.

[19] N. Chernov and C. Lesort. Statistical efficiency of curve fitting algorithms. *Comp. Stat. Data Anal.*, 47:713–728, 2004.

[20] N. Chernov and H. Ma. Least squares fitting of quadratic curves and surfaces. *Computer Vision*, pages 285–302, 2011.

[21] N. Chernov, G. Ososkov, and I. Silin. Robust fitting of ellipses to non-complete and contaminated data. *Czech. J. Phys.*, 50:347–354, 2000.

[22] N. Chernov and P. Sapirstein. Fitting circles to data with correlated noise. *Comput. Statist. Data Anal.*, 52:5328–5337, 2008.

[23] W. Chojnacki, M. J. Brooks, and A. van den Hengel. Rationalising the renormalisation method of Kanatani. *J. Math. Imaging & Vision*, 14:21–38, 2001.

[24] W. Chojnacki, M. J. Brooks, A. van den Hengel, and D. Gawley. FNS, CFNS and HEIV: A unifying approach. *J. Math. Imaging Vision*, 23:175–183, 2005.

[25] Y. Cui, J. Weng, and H. Reynolds. Estimation of ellipse parameters using optimal minimum variance estimator. *Pattern Recognition*, 17:309–316, 1996.

[26] D. Eberly. *3D Game Engine Design, 2nd ed.* Morgan Kaufmann Publishers, 2007.

[27] A. W. Fitzgibbon, M. Pilu, and R. B. Fisher. Direct least squares fitting of ellipses. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 21:476–480, 1999.

[28] P. R. Freeman. Note: Thom's survey of the Avebury ring. *J. Hist. Astronom.*, 8:134–136, 1977.

[29] W. Gander, G. H. Golub, and R. Strebel. Least squares fitting of circles and ellipses. *BIT*, 34:558–578, 1994.

[30] R. Halíř and J. Flusser. Numerically stable direct least squares fitting of ellipses. In *Sixth Conf. Cent. Europe Comput. Graph. Vis., WSCG'98, Conf. Proc.*, volume 1, pages 125–132, Plzen, Czech Rep., 1998.

[31] R. Halíř and Ch. Menard. Diameter estimation for archaeological pottery using active vision. In Axel Pinz, editor, *Proc. 20th Workshop Austrian Assoc. Pattern Recognition (ÖAGM'96)*, pages 251–261, Schloss Seggau, Leibnitz, 1996.

[32] http://www.math.uab.edu/ chernov/cl.

[33] http://www.math.uab.edu/ chernov/cl/conics.

[34] K. Kanatani. Statistical bias of conic fitting and renormalization. *IEEE Trans. Pattern Analysis Machine Intelligence*, 16:320–326, 1994.

[35] K. Kanatani. Statistical optimization for geometric fitting: Theoretical accuracy bound and high order error analysis. *Int. J. Computer Vision*, 80:167–188, 2008.

[36] C. H. Kummell. Reduction of observation equations which contain more than one observed quantity. *Analyst, London*, 6:97–105, 1879.

[37] U. M. Landau. Estimation of a circular arc center and its radius. *CVGIP: Image Understanding*, 38:317–326, 1987.

[38] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quart. Appl. Math.*, 2:164–168, 1944.

[39] D. Marquardt. An algorithm for least squares estimation of nonlinear parameters. *SIAM J. Appl. Math.*, 11:431–441, 1963.

[40] J. J. Moré, B. B. Garbow, and K. E. Hillstrom. User guide for MINPACK-1. Technical Report ANL-80-74, Argonne National Lab., IL (USA), 1980.

[41] G. Ososkov, I. Silin, and N. Chernov. Robust fitting of ellipses to non-complete and contaminated data. *Czech. J. Physics*, 50:347–354, 2000.

[42] K. Paton. Conic sections in chromosome analysis. *Pattern Recogn.*, 2:39–51, 1970.

[43] S.-C. Pei and J.-H. Horng. Optimum approximation of digital planar curves using circular arcs. *Pattern Recogn.*, 29:383–388, 1996.

[44] P. L. Rosin. Assessing error of fit functions for ellipses. *Graphical Models Image Process*, 58:494–502.

[45] P. L. Rosin. A note on the least squares fitting of ellipses. *Pattern Recognition Letters*, 14:799–808, 1993.

[46] P. L. Rosin and G. A. W. West. Segmentation of edges into lines and arcs. *Image Vision Comp.*, 7:109–114, 1989.

[47] R. Safaee-Rad, I. Tchoukanov, B. Benhabib, and K. C. Smith. Accurate parameter estimation of quadratic curves from grey-level images. *CVGIP: Image Understanding*, 54:259–274, 1991.

[48] P. D. Sampson. Fitting conic sections to very scattered data: an iterative refinement of the Bookstein algorithm. *Comp. Graphics Image Proc.*, 18:97–108, 1982.

[49] B. Sarkar, L. K. Singh, and D. Sarkar. Approximation of digital curves with line segments and circular arcs using genetic algorithms. *Pattern Recogn. Letters*, 24:2585–2595, 2003.

[50] E. Saund. Identifying salient circular arcs on curves. *CVGIP: Image Understanding*, 58:327–337, 1993.

[51] H. Späth. Least-squares fitting by circles. *Computing*, 57:179–185, 1996.

[52] H. Späth. Least squares fitting of ellipses and hyperbolas. *Comput. Stat.*, 12:329–341, 1997.

[53] H. Späth. Orthogonal distance fitting by cicles and ellipses with given area. *Comput. Stat.*, 12:343–354, 1997.

[54] H. Späth. Orthogonal least squares fitting by conic sections. In *Recent Advances in Total Least Squares techniques and Errors-in-Variables Modeling*, pages 259–264. SIAM, 1997.

[55] P. Sturm and P. Gargallo. Conic fitting using the geometric distance. *Proc. Asian Conf. Comp. Vision*, 2:784–795, 2007.

[56] G. Taubin. Estimation of planar curves, surfaces and nonplanar space curves defined by implicit equations, with applications to edge and range image segmentation. *IEEE Trans. Pattern Analysis Machine Intelligence*, 13:1115–1138, 1991.

[57] A. Thom. A statistical examination of the megalithic sites in britain. *J. Royal Statist. Soc. A*, 118:275–295, 1955.

[58] A. Thom and A. S. Thom. A megalithic lunar observatory in Orkney: the ring of Brogar and its cairns. *J. Hist. Astronom.*, 4:111–123, 1973.

[59] A. Thom, A. S. Thom, and T. R. Foord. Avebury (1): a new assessment of the geometry and metrology of the ring. *J. Hist. Astronom.*, 7:183–192, 1976.

[60] M. E. Whalen. Ceramic vessel size estimation from sherds: An experiment and a case study. *J. Field Archaeology*, 25:219–227, 1998.

[61] Papliński A. & Esson Wijewickrema, S. Orthogonal distance fitting revisited. *Tech. report, Clayton School Inf. Technol.*, page 205, 2006.

[62] www.geom.uiuc.edu/docs/reference/CRC formulas/node28.html.

[63] E. Zelniker and V. Clarkson. A statistical analysis of the Delogne-Kåsa method for fitting circles. *Digital Signal Proc.*, 16:498–522, 2006.

[64] Z. Zhang. Parameter estimation techniques: A tutorial with application to conic fitting. *Inter. J. Image & Vision Comp.*, 15:59–76, 1997.

APPENDIX A

Algebraic Fits

Here we review non-geometric (algebraic) fits that are used to provide an initial guess, i.e., a curve that initializes an iterative procedure solving the geometric fitting problem (6.2).

Suppose again that one fits an implicit curve $P(x, y; \Theta) = 0$ to observed points (x_1, y_1), ..., (x_n, y_n). Perhaps the simplest non-geometric fit is the one minimizing

$$\mathcal{F}_1(\Theta) = \sum_{i=1}^{n} [P(x_i, y_i; \Theta)]^2. \tag{A.1}$$

To justify this method one usually notes that $P(x_i, y_i; \Theta) = 0$ if and only if the point (x_i, y_i) lies on the curve, and $[P(x_i, y_i; \Theta)]^2$ is small when the point lies near the curve. The minimization of (A.1) is called *algebraic fit* and $|P(x_i, y_i; \Theta)|$ is called the corresponding *algebraic distance*.

When the curve is defined by an algebraic equation, such as

$$Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0, \tag{A.2}$$

then an unconstrained minimization of (A.1) produces the unwanted degenerate solution: $A = B = C = D = E = F = 0$. To avoid it, one can impose a constraint, such as $A^2 + B^2 + C^2 + D^2 + E^2 + F^2 = 1$. The resulting fit would not be invariant under rotations or translations of the data set, i.e., the resulting curve would depend on the choice of the coordinate system, which is hardly acceptable; see [**14, 29**].

Better constraints (i.e., those that are invariant under translations and rotations) are $A+C = 1$ (see [**29**]), or $A^2+B^2/2+C^2 = 1$ (see [**14**]), or $4AC-B^2 = 1$ (see [**27**]). The last constraint guarantees that the resulting curve will be an ellipse (rather than a hyperbola or parabola), see 3.1.

But practical experience shows that all algebraic fits, with or without constraints, are statistically inaccurate and biased, in one way or another. The main reason is that algebraic distances may be substantially different from geometric distances [**64**].

This defect can be compensated for by using linear approximation

$$\frac{|P(x_i, y_i; \Theta)|}{\|\nabla P(x_i, y_i; \Theta)\|} = d_i + \mathcal{O}(d_i^2)$$

where $\nabla P = \big(\partial P/\partial x, \partial P/\partial y\big)$ denotes the gradient vector. This leads to an 'approximate geometric fit', which minimizes

$$\mathcal{F}_2(\Theta) = \sum_{i=1}^{n} \frac{[P(x_i, y_i; \Theta)]^2}{\|\nabla P(x_i, y_i; \Theta)\|^2}. \tag{A.3}$$

This method is called *gradient weighted algebraic fit*. It was applied to quadratic curves by Sampson [**48**] and popularized by Taubin [**56**].

If the curve is defined by an algebraic equation, such as (A.2), then both numerator and denominator of each fraction in (A.3) are homogeneous quadratic polynomials of the parameters. As a result, $\mathcal{F}_2$ is invariant under scalings of the parameter vector (A, B, C, D, E, F), hence no additional constraints are needed anymore. The minimization of (A.3) produces a more accurate fit than the simple algebraic fits (A.1) do; see statistical analysis in [**19**].

But the problem of minimizing (A.3) is that it has no closed form solution and must be solved by iterations. Various iterative schemes for the minimization of (A.3) have been developed (see [**23, 24, 34**]); some of them have become standard in computer vision industry. They are all too complex to be used for an initialization of the geometric fit (6.2) (besides, each of them needs its own initialization to get started...). In this sense, the minimization of (A.3) and that of (6.2) can be regarded as two independent approaches to the task of fitting curves to data. While (6.2) is called Maximum Likelihood Estimation (MLE), that of (A.3) is called Approximate Maximum Likelihood Estimation (AMLE); see [**23, 24**].

One may wonder if the AMLE (A.3) can be used *instead of* the MLE (6.2), as the minimization of (A.3) is technically simpler than that of (6.2). Most researchers,

however, agree that the answer is NO, i.e., the minimization of (6.2) would produce a better fit than that of (A.3); see comments in [**4**, p. 12]. (Though a complete statistical analysis has yet to be done.)

Taubin [**56**] simplified (A.3) and converted it into a non-iterative fit that minimizes

$$\mathcal{F}_3(\Theta) = \frac{\sum [P(x_i, y_i; \Theta)]^2}{\sum \|\nabla P(x_i, y_i; \Theta)\|^2}. \tag{A.4}$$

Note that Taubin simply summed up all the numerators and all the denominators in (A.3) separately.

If one fits conics defined by algebraic equation (A.2), then both numerator and denominator of (A.4) are homogeneous quadratic polynomials of the components of the parameter vector $\mathbf{A} = (A, B, C, D, E, F)^T$. Thus one can rewrite (A.4) as

$$\mathcal{F}_4(\mathbf{A}) = \frac{\mathbf{A}^T\mathbf{M}\mathbf{A}}{\mathbf{A}^T\mathbf{N}\mathbf{A}} \quad \rightarrow \quad \min, \tag{A.5}$$

where $\mathbf{M}$ and $\mathbf{N}$ are some 6×6 symmetric positive semi-definite matrices. Since $\mathcal{F}_4(\mathbf{A})$ is invariant under scalings of the vector $\mathbf{A}$, one can solve (A.5) by minimizing $\mathcal{F}_5(\mathbf{A}) = \mathbf{A}^T\mathbf{M}\mathbf{A}$ under the constraint $\mathbf{A}^T\mathbf{N}\mathbf{A} = 1$. Introducing a Lagrange multiplier η we can minimize the function

$$\mathcal{F}_6(\mathbf{A}, \eta) = \mathbf{A}^T\mathbf{M}\mathbf{A} - \eta(\mathbf{A}^T\mathbf{N}\mathbf{A} - 1).$$

Differentiating with respect to $\mathbf{A}$ gives the first order necessary condition

$$\mathbf{M}\mathbf{A} = \eta\mathbf{N}\mathbf{A}, \tag{A.6}$$

thus $\mathbf{A}$ must be a generalized eigenvector of the matrix pair $(\mathbf{M}, \mathbf{N})$. Moreover, pre-multiplying (A.6) by $\mathbf{A}$ we see that $\mathbf{A}^T\mathbf{M}\mathbf{A} = \eta$, and because we are minimizing $\mathbf{A}^T\mathbf{M}\mathbf{A}$, the desired vector $\mathbf{A}$ must correspond to the *smallest* (non-negative) eigenvalue η. (We note that $\mathbf{N}$ here is singular; one usually eliminates F to reduce $\mathbf{A}$ to a 5-vector $\mathbf{A}' = (A, B, C, D, E)^T$ and the 6×6 problem (A.6) to a 5×5 problem $\mathbf{M}'\mathbf{A}' = \eta'\mathbf{N}'\mathbf{A}'$, where the 5×5 matrix $\mathbf{N}'$ is positive definite; see details in [**35**].)

Solving a generalized eigenvalue problem takes just one call of a standard matrix function (such functions are included in most modern software packages, e.g., in MATLAB). Thus Taubin's fit is regarded as a fast non-iterative procedure. In practice the Taubin's fit is only marginally slower than the simple algebraic fit minimizing (A.1).

Its advantage is that Taubin's fit is more balanced than any algebraic fit. It has a much smaller bias; see statistical analysis in [**35**]. It is important to note that it produces a conic that may be of any kind – an ellipse, a hyperbola, or a parabola. If one fits conics of a certain type (e.g., ellipses), then Taubin's fit must be supplemented with another simple fit whenever it gives the wrong curve. Experimental tests show that Taubin's fit provides a better initialization of iterative procedures than simple algebraic fits do [**17**].

APPENDIX B

Minimization Schemes

This appendix is borrowed from N. Chernov's book, see page 70-78 in [**18**].

B.1. Classical minimization schemes

We begin with a brief description of general numerical schemes used to minimize smooth functions of several variables, especially those adopted to least squares problems. First we recall two classical algorithms that are a part of any standard numerical analysis course.

B.1.1. Steepest descent. Suppose we need to find the minimum of a smooth function $\mathcal{G}\colon \mathbb{R}^k \to \mathbb{R}$, i.e.

$$z = \mathcal{G}(\mathbf{a}), \qquad \mathbf{a} = (a_1, \dots, a_k) \in \mathbb{R}^k \tag{B.1}$$

of k variables. Iterative procedures usually compute a sequence of points $\mathbf{a}^{(0)}$, $\mathbf{a}^{(1)}$, ... that presumably converges to a point where $\mathcal{G}$ takes its minimum value. The starting point $\mathbf{a}^{(0)}$ (the *initial guess*) is assumed to be chosen somehow, and the procedure follows a certain rule to determine $\mathbf{a}^{(i+1)}$ given $\mathbf{a}^{(i)}$.

That is, to define a procedure, it suffices to describe the rule of constructing the next approximation, $\mathbf{a}'$, from the current approximation, $\mathbf{a}$.

We always assume that the derivatives of the function $\mathcal{G}$ can be evaluated; hence one can find the gradient vector $\nabla\mathcal{G}(\mathbf{a})$, and then the most logical move from $\mathbf{a}$ would be in the direction opposite to $\nabla\mathcal{G}(\mathbf{a})$, where the function $\mathcal{G}$ decreases most rapidly. This method is called the *steepest descent*. It can be described by a formula

$$\mathbf{a}' = \mathbf{a} - \eta\,\nabla\mathcal{G}(\mathbf{a}),$$

where $\eta > 0$ is a factor. The choice of η is based on the following general considerations.

B.1.2. Choosing the step length. . If η is too large, one may 'overstep' the region where $\mathcal{G}$ takes small values and land too far. If η is too small, the progress will be slow, but at least the function will decrease, i.e. one gets $\mathcal{G}(\mathbf{a}') < \mathcal{G}(\mathbf{a})$.

The simplest approach is to set $\eta = 1$, compute $\mathbf{a}'$, and then check if it is acceptable. If $\mathcal{G}(\mathbf{a}') < \mathcal{G}(\mathbf{a})$, the value $\mathbf{a}'$ is accepted, otherwise one 'backtracks' by trying smaller values of η (for instance, $\eta = 1/2$, then $\eta = 1/4$, etc.) until $\mathbf{a}'$ is acceptable.

Generally, the steepest descent is a reliable method, but it usually converges slowly (at best, linearly).

Newton-Raphson method. If the second derivatives of $\mathcal{G}$ are available, one can compute both the gradient vector and the Hessian matrix of the second order partial derivatives:

$$\mathbf{D} = \nabla \mathcal{G}(\mathbf{a}) \qquad \text{and} \qquad \mathbf{H} = \nabla^2 \mathcal{G}(\mathbf{a}) \tag{B.2}$$

and approximate $\mathcal{G}$ in a vicinity of $\mathbf{a}$ by the quadratic part of its Taylor polynomial:

$$\mathcal{G}(\mathbf{a} + \mathbf{h}) \approx \mathcal{G}(\mathbf{a}) + \mathbf{D}^T \mathbf{h} + \tfrac{1}{2} \mathbf{h}^T \mathbf{H} \mathbf{h}, \tag{B.3}$$

where $\mathbf{h} = \mathbf{a}' - \mathbf{a}$ denotes the step. Now one can choose $\mathbf{h}$ as the critical point of this quadratic approximation, i.e. find $\mathbf{h}$ by solving

$$\mathbf{D} + \mathbf{H}\mathbf{h} = 0. \tag{B.4}$$

This is the Newton-Raphson method.

It converges fast (quadratically) if the current iteration is already close enough to the minimum of $\mathcal{G}$. However, this method may run into various problems. First, just as the steepest descent, it may 'overstep' the region where $\mathcal{G}$ takes small values and land too far, then one has to 'backtrack'. Second, the matrix $\mathbf{H}$ may not be positive-definite, then the quadratic approximation in (B.3) will not even have a minimum:

the solution of (B.4) will be a saddle point or a maximum. In that case the quadratic approximation in (B.3) seems to be quite useless.

Fortunately, the least squares problems allow an efficient way to get around the last trouble, see the next section.

B.2. Gauss-Newton method

B.2.1. Least squares problem. Consider a problem in which we are to minimize a function

$$\mathcal{G}(\mathbf{a}) = \sum_{i=1}^{n} g_i^2(\mathbf{a}) \tag{B.5}$$

of k variables $\mathbf{a} = (a_1, \ldots, a_k)$. We assume that $n > k$, and g_i have derivatives.

B.2.2. Newton-Raphson approach to the least squares problem. . As in Newton-Raphson scheme, we start by approximating $\mathcal{G}(\mathbf{a}+\mathbf{h})$ by a quadratic part of Taylor polynomial:

$$\mathcal{G}(\mathbf{a}+\mathbf{h}) \approx \mathcal{G}(\mathbf{a}) + \mathbf{D}^T\mathbf{h} + \tfrac{1}{2}\,\mathbf{h}^T\mathbf{H}\mathbf{h}. \tag{B.6}$$

where

$$\mathbf{D} = \nabla\mathcal{G}(\mathbf{a}) = 2\sum_{i=1}^{n} g_i(\mathbf{a})\nabla g_i(\mathbf{a}) \tag{B.7}$$

is the gradient of $\mathcal{G}$ and

$$\mathbf{H} = \nabla^2\mathcal{G}(\mathbf{a}) = 2\sum_{i=1}^{n}[\nabla g_i(\mathbf{a})][\nabla g_i(\mathbf{a})]^T + 2\sum_{i=1}^{n} g_i(\mathbf{a})\nabla^2 g_i(\mathbf{a}) \tag{B.8}$$

is the Hessian matrix of $\mathcal{G}$. The Newton-Raphson method (B.4) uses both $\mathbf{D}$ and $\mathbf{H}$.

B.2.3. Gauss-Newton for the least squares problem. The Gauss-Newton method drops the last sum of (B.8), i.e. it replaces $\mathbf{H}$ with

$$\mathbf{H}^{\diamond} = 2\sum_{i=1}^{n}[\nabla g_i(\mathbf{a})][\nabla g_i(\mathbf{a})]^T. \tag{B.9}$$

To justify this replacement, one usually notes that in typical least squares applications $g_i(\mathbf{a})$ are small, hence the second sum in (B.8) is much smaller than the first, so its

removal will not change the Hessian matrix $\mathbf{H}$ much. Also one notes that modifying $\mathbf{H}$ cannot alter the limit point of the procedure, it can only affect the path that the iterations take to approach that limit[1].

B.2.4. Advantages. Replacing $\mathbf{H}$ with $\mathbf{H}^\diamond$ has two immediate advantages:

(a) The computation of second order derivatives of g_i is no longer necessary;

(b) Unlike $\mathbf{H}$, the new matrix $\mathbf{H}^\diamond$ is always positive semi-definite.

Assume for a moment that $\mathbf{H}^\diamond$ is nonsingular. Then the quadratic expression

$$\mathcal{G}(\mathbf{a}) + \mathbf{D}^T\mathbf{h} + \tfrac{1}{2}\,\mathbf{h}^T\mathbf{H}^\diamond\mathbf{h}$$

has a minimum, it is attained at $\mathbf{h}$ that satisfies equation

$$\mathbf{D} + \mathbf{H}^\diamond\mathbf{h} = \mathbf{0}. \tag{B.10}$$

Hence $\mathbf{h} = -(\mathbf{H}^\diamond)^{-1}\mathbf{D}$, and the next approximation is $\mathbf{a}' = \mathbf{a} + \mathbf{h}$.

B.2.5. Applying methods of linear algebra. . Introducing matrix notation $\mathbf{g} = (g_1(\mathbf{a}), \dots, g_n(\mathbf{a}))^T$ and

$$\mathbf{J} = \begin{bmatrix} \partial g_1/\partial a_1 & \dots & \partial g_1/\partial a_k \\ \vdots & \ddots & \vdots \\ \partial g_n/\partial a_1 & \dots & \partial g_n/\partial a_k \end{bmatrix} \tag{B.11}$$

we obtain $\mathbf{D} = 2\mathbf{J}^T\mathbf{g}$ and $\mathbf{H}^\diamond = 2\mathbf{J}^T\mathbf{J}$. Therefore, $\mathbf{h}$ is the solution of

$$\mathbf{J}^T\mathbf{J}\mathbf{h} = -\mathbf{J}^T\mathbf{g}. \tag{B.12}$$

This equation corresponds to the overdetermined linear system

$$\mathbf{J}\mathbf{h} \approx -\mathbf{g}, \tag{B.13}$$

[1]However it can (and does!) slow down the rate of convergence, especially if $g_i(\mathbf{a})$ are not so small, see more on that in the end of this section.

which is the classical least squares problem of linear algebra. Its (minimum-norm) solution is

$$\mathbf{h} = -\mathbf{J}^{-}\mathbf{g} = -(\mathbf{J}^T\mathbf{J})^{-}\mathbf{J}^T\mathbf{g} = -(\mathbf{H}^{\diamond})^{-}\mathbf{D}, \tag{B.14}$$

where $(\cdot)^{-}$ denotes the Moore-Penrose pseudoinverse. This formula works whether $\mathbf{H}^{\diamond}$ is singular or not.

REMARK B.1. *One can arrive at* (B.12) *differently. As our goal is to minimize* $\mathcal{G} = \|\mathbf{g}\|^2$*, one can replace* $\mathbf{g}(\mathbf{a}+\mathbf{h})$ *with its linear approximation* $\mathbf{g}(\mathbf{a}) + \mathbf{J}\mathbf{h}$ *and minimize* $\|\mathbf{g}(\mathbf{a}) + \mathbf{J}\mathbf{h}\|^2$ *with respect to* $\mathbf{h}$*; this is exactly the classical least squares problem* (B.13)*, and its solution is given by* (B.14)*.*

B.2.6. Speed of convergence. Many authors assert that the Gauss-Newton method, just like its Newton-Raphson prototype, converges quadratically, but this is not exactly true. The modification of $\mathbf{H}$, however small, does affect the asymptotic speed of convergence, and it becomes linear. Precisely, if $\mathbf{a}^*$ denotes the limit point, then one can only guarantee that

$$\|\mathbf{a}' - \mathbf{a}^*\| < c\|\mathbf{a} - \mathbf{a}^*\|$$

with some $c < 1$. However, the convergence constant c here is proportional to $\mathcal{G}(\mathbf{a}^*)$, hence it is actually close to zero when $g_i(\mathbf{a}^*)$'s are small. That does not make the convergence quadratic, but with some degree of informality it can be described as *nearly* quadratic.

B.3. Levenberg-Marquardt correction

The Gauss-Newton method works well under favorable conditions, in which case its convergence is fast, but it may still overstep the region where $\mathcal{G}$ takes small values, as we noted above, and its performance may be problematic if the 'design matrix' $\mathbf{N} = \mathbf{J}^T\mathbf{J}$ happens to be near-singular.

B.3.1. Augmenting the design matrix. The Levenberg-Marquardt correction aims at eliminating these drawbacks. The design matrix $\mathbf{N}$ is augmented to

$$\mathbf{N}_\lambda = \mathbf{N} + \lambda \mathbf{I}, \tag{B.15}$$

where $\lambda > 0$ is an additional 'control' parameter and $\mathbf{I}$ is the $k \times k$ identity matrix. In other words, the diagonal entries of $\mathbf{N}$ are increased by λ. Then, instead of (B.12), one solves the new system

$$\mathbf{N}_\lambda \mathbf{h} = -\mathbf{J}^T \mathbf{g} \tag{B.16}$$

to determine $\mathbf{h}$. Note that the matrix $\mathbf{N}_\lambda$, with $\lambda > 0$, is always positive definite (while $\mathbf{N}$ is only guaranteed to be positive semi-definite), and in fact all the eigenvalues of $\mathbf{N}_\lambda$ are $\geq \lambda$.

B.3.2. Checkpoint. . After $\mathbf{h}$ has been computed, the algorithm passes through a checkpoint. If the new approximation $\mathbf{a}' = \mathbf{a} + \mathbf{h}$ reduces the value of $\mathcal{G}$, i.e. if $\mathcal{G}(\mathbf{a}') < \mathcal{G}(\mathbf{a})$, it is accepted and λ is decreased by a certain factor α before the next iteration (suppressing the corrective term $\lambda \mathbf{I}$).

Otherwise the new value $\mathbf{a}' = \mathbf{a} + \mathbf{h}$ is rejected, λ is increased by a certain factor β and the augmented normal equations $\mathbf{N}_\lambda \mathbf{h} = -\mathbf{J}^T \mathbf{g}$ are solved again. These recursive attempts continue until the increment $\mathbf{h}$ leads to a smaller value of $\mathcal{G}$. This is bound to happen, since for large λ the method approaches the steepest descent.

B.3.3. Advantages. . In other words, when λ increases, the recomputed vector $\mathbf{h}$ not only gets smaller but also turns (rotates) and aligns somewhat better with the negative gradient vector $-\nabla \mathcal{G}(\mathbf{a})$. As $\lambda \to \infty$, the length of $\mathbf{h}$ approaches zero and its direction converges to that of $-\nabla \mathcal{G}(\mathbf{a})$ (Figure B.1).

We see that the Levenberg-Marquardt correction combines two classical ideas:

(a) The quadratic approximation to the function, which works well in a vicinity of its minimum and yields a fast (nearly quadratic) convergence.

1. Initialize $\mathbf{a}_0$ and λ_0, set $k = 0$.
2. At the current point $\mathbf{a}_k$ compute the vector $\mathbf{g}_k$ and its gradient $\mathbf{J}_k$.
3. Compute $\mathbf{h}_k$ by solving equation $(\mathbf{J}_k^T\mathbf{J}_k + \lambda_k\mathbf{I})\mathbf{h}_k = -\mathbf{J}_k^T\mathbf{g}_k$.
4. Compute the vector $\mathbf{g}'$ at the point $\mathbf{a}' = \mathbf{a}_k + \mathbf{h}_k$.
5. If $\|\mathbf{g}'\|^2 \geq \|\mathbf{g}_k\|^2$, reset $\lambda_k := \beta\lambda_k$ and return to Step 3.
6. Update $\lambda_{k+1} = \alpha\lambda_k$ and $\mathbf{a}_{k+1} = \mathbf{a}_k + \mathbf{h}_k$, increment k, return to Step 2.

Table B.1: Levenberg-Marquardt algorithm.

(b) The steepest descent scheme that ensures reliability in difficult cases.

As the algorithm is based on a reasonable balance between these two principles, it is sometimes referred to as Marquardt compromise.

B.3.4. Practical issues. . In many implementations, the parameter λ is initialized to a small value, e.g. 10^{-3} or 10^{-4}. A common choice for α and β is $\alpha = 0.1$ and $\beta = 10$.

B.4. Trust region

To complete our survey of general minimization schemes, we will describe the trust region method, which currently constitutes 'the state of the art'.

B.4.1. Motivation. . The Levenberg-Marquardt algorithm is flexible enough to avoid obvious pitfalls of the classical minimization schemes and virtually guarantees convergence to a minimum of the objective function. Its drawback, though, is that the control parameter λ is just an abstract variable whose values have no apparent relation to the problem at hand. Therefore it is hard to properly initialize λ. Also, the Levenberg-Marquardt simplistic rules for updating λ (by arbitrarily chosen factors α and β) often cause erratic, suboptimal performance.

A variant of the Levenberg-Marquardt method was developed in the 1970s that fixed the above drawback. It was popularized by Moré in his well written 1978 paper [**?**]. Eventually this method came to be known as *trust region* and was adopted in nearly all standard software packages.We describe its main ideas here, referring to [**?**] for further technical details.

B.4.2. Geometric description of Levenberg-Marquardt. The Levenberg-Marquardt method can be interpreted geometrically as follows. Recall that an iteration of the Gauss-Newton method consists in minimization of the quadratic function

$$\mathcal{Q}(\mathbf{a}+\mathbf{h}) = \|\mathbf{g}(\mathbf{a}) + \mathbf{J}\mathbf{h}\|^2,$$

which approximates the given function $\mathcal{G}(\mathbf{a}+\mathbf{h})$ in a vicinity of the current iteration $\mathbf{a}$. Figure B.1 shows the contour map (the sets of level curves) of $\mathcal{Q}(\mathbf{a}+\mathbf{h})$ in the 2D case; the level curves are concentric ellipses.

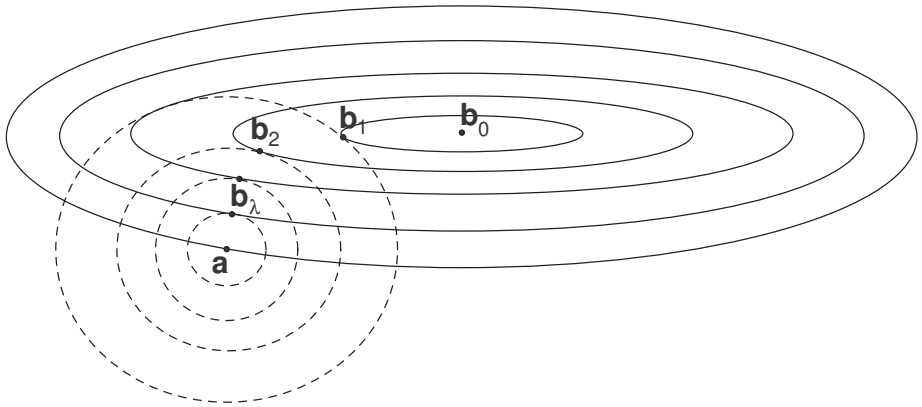

Figure B.1: The level curves of $\mathcal{Q}(\mathbf{a}+\mathbf{h})$ are concentric ellipses (solid ovals), the boundaries of trust regions are dashed circles around $\mathbf{a}$.

The 'pure' Gauss-Newton step (i.e. when $\lambda = 0$) lands at the minimum of $\mathcal{Q}(\mathbf{a}+\mathbf{h})$, i.e. at the ellipses' center $\mathbf{b}_0$. When $\lambda > 0$, the Levenberg-Marquardt step (the solution of (B.16)) lands at some other point, $\mathbf{b}_\lambda = \mathbf{a}+\mathbf{h}$, closer to $\mathbf{a}$. At that point

we have, according to (B.16),

$$\operatorname{grad} \mathcal{Q}(\mathbf{a}+\mathbf{h}) = -\lambda \mathbf{h}.$$

Accordingly, the vector $\mathbf{h} = \mathbf{b}_\lambda - \mathbf{a}$ crosses the level curve of $\mathcal{Q}(\mathbf{a}+\mathbf{h})$ passing through $\mathbf{b}_\lambda$, orthogonally. This means that $\mathbf{b}_\lambda$ provides the minimum of the function $\mathcal{Q}(\mathbf{a}+\mathbf{h})$ *restricted* to the ball $\mathbb{B}(\mathbf{a}, \Delta)$ whose center is $\mathbf{a}$ and whose radius is $\Delta = \|\mathbf{h}\|$. In other words, the Levenberg-Marquardt step with $\lambda > 0$ minimizes the restriction of the quadratic approximation $\mathcal{Q}(\mathbf{a}+\mathbf{h})$ to a certain ball around the current approximation $\mathbf{a}$ whose radius Δ is determined by λ.

As $\lambda > 0$ grows, the point $\mathbf{b}_\lambda$ moves closer to $\mathbf{a}$, and Δ decreases. At the same time the vector $\mathbf{h} = \mathbf{b}_\lambda - \mathbf{a}$ rotates and makes a larger angle with the level curves. In the limit $\lambda \to \infty$, the ball $\mathbb{B}(\mathbf{a}, \Delta)$ shrinks to the point $\mathbf{a}$, i.e. $\Delta \to 0$, and the vector $\mathbf{h}$ ultimately aligns with the direction of the steepest descent.

B.4.3. Replacing the control parameter. . Therefore, there is a one-to-one correspondence between $\lambda > 0$ and $\Delta \in (0, \|\mathbf{b}_0 - \mathbf{a}\|)$, hence one can use Δ, instead of λ, as a control parameter, i.e. adjust Δ from iteration to iteration. As Δ has a clear meaning (further explained below), its initialization and its update at each iteration can be done in a more sensible way than the way λ is treated in the Levenberg-Marquardt scheme.

One benefit of dealing with Δ is that one can directly control the region in the parameter space where the quadratic approximation $\mathcal{Q}(\mathbf{a}+\mathbf{h})$ is minimized. It is called the *trust region*, the idea behind it is that we minimize the approximation $\mathcal{Q}(\mathbf{a}+\mathbf{h})$ where it can be *trusted* and do not go too far where the approximation is not deemed reliable.

B.4.4. New updating rules. . This interpretation of Δ also leads to the following update strategy. After a step $\mathbf{h}$ is computed, one finds the ratio

$$r = \frac{\text{Ared}}{\text{Pred}} = \frac{\mathcal{G}(\mathbf{a}) - \mathcal{G}(\mathbf{a}+\mathbf{h})}{\mathcal{G}(\mathbf{a}) - \mathcal{Q}(\mathbf{a}+\mathbf{h})}$$

1. Initialize $\mathbf{a}_0$ and Δ_0, set $k = 0$.
2. At the current point $\mathbf{a}_k$ compute the vector $\mathbf{g}_k$ and its gradient $\mathbf{J}_k$.
3. For the current $\mathbf{a}_k$ and Δ_k, determine λ_k.
4. Compute $\mathbf{h}_k$ by solving equation $(\mathbf{J}_k^T\mathbf{J}_k + \lambda_k\mathbf{I})\mathbf{h}_k = -\mathbf{J}_k^T\mathbf{g}_k$.
5. Find the ratio $r_k = \text{Ared}_k/\text{Pred}_k$.
6. If $r_k < 0$, reset $\Delta_k := \frac{1}{2}\Delta_k$ and return to Step 3.
7. If $r_k < 0.25$, update $\Delta_{k+1} = \frac{1}{2}\Delta_k$; if $r > 0.75$, update $\Delta_{k+1} = 2\Delta_k$.
8. Update $\mathbf{a}_{k+1} = \mathbf{a}_k + \mathbf{h}_k$, increment k, and return to Step 2.

Table B.2: Trust region algorithm.

of the *Actual reduction*, Ared, and the *Predicted reduction*, Pred, of the objective function. One should note that $\mathcal{G}(\mathbf{a}) = \mathcal{Q}(\mathbf{a})$, hence the denominator is always positive, but the numerator is positive only if the actual reduction occurs, in which case of course we should accept the step $\mathbf{h}$.

However, we adjust Δ based on the value of r. A common strategy is as follows. If $r < 0.25$, then the quadratic approximation is not deemed quite reliable (despite the actual decrease of the objective function) and we *reduce* the size Δ before the next iteration (say, by $\Delta := \Delta/2$). Only if $r > 0.75$, then the approximation is regarded as sufficiently accurate and we *increase* Δ (by $\Delta := 2\Delta$). In the intermediate case $0.25 \leq r \leq 0.75$ we leave Δ *unchanged*. Note that these rules are quite different (in a sense, more conservative) than the simplistic rules of updating λ in the previous section!